MATHEMATICS OF COMPUTATION
Volume 86, Number 303, January 2017, Pages 315–343
http://dx.doi.org/10.1090/mcom/3104
Article electronically published on April 13, 2016

A SEQUENTIAL UPDATING SCHEME OF THE LAGRANGE MULTIPLIER FOR SEPARABLE CONVEX PROGRAMMING

YU-HONG DAI, DEREN HAN, XIAOMING YUAN, AND WENXING ZHANG

Abstract. The augmented Lagrangian method (ALM) is a benchmark for solving convex minimization problems with linear constraints. Solving the augmented subproblems over the primal variables can be regarded as sequentially providing inputs for updating the Lagrange multiplier (i.e., the dual variable). We consider the separable case of a convex minimization problem where its objective function is the sum of more than two functions without coupled variables. When applying the ALM to this case, at each iteration we can (sometimes must) split the resulting augmented subproblem in order to generate decomposed subproblems which are often easy enough to have closed-form solutions. But the decomposition of primal variables only provides less accurate inputs for updating the Lagrange multiplier, and it points out the lack of convergence for such a decomposition scheme. To remedy this difficulty, we propose to update the Lagrange multiplier sequentially once each decomposed subproblem over the primal variables is solved. This scheme updates both the primal and dual variables in Gauss-Seidel fashion. In addition to the exact version which is useful enough for the case where the functions in the objective are all simple such that the decomposed subproblems all have closed-form solutions, we investigate an inexact version of this scheme which allows the decomposed subproblems to be solved approximately subject to certain inexactness criteria. Some preliminary numerical results when the proposed scheme is respectively applied to an image decomposition problem and an allocation problem are reported.

1. Introduction

Let us start with the canonical convex minimization model with linear constraints

$$\min \{\theta(x) \mid Ax = b,\ x \in \mathcal{X}\}, \tag{1.1}$$

where $\theta : \mathcal{R}^n \to \mathcal{R}$ is a closed proper convex function (could be nonsmooth), $A \in \mathcal{R}^{l\times n}$, $b \in \mathcal{R}^l$ and $\mathcal{X} \subseteq \mathcal{R}^n$ is a nonempty closed convex set. A benchmark for

Received by the editor June 29, 2013 and, in revised form, August 16, 2014 and July 27, 2015.
2010 *Mathematics Subject Classification.* Primary 90C25, 65K10, 94A08, 68W10.
Key words and phrases. Convex programming, augmented Lagrangian method, splitting method, method of multipliers, image processing.
The first author was partially supported by the China National Funds for Distinguished Young Scientists Grant 11125107, the Key Project of Chinese National Programs for Fundamental Research and Development Grant 2015CB856000, NSFC Grant 11331012, and the CAS Program for Cross & Cooperative Team of the Science & Technology Innovation.
The second author was supported by a project funded by PAPD of Jiangsu Higher Education Institutions and the NSFC grants 11371197, 11431002.
The third author was supported by the General Research Fund of Hong Kong: HKBU203613.
The fourth author was supported by the NSFC grant 11301055.

solving (1.1) is the augmented Lagrangian method (ALM) proposed in [28,39]:

$$(1.2)\qquad \begin{cases} x^{k+1} = \arg\min\left\{\mathcal{L}_H(x,\lambda^k) \mid x \in \mathcal{X}\right\}, \\ \lambda^{k+1} = \lambda^k - H(Ax^{k+1} - b), \end{cases}$$

where $\mathcal{L}_H(x,\lambda)$ denotes the augmented Lagrangian function of (1.1):

$$(1.3)\qquad \mathcal{L}_H(x,\lambda) := \theta(x) - \lambda^T(Ax-b) + \tfrac{1}{2}\|Ax-b\|_H^2$$

with $\lambda \in \mathcal{R}^l$ the Lagrange multiplier and $H \in \mathcal{R}^{l\times l}$ a positive definite matrix playing the role of a penalty parameter. Usually, we can simply take $H = \beta \cdot I_l$ where I_l is the identity matrix in $\mathcal{R}^{l\times l}$ and $\beta > 0$ is a scalar. In [41], the ALM was explained as an application of the proximal point algorithm (PPA) in [33] to the dual of (1.1), and some nice convergence properties were derived therein. It's worth mentioning that the convergence of ALM was established in the sense that the sequence $\{\lambda^k\}$ is convergent to λ^*, an optimal solution of the dual of (1.1). Thus, we can also understand the ALM iterative scheme (1.2) as that solving the augmented subproblem over the primal variable is to provide an input (more specifically, the gradient $-H(Ax^{k+1}-b)$) for updating the dual variable λ. For more detail we refer the reader to [41], where the ALM was actually explained as an application of the gradient ascent method to the dual of (1.1).

For applications in various areas, the abstract model (1.1) can usually be specified as concrete models with such a separable structure that the objective consists of multiple individual objectives and each of them relies on its own decision variables. That is, the objective function is the sum of more than one function without coupled variables. But, all the decision variables are linked via some linear constraints (e.g., resource constraints). For many areas such as image processing and statistical learning, the objective function of (1.1) is composed of a data-fidelity term and several regularization terms. We thus consider the most general separable form of (1.1):

$$(1.4)\qquad \min\left\{\sum_{i=1}^m \theta_i(x_i) \;\middle|\; \sum_{i=1}^m A_i x_i = b,\ x_i \in \mathcal{X}_i,\ i = 1,2,\cdots,m\right\},$$

where $\theta_i : \mathcal{R}^{n_i} \to \mathcal{R}$ are all closed proper convex functions (could be nonsmooth); $\mathcal{X}_i \subseteq \mathcal{R}^{n_i}$ are nonempty closed convex sets; $A_i \in \mathcal{R}^{l\times n_i}$; $b \in \mathcal{R}^l$; and $\sum_{i=1}^m n_i = n$. Our discussion is mainly inspired by the scenario where it is meaningful to individually take advantage of θ_i's properties; we thus do not consider any algorithmic design that ignores the separable structure of (1.4). Each A_i is further assumed to be full column rank. The solution set of (1.4) is assumed to be nonempty.

Let the augmented Lagrangian function of (1.4) be

$$(1.5)\quad \mathcal{L}_H(x_1,x_2,\cdots,x_m,\lambda) := \sum_{i=1}^m \theta_i(x_i) - \lambda^T\left(\sum_{i=1}^m A_i x_i - b\right) + \frac{1}{2}\left\|\sum_{i=1}^m A_i x_i - b\right\|_H^2$$

with λ and H having the same meanings as in (1.3). Denoting $\mathbf{x} := (x_1,x_2,\cdots,x_m)$ and $\mathcal{X} := \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_m$, the iterative scheme of ALM for (1.4) reads as

$$(1.6)\qquad \begin{cases} \mathbf{x}^{k+1} = \arg\min\left\{\mathcal{L}_H(\mathbf{x},\lambda^k) \mid \mathbf{x} \in \mathcal{X}\right\}, \\ \lambda^{k+1} = \lambda^k - H(\sum_{i=1}^m A_i x_i^{k+1} - b). \end{cases}$$

Conceptually, this straightforward application of the ALM is capable of offering an accurate input for updating the Lagrange multiplier provided that the minimization problem over all primal variables can be solved exactly at each iteration.

This ideal purpose of solving the $\mathbf{x}$-subproblem in (1.6) exactly, however, is usually not achievable. We thus have to make a trade-off between the solvability of the $\mathbf{x}$-subproblem in (1.6) and its accuracy for the updating of the Lagrange multiplier. In fact, many concrete applications of (1.4) share the common feature that each function in the objective may have some specific properties. Thus, a natural idea to improve the solvability of the $\mathbf{x}$-subproblem in (1.6) is to decompose it into m smaller subproblems so that each of these decomposed subproblems only involves one $\theta_i(x_i)$ and thus the properties of θ_i's could be used effectively in algorithmic design. One success is the Douglas-Rachford alternating direction method of multipliers (ADMM) in [18] for solving the special case of (1.4) with $m = 2$. More specifically, the ADMM decomposes the ALM subproblem at each iteration into two smaller subproblems in Gauss-Seidel fashion, and its iterative scheme reads as

$$\begin{cases} x_1^{k+1} = \arg\min\big\{\mathcal{L}_H(x_1, x_2^k, \lambda^k) \mid x_1 \in \mathcal{X}_1\big\}, \\ x_2^{k+1} = \arg\min\big\{\mathcal{L}_H(x_1^{k+1}, x_2, \lambda^k) \mid x_2 \in \mathcal{X}_2\big\}, \\ \lambda^{k+1} = \lambda^k - H\big(A_1x_1^{k+1} + A_2x_2^{k+1} - b\big), \end{cases} \tag{1.7}$$

with

$$\mathcal{L}_H(x_1, x_2, \lambda) := \theta_1(x_1) + \theta_2(x_2) - \lambda^T(A_1x_1 + A_2x_2 - b) + \frac{1}{2}\|A_1x_1 + A_2x_2 - b\|_H^2.$$

The ADMM scheme (1.7) enjoys the advantage that the generated subproblems could have closed-form solutions provided that θ_1 and θ_2 are special enough; thus it is impressively efficient for some applications arising in areas such as image processing, compressive sensing, computer vision, statistical learning; see, e.g., [5, 10, 15] for some recent review papers in which a number of novel applications of ADMM and some important references can be found. Although the decomposed x_1- and x_2-subproblems in (1.7) provide an approximation to the all-in-one $\mathbf{x}$-subproblem in (1.6) with $\mathbf{x} = (x_1, x_2)$, the convergence of ADMM is still guaranteed; see, e.g., [12, 13, 26, 27]. However, when the ADMM scheme (1.7) is extended to the general case of (1.4) where $m \geq 3$, according to [6], the convergence may not hold. That is, the iterative scheme

$$\begin{cases} x_1^{k+1} = \arg\min\big\{\mathcal{L}_H(x_1, x_2^k, \cdots, x_m^k, \lambda^k) \mid x_1 \in \mathcal{X}_1\big\}, \\ \cdots \\ x_i^{k+1} = \arg\min\big\{\mathcal{L}_H(x_1^{k+1}, \cdots, x_{i-1}^{k+1}, x_i, x_{i+1}^k, \cdots, x_m^k, \lambda^k) \mid x_i \in \mathcal{X}_i\big\}, \\ \cdots \\ x_m^{k+1} = \arg\min\big\{\mathcal{L}_H(x_1^{k+1}, \cdots, x_{m-1}^{k+1}, x_m, \lambda^k) \mid x_m \in \mathcal{X}_m\big\}, \\ \lambda^{k+1} = \lambda^k - H\big(\sum_{j=1}^m A_jx_j^{k+1} - b\big) \end{cases} \tag{1.8}$$

is not necessarily convergent under our setting of (1.4). We refer to [20, 29] for the convergence of this ADMM's extension (1.8) under additional assumptions. The lack of convergence of (1.8) can be intuitively understood as that the $\mathbf{x}$-subproblem in (1.6) is decomposed more than twice and thus it is not able to provide inputs that are accurate enough for the updating of the Lagrange multiplier.

With the understanding of treating the output of the decomposed ALM subproblems as inputs for updating the Lagrange multiplier, one idea to overcome the lack of convergence of (1.8) is to update the Lagrange multiplier λ sequentially right after each primal variable x_i is updated. Let us start with the case of (1.4) where $m = 2$. For this case, a revised version of (1.7) which updates the Lagrange multiplier sequentially is

$$
(1.9) \qquad \begin{cases} x_1^{k+1} &= \arg\min \big\{ \mathcal{L}_H(x_1, x_2^k, \lambda^k) \mid x_1 \in \mathcal{X}_1 \big\}, \\ \lambda^{k+\frac{1}{2}} &= \lambda^k - H(A_1 x_1^{k+1} + A_2 x_2^k - b), \\ x_2^{k+1} &= \arg\min \big\{ \mathcal{L}_H(x_1^{k+1}, x_2, \lambda^{k+\frac{1}{2}}) \mid x_2 \in \mathcal{X}_2 \big\}, \\ \lambda^{k+1} &= \lambda^{k+\frac{1}{2}} - H\big(A_1 x_1^{k+1} + A_2 x_2^{k+1} - b\big). \end{cases}
$$

In (1.9), both the primal and dual variables are updated in Gauss-Seidel fashion, and one may expect that (1.9) achieves faster convergence than (1.7) because the Lagrange multiplier is updated more timely. In the literature, the scheme (1.9) is also called symmetric ADMM since the Lagrange multiplier is updated twice at each iteration symmetrically with respect to the variables x_1 and x_2. In [14, 16], it was analyzed that the scheme (1.9) is actually an application of the Peaceman-Rachford splitting method (PRSM) in [30, 38], a very standard method in earlier PDE literature. In [17], it was stated that "very often PRSM is faster than the Douglas-Rachford splitting method (DRSM) in [9, 30]" (note that ADMM is only an application of DRSM, as analyzed in [12]), and the efficiency of PRSM has been verified numerically in existing literature such as [3, 16]. But in [17, 30], it was also stated that "PRSM is less stable than DRSM", and the convergence analysis of PRSM is in general much harder than DRSM. A counterexample illustrating that the sequence generated by PRSM could only stay with a constant distance to the solution set and thus is divergent was given in [8], and in [24], the reason for divergence of (1.9) is explained from the perspective of contraction methods.

If we extend the iterative scheme (1.9) straightforwardly to the general case (1.4) with a generic $m \geq 3$, the scheme reads as

(1.10)

$$
\begin{cases} x_1^{k+1} &= \arg\min \big\{ \mathcal{L}_H(x_1, x_2^k, \cdots, x_m^k, \lambda^k) \mid x_1 \in \mathcal{X}_1 \big\}, \\ \lambda^{k+\frac{1}{m}} &= \lambda^k - H\big(A_1 x_1^{k+1} + \sum_{j=2}^m A_j x_j^k - b\big), \\ \cdots & \\ x_i^{k+1} &= \arg\min \big\{ \mathcal{L}_H(x_1^{k+1}, \cdots, x_{i-1}^{k+1}, x_i, x_{i+1}^k, \cdots, x_m^k, \lambda^{k+\frac{i-1}{m}}) \mid x_i \in \mathcal{X}_i \big\}, \\ \lambda^{k+\frac{i}{m}} &= \lambda^{k+\frac{i-1}{m}} - H\big(\sum_{j=1}^i A_j x_j^{k+1} + \sum_{j=i+1}^m A_j x_j^k - b\big), \\ \cdots & \\ \lambda^{k+\frac{m-1}{m}} &= \lambda^{k+\frac{m-2}{m}} - H\big(\sum_{j=1}^{m-1} A_j x_j^{k+1} + A_m x_m^k - b\big), \\ x_m^{k+1} &= \arg\min \big\{ \mathcal{L}_H(x_1^{k+1}, \cdots, x_{m-1}^{k+1}, x_m, \lambda^{k+\frac{m-1}{m}}) \mid x_m \in \mathcal{X}_m \big\}, \\ \lambda^{k+1} &= \lambda^{k+\frac{m-1}{m}} - H\big(\sum_{j=1}^m A_j x_j^{k+1} - b\big). \end{cases}
$$

Note that the scheme (1.10) is a natural extension of (1.9) for the case of (1.4) where $m \geq 3$, and it can be regarded as embedding a full Gauss-Seidel decomposition for both the primal and dual variables into the scheme (1.8). However, according to [23], the scheme (1.10) is also not necessarily convergent under the setting of our discussion.

To obtain a scheme with provable convergence based on the scheme (1.10), our strategy is to supplement a simple linear combination between the previous iterate and the output of (1.10). Thus, the new iterate is generated in a prediction-correction fashion where the output of (1.10) can be regarded as a predictor of the new iterate. Moreover, in our algorithmic design, we make the effort to simplify the update of λ before individual primal variables are solved at each iteration. That is, when updating $\lambda^{k+\frac{i}{m}}$, we ignore the quantities $A_j x_j^{k+1}$ with $j = 1, 2, \cdots, i-1$

and $A_j x_j^k$ with $j = i+1, \cdots, m$ and simplify it as

$$\lambda^{k+\frac{i}{m}} = \lambda^{k+\frac{i-1}{m}} + \mu H\big(A_i x_i^k - A_i x_i^{k+1}\big), \quad i = 1, 2, \cdots, m-1,$$

where the parameter $\mu \geq 1$ will be explained later. Furthermore, for the crossing and quadratic terms in the objective of the x_i-subproblem, we also ignore the quantities $A_j x_j^{k+1}$ with $j = 1, 2, \cdots, i-1$ and $A_j x_j^k$ with $j = i+1, \cdots, m$. For example, the quadratic term

$$\tfrac{1}{2}\left\|\textstyle\sum_{j=1}^{i-1} A_j x_j^{k+1} + A_i x_i + \sum_{j=i+1}^{m} A_j x_j^k\right\|_H^2$$

in the augmented Lagrangian function of (1.10) is altered to

$$\frac{\mu}{2}\|A_i x_i - A_i x_i^k\|_H^2.$$

Theoretically, we can understand this simplification of the updating scheme as a proximal point regularization where the proximity of the updated Lagrange multiplier to the previous one is controlled by the difference of the just-obtained primal variable x_i^{k+1}, and the parameter μ plays the role of controlling the weight of proximity. Algorithmically, this succinctness consideration might be of interest in the sense of saving storage and multiplication computation — thinking about huge-scale applications of (1.4) where the multiplication in the form of $A_i x_i^k$ might also be unaffordable. Finally, for the last time updating λ^{k+1} when all the primal variables x_i $(i = 1, 2, \cdots, m)$ are updated at each iteration and the Lagrange multiplier is updated for $(m-1)$ times already, we coordinate all the updates and execute a regular updating step for the Lagrange multiplier. That is, the λ^{k+1} step in (1.10) is retained.

The rest of this paper is organized as follows. Some necessary preliminaries for further analysis are provided in Section 2. In Section 3, we delineate the sequential updating scheme of the Lagrange scheme for solving (1.4), and in Section 4, we analyze its global convergence. In Section 5, we present an inexact version of the sequential updating scheme of the Lagrange scheme and establish its convergence. Then, in Section 6, we report some numerical results when the sequential updating scheme of the Lagrange scheme is applied to solve image decomposition and allocation problems. Finally, some concluding remarks are given in Section 7.

2. Preliminaries

In this section, we summarize some basic concepts and preliminaries that will be useful in the subsequent sections.

For any vector $x \in \mathcal{R}^n$, we denote by $\|x\|_p := \left(\sum_{i=1}^n |x_i|^p\right)^{1/p}$ the p-norm. In particular, $\|x\| := \|x\|_2$. Let $\mathbf{1}$ be the vector with all elements being one and let I represent the identity matrix with appropriate dimension. For any vector $x = (x_1, x_2, \cdots, x_n) \in \mathcal{R}^n$, diag$(x)$ represents a diagonal matrix whose diagonal elements are x_i's. Let $\mathcal{S}^n$ be the set of n-by-n symmetric matrices, and $\mathcal{S}_{++}^n$ be the set of positive definite matrices in $\mathcal{S}^n$. For any $H \in \mathcal{S}_{++}^n$, $\|x\|_H := \sqrt{x^T H x}$ denotes the matrix norm of x. The smallest and largest eigenvalues of a matrix H are denoted by $\text{eig}_{\min}(H)$ and $\text{eig}_{\max}(H)$, respectively.

Let $\theta : \mathcal{R}^n \to (-\infty, +\infty]$. If the domain of θ which is denoted by dom$\theta := \{x \in \mathcal{R}^n \mid \theta(x) < +\infty\}$ is nonempty, then θ is said to be proper. Let $\Gamma_0(\mathcal{R}^n)$ denote the set of proper convex functions from $\mathcal{R}^n$ to $(-\infty, +\infty]$. For any $\theta \in \Gamma_0(\mathcal{R}^n)$, the proximity function of θ, denoted by prox_θ, is defined as (see, e.g., [7,35])

$$\text{prox}_\theta(x) := \arg\min_y \left\{\theta(y) + \tfrac{1}{2}\|y-x\|^2\right\}, \quad \forall x \in \mathcal{R}^n. \tag{2.1}$$

The subdifferential of θ at x, denoted by $\partial\theta(x) : \mathcal{R}^n \to 2^{\mathcal{R}^n}$, is defined as

$$\partial\theta(x) := \left\{\xi \in \mathcal{R}^n \mid \theta(y) \geq \theta(x) + \xi^T(y-x), \ \ \forall y \in \mathcal{R}^n\right\}.$$

Let $\mathbf{u} := (x_1, x_2, \cdots, x_m, \lambda)$ and $\mathcal{U} := \mathcal{X}_1 \times \cdots \times \mathcal{X}_m \times \mathcal{R}^l$. According to the first-order optimality condition, (1.4) can be characterized by the following variational inequality problem (VIP): Find $\mathbf{u}^* \in \mathcal{U}$ such that for some $\xi_i^* \in \partial\theta_i(x_i^*)$ $(i = 1, 2, \cdots, m)$, the inequalities

$$\text{(VIP)} \qquad \left\{\begin{array}{l} (x_1 - x_1^*)^T\{\xi_1^* - A_1^T\lambda^*\} \geq 0, \\ (x_2 - x_2^*)^T\{\xi_2^* - A_2^T\lambda^*\} \geq 0, \\ \qquad \cdots\cdots \\ (x_m - x_m^*)^T\{\xi_m^* - A_m^T\lambda^*\} \geq 0, \\ (\lambda - \lambda^*)^T(\sum_{i=1}^m A_i x_i^* - b) \geq 0, \end{array}\right. \qquad \forall \mathbf{u} \in \mathcal{U},$$

are satisfied. By denoting

$$\mathbf{F}(\mathbf{u}) := \begin{pmatrix} \partial\theta_1(x_1) - A_1^T\lambda \\ \partial\theta_2(x_2) - A_2^T\lambda \\ \cdots\cdots \\ \partial\theta_m(x_m) - A_m^T\lambda \\ \sum_{i=1}^m A_i x_i - b \end{pmatrix}, \tag{2.2}$$

the above VIP can be written as: Find $\mathbf{u}^* \in \mathcal{U}$ such that for some $\vartheta^* \in \mathbf{F}(\mathbf{u}^*)$,

$$(\mathbf{u} - \mathbf{u}^*)^T\vartheta^* \geq 0, \quad \forall\, \mathbf{u} \in \mathcal{U}. \tag{2.3}$$

We denote by $\mathcal{U}^*$ the set of $\mathbf{u}^*$ satisfying (2.3). Then, $\mathcal{U}^*$ is nonempty if the solution set of (1.4) is nonempty (see, e.g., [11]).

Thanks to an anonymous referee, we will use the following lemma (the deterministic version of Theorem 1 in [40]; see also [4]) to prove the convergence for the proposed algorithm.

Lemma 2.1 ([40, Th. 1]). *Let z_k, β_k, ξ_k and ζ_k be nonnegative sequences such that*

$$z_{k+1} \leq z_k(1+\beta_k) + \xi_k - \zeta_k.$$

If $\sum_{k=1}^\infty \beta_k < \infty$ and $\sum_{k=1}^\infty \xi_k < \infty$, then we have

i. $\lim_{k\to\infty} z_k$ *exists and is finite;*
ii. $\sum_{k=1}^\infty \zeta_k < \infty$.

3. A sequential updating scheme of the Lagrange multiplier for (1.4)

In this section, we present a *sequential updating scheme of the Lagrange multiplier* for solving (1.4).

With a given $H \in \mathcal{S}^n_{++}$ and $\mu \geq 1$, we define two $(n+l)$-by-$(n+l)$ matrices as

$$M := \begin{pmatrix} \mu A_1^T H A_1 & 0 & \cdots & 0 & 0 \\ \mu A_2^T H A_1 & \mu A_2^T H A_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mu A_m^T H A_1 & \mu A_m^T H A_2 & \cdots & \mu A_m^T H A_m & 0 \\ 0 & 0 & \cdots & 0 & H^{-1} \end{pmatrix}, \tag{3.1}$$

$$N := \begin{pmatrix} 2\mu A_1^T H A_1 & \mu A_1^T H A_2 & \cdots & \mu A_1^T H A_m & A_1^T \\ \mu A_2^T H A_1 & 2\mu A_2^T H A_2 & \cdots & \mu A_2^T H A_m & A_2^T \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mu A_m^T H A_1 & \mu A_m^T H A_2 & \cdots & 2\mu A_m^T H A_m & A_m^T \\ A_1 & A_2 & \cdots & A_m & 2H^{-1} \end{pmatrix}. \tag{3.2}$$

These two matrices help us present theoretical analysis in compact notation later. Note that when A_i's are assumed to be full column rank and $\mu \geq 1$, it is easy to see that the matrix N defined in (3.2) is positive definite.

The sequential updating scheme of the Lagrange multiplier for (1.4) is summarized as Algorithm 1.

Algorithm 1: Sequential updating scheme of the Lagrange multiplier for (1.4)

```
    Input: Choose $\mu \geq 1$, $\gamma \in (0,2)$, $\epsilon > 0$, $\mathbf{u}^0 = (x_1^0, x_2^0, \cdots, x_m^0, \lambda^0) \in \mathcal{U}$,
           $\tilde{\mathbf{u}}^0 = (\tilde{x}_1^0, \tilde{x}_2^0, \cdots, \tilde{x}_m^0, \tilde{\lambda}^0) \in \mathcal{U}$, $H \in \mathcal{S}^l_{++}$ and $G \in \mathcal{S}^{n+l}_{++}$. Set $k = 0$.
 1  while $\max\left\{\max_{1\leq i\leq m} \|A_i x_i^k - A_i \tilde{x}_i^k\|, \|\lambda^k - \tilde{\lambda}^k\|\right\} > \epsilon$ do
 2      for $i = 1, 2, \cdots, m$ do
 3          if $i = 1$ then
 4              $\lambda^{k+\frac{i-1}{m}} = \lambda^k - H\left(\sum_{j=1}^m A_j x_j^k - b\right)$.
 5          else
 6              $\lambda^{k+\frac{i-1}{m}} = \lambda^{k+\frac{i-2}{m}} + \mu H(A_{i-1}x_{i-1}^k - A_{i-1}\tilde{x}_{i-1}^k)$.
 7          end
 8          $\tilde{x}_i^k = \arg\min_{x_i \in \mathcal{X}_i} \left\{\theta_i(x_i) - (\lambda^{k+\frac{i-1}{m}})^T A_i x_i + \frac{\mu}{2}\|A_i x_i - A_i x_i^k\|_H^2\right\}$.
 9      end
10      $\tilde{\lambda}^k = \lambda^k - H\left(\sum_{i=1}^m A_i \tilde{x}_i^k - b\right)$.
11      $\alpha_k = \|\mathbf{u}^k - \tilde{\mathbf{u}}^k\|_N^2 / \left(2\|G^{-1}M(\mathbf{u}^k - \tilde{\mathbf{u}}^k)\|_G^2\right)$.
12      $\mathbf{u}^{k+1} = \mathbf{u}^k - \gamma\alpha_k G^{-1}M(\mathbf{u}^k - \tilde{\mathbf{u}}^k)$.
13      $k = k + 1$.
14  end
```

Remark 3.1. We abuse slightly the notation "$k + \frac{0}{m}$" in Algorithm 1; it is not k by calculation. That is, $\lambda^{k+\frac{0}{m}}$ and λ^k are different.

Remark 3.2. In general, the $\tilde{x}_i$-subproblems require inner iterations to pursue approximate solutions, and this is the motivation for considering an inexact version of Algorithm 1 in Section 4, which allows us to solve these subproblems approximately subject to certain approximate criteria. Even when these subproblems are easy enough to have closed-form solutions for some specific applications (e.g., the example in Section 6), these subproblems usually dominate the computation of Algorithm 1, and the computation of the correction steps is often relatively much cheaper (and it is completely computable and requires no optimization).

Remark 3.3. For the correction step in Line 12, the α_k is a step size along the descent direction $-G^{-1}M(\mathbf{u}^k - \tilde{\mathbf{u}}^k)$. We choose it judiciously according to the formula in Line 11 because it helps us ensure the convergence of Algorithm 1 (see Lemma 4.2). For the parameter γ, it is a relaxation (or scaling) factor, and the restriction $\gamma \in (0,2)$ is to ensure the contraction of Algorithm 1's sequence (see Theorem 4.3) theoretically and also to accelerate the convergence empirically. Our experience is to choose an aggressive value close to 2, e.g., $\gamma \approx 1.5$. For the matrix $G \in \mathcal{S}^{n+l}_{++}$, we can simply choose it as the identity matrix in practice.

The following lemma shows that it is reasonable to use the stopping criterion in Line 1 of Algorithm 1.

Lemma 3.1. *If* $A_i x_i^k = A_i \tilde{x}_i^k$ $(i = 1, 2, \cdots, m)$ *and* $\lambda^k = \tilde{\lambda}^k$, *then* $\tilde{\mathbf{u}}^k = (\tilde{x}_1^k, \tilde{x}_2^k, \cdots, \tilde{x}_m^k, \tilde{\lambda}^k)$ *is a solution of VIP* (2.3).

Proof. The optimality condition of the $\tilde{x}_i^k$-subproblem in Line 8 is to find $\tilde{x}_i^k \in \mathcal{X}_i$ such that for some $\tilde{\xi}_i^k \in \partial\theta_i(\tilde{x}_i^k)$ the following inequality holds:

$$(x_i' - \tilde{x}_i^k)^T \left\{ \tilde{\xi}_i^k - A_i^T \lambda^{k+\frac{i-1}{m}} + \mu A_i^T H A_i (\tilde{x}_i^k - x_i^k) \right\} \geq 0, \quad \forall x_i' \in \mathcal{X}_i.$$

Line 10 in Algorithm 1 can also be written as

$$(\lambda' - \tilde{\lambda}^k)^T \left\{ \sum_{i=1}^m A_i \tilde{x}_i^k - b + H^{-1}(\tilde{\lambda}^k - \lambda^k) \right\} \geq 0, \quad \forall \lambda' \in \mathcal{R}^l.$$

Consequently, the VIP characterization of Algorithm 1 is to find $\tilde{\mathbf{u}}^k = (\tilde{x}_1^k, \tilde{x}_2^k, \cdots, \tilde{x}_m^k, \tilde{\lambda}^k)$ such that for some $\tilde{\xi}_i^k \in \partial\theta_i(\tilde{x}_i^k)$ $(i = 1, 2, \cdots, m)$,

$$\text{(3.3)} \quad \begin{pmatrix} x_1' - \tilde{x}_1^k \\ x_2' - \tilde{x}_2^k \\ \cdots \\ x_m' - \tilde{x}_m^k \\ \lambda' - \tilde{\lambda}^k \end{pmatrix}^T \begin{pmatrix} \tilde{\xi}_1^k - A_1^T \lambda^{k+\frac{0}{m}} + \mu A_1^T H A_1 (\tilde{x}_1^k - x_1^k) \\ \tilde{\xi}_2^k - A_2^T \lambda^{k+\frac{1}{m}} + \mu A_2^T H A_2 (\tilde{x}_2^k - x_2^k) \\ \cdots\cdots \\ \tilde{\xi}_m^k - A_m^T \lambda^{k+\frac{m-1}{m}} + \mu A_m^T H A_m (\tilde{x}_m^k - x_m^k) \\ \sum_{i=1}^m A_i \tilde{x}_i^k - b + H^{-1}(\tilde{\lambda}^k - \lambda^k) \end{pmatrix} \geq 0, \ \forall \mathbf{u}' \in \mathcal{U}.$$

Combining the schemes of $\lambda^{k+\frac{i-1}{m}}$ and $\tilde{\lambda}^k$ in Lines 4, 6 and 10, we deduce that

(3.4)
$$\lambda^{k+\frac{i-1}{m}} = \begin{cases} \tilde{\lambda}^k - H \sum\limits_{j=1}^m (A_j x_j^k - A_j \tilde{x}_j^k), & i = 1, \\ \tilde{\lambda}^k - H \sum\limits_{j=1}^m (A_j x_j^k - A_j \tilde{x}_j^k) + \mu H \sum\limits_{j=1}^{i-1} (A_j x_j^k - A_j \tilde{x}_j^k), & i = 2, 3, \cdots, m. \end{cases}$$

Substituting (3.4) into VIP (3.3) and rearranging terms, we have

(3.5)
$$\begin{pmatrix} x_1' - \tilde{x}_1^k \\ x_2' - \tilde{x}_2^k \\ \cdots \\ x_m' - \tilde{x}_m^k \\ \lambda' - \tilde{\lambda}^k \end{pmatrix}^T \left\{ \begin{pmatrix} \tilde{\xi}_1^k - A_1^T \tilde{\lambda}^k \\ \tilde{\xi}_2^k - A_2^T \tilde{\lambda}^k \\ \cdots \\ \tilde{\xi}_m^k - A_m^T \tilde{\lambda}^k \\ \sum_{i=1}^m A_i \tilde{x}_i^k - b \end{pmatrix} + \begin{pmatrix} A_1^T \\ A_2^T \\ \vdots \\ A_m^T \\ 0 \end{pmatrix} H \sum_{j=1}^m (A_j x_j^k - A_j \tilde{x}_j^k) \right.$$
$$\left. - \begin{pmatrix} \mu A_1^T H A_1 & 0 & \cdots & 0 & 0 \\ \mu A_2^T H A_1 & \mu A_2^T H A_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mu A_m^T H A_1 & \mu A_m^T H A_2 & \cdots & \mu A_m^T H A_m & 0 \\ 0 & 0 & \cdots & 0 & H^{-1} \end{pmatrix} \begin{pmatrix} x_1^k - \tilde{x}_1^k \\ x_2^k - \tilde{x}_2^k \\ \cdots \\ x_m^k - \tilde{x}_m^k \\ \lambda^k - \tilde{\lambda}^k \end{pmatrix} \right\} \geq 0, \ \forall \mathbf{u}' \in \mathcal{U}.$$

Defining

$$\eta(\mathbf{u}^k, \tilde{\mathbf{u}}^k) := \begin{pmatrix} A_1^T \\ A_2^T \\ \vdots \\ A_m^T \\ 0 \end{pmatrix} H \sum_{j=1}^m (A_j x_j^k - A_j \tilde{x}_j^k)$$

and combining the definitions of $\mathbf{F}$ in (2.2) and M in (3.1), the inequality (3.5) can be condensed into

$$(\mathbf{u}' - \tilde{\mathbf{u}}^k)^T \left\{ \mathbf{F}(\tilde{\mathbf{u}}^k) + \eta(\mathbf{u}^k, \tilde{\mathbf{u}}^k) - M(\mathbf{u}^k - \tilde{\mathbf{u}}^k) \right\} \geq 0, \ \ \forall \mathbf{u}' \in \mathcal{U}. \tag{3.6}$$

If $A_i x_i^k = A_i \tilde{x}_i^k$ $(i = 1, 2, \cdots, m)$ and $\lambda^k = \tilde{\lambda}^k$, it yields that

$$\eta(\mathbf{u}^k, \tilde{\mathbf{u}}^k) = 0 \quad \text{and} \quad M(\mathbf{u}^k - \tilde{\mathbf{u}}^k) = 0.$$

Consequently, the inequality (3.6) further reduces to

$$(\mathbf{u}' - \tilde{\mathbf{u}}^k)^T \mathbf{F}(\tilde{\mathbf{u}}^k) \geq 0, \ \ \forall \mathbf{u}' \in \mathcal{U},$$

which indicates that $\tilde{\mathbf{u}}^k$ is a solution of VIP (2.3). □

4. Convergence

We now establish the global convergence of Algorithm 1 under some mild assumptions. We first introduce a matrix $Q \in \mathcal{R}^{n+l}$ defined as

$$Q = \begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 \\ A_1 & A_2 & \cdots & A_m & 0 \end{pmatrix}.$$

Obviously, the matrices M in (3.1), N in (3.2) and Q above satisfy the relationship

$$N = M + M^T + Q + Q^T. \tag{4.1}$$

Lemma 4.1. *Let $(x_1^*, x_2^*, \cdots, x_m^*)$ be an arbitrary solution of* (1.4) *and λ^* be the corresponding Lagrange multiplier. Then for any given $\mathbf{u}^k$, the point $\tilde{\mathbf{u}}^k$ generated by Algorithm* 1 *satisfies*

$$(\mathbf{u}^k - \mathbf{u}^*)^T M(\mathbf{u}^k - \tilde{\mathbf{u}}^k) \geq \frac{1}{2}\|\mathbf{u}^k - \tilde{\mathbf{u}}^k\|_N^2. \tag{4.2}$$

Proof. By adding (2.3) with $\mathbf{u} := \tilde{\mathbf{u}}^k$ to (3.6) with $\mathbf{u}' := \mathbf{u}^*$, we deduce that

$$(\tilde{\mathbf{u}}^k - \mathbf{u}^*)^T \left\{\mathbf{F}(\mathbf{u}^*) - \mathbf{F}(\tilde{\mathbf{u}}^k) - \eta(\mathbf{u}^k, \tilde{\mathbf{u}}^k) + M(\mathbf{u}^k - \tilde{\mathbf{u}}^k)\right\} \geq 0. \tag{4.3}$$

Because of the monotonicity of $\mathbf{F}(\mathbf{u})$ on $\mathcal{U}$, the inequality (4.3) can be written as

$$\begin{aligned}(\tilde{\mathbf{u}}^k - \mathbf{u}^*)^T M(\mathbf{u}^k - \tilde{\mathbf{u}}^k) &\geq (\tilde{\mathbf{u}}^k - \mathbf{u}^*)^T \eta(\mathbf{u}^k, \tilde{\mathbf{u}}^k) \\ &= (\lambda^k - \tilde{\lambda}^k)^T \sum_{j=1}^m (A_j x_j^k - A_j \tilde{x}_j^k) \\ &= (\mathbf{u}^k - \tilde{\mathbf{u}}^k)^T Q(\mathbf{u}^k - \tilde{\mathbf{u}}^k), \end{aligned} \tag{4.4}$$

where the first equality follows the fact that $\sum_{i=1}^m A_i x_i^* = b$. Furthermore, we can rewrite (4.4) as

$$\begin{aligned}(\mathbf{u}^k - \mathbf{u}^*)^T M(\mathbf{u}^k - \tilde{\mathbf{u}}^k) &\geq (\mathbf{u}^k - \tilde{\mathbf{u}}^k)^T (M + Q)(\mathbf{u}^k - \tilde{\mathbf{u}}^k) \\ &= \frac{1}{2}(\mathbf{u}^k - \tilde{\mathbf{u}}^k)^T (M + M^T + Q + Q^T)(\mathbf{u}^k - \tilde{\mathbf{u}}^k) \\ &= \frac{1}{2}(\mathbf{u}^k - \tilde{\mathbf{u}}^k)^T N(\mathbf{u}^k - \tilde{\mathbf{u}}^k) \\ &= \frac{1}{2}\|\mathbf{u}^k - \tilde{\mathbf{u}}^k\|_N^2, \end{aligned} \tag{4.5}$$

where the second equality is the consequence of (4.1). □

By the definition of α_k in Line 11 of Algorithm 1, it is obvious that $\alpha_k \geq 0$ for all k's. Moreover, we will prove in the following lemma that the sequence $\{\alpha_k\}$ is uniformly bounded away from zero; i.e., there exists a constant $\alpha_{\min} > 0$ such that $\alpha_k \geq \alpha_{\min}$ for all k's.

Lemma 4.2. *For the sequence $\{\alpha_k\}$ generated by Algorithm* 1, *there exists a constant $\alpha_{\min} > 0$ such that $\alpha_k \geq \alpha_{\min}$ for all $k \geq 0$.*

Proof. Since A_i's are assumed to be full column rank and $\mu \geq 1$, the matrix N in (3.2) is positive definite, i.e., the minimum eigenvalue $\text{eig}_{\min}(N) > 0$. By the definition of α_k in Line 11 of Algorithm 1, we deduce that

$$\alpha_k = \frac{\|\mathbf{u}^k - \tilde{\mathbf{u}}^k\|_N^2}{2\|G^{-1}M(\mathbf{u}^k - \tilde{\mathbf{u}}^k)\|_G^2} \geq \frac{\text{eig}_{\min}(N)}{2\|M^T G^{-1} M\|} =: \alpha_{\min}, \tag{4.6}$$

which completes the proof. □

Remark 4.1. By the equation (4.5), we have that

$$\begin{aligned}\tfrac{1}{2}\|\mathbf{u}^k - \tilde{\mathbf{u}}^k\|_N^2 &= (\mathbf{u}^k - \tilde{\mathbf{u}}^k)^T (M + Q)(\mathbf{u}^k - \tilde{\mathbf{u}}^k) \\ &= (\mathbf{u}^k - \tilde{\mathbf{u}}^k)^T M(\mathbf{u}^k - \tilde{\mathbf{u}}^k) + (\lambda^k - \tilde{\lambda}^k)^T \sum_{i=1}^m (A_i x_i^k - A_i \tilde{x}_i^k).\end{aligned}$$

It is numerically preferred that this equation be employed for computing α_k.

We now establish the global convergence of Algorithm 1 in the following theorem.

Theorem 4.3. *The sequence* $\{\mathbf{u}^k\}$ *generated by Algorithm* 1 *converges to a solution of VIP* (2.3)*.*

Proof. It follows from Line 12 of Algorithm 1 that

$$\begin{aligned}\|\mathbf{u}^{k+1}-\mathbf{u}^*\|_G^2 &= \|\mathbf{u}^k-\mathbf{u}^*\|_G^2 - 2\gamma\alpha_k(\mathbf{u}^k-\mathbf{u}^*)^T M(\mathbf{u}^k-\tilde{\mathbf{u}}^k)\\ &\quad + \gamma^2\alpha_k^2\|G^{-1}M(\mathbf{u}^k-\tilde{\mathbf{u}}^k)\|_G^2\\ &\le \|\mathbf{u}^k-\mathbf{u}^*\|_G^2 - \frac{1}{2}\gamma(2-\gamma)\alpha_k\|\mathbf{u}^k-\tilde{\mathbf{u}}^k\|_N^2\\ &\le \|\mathbf{u}^k-\mathbf{u}^*\|_G^2 - \frac{1}{2}\gamma(2-\gamma)\alpha_{\min}\|\mathbf{u}^k-\tilde{\mathbf{u}}^k\|_N^2, \end{aligned} \tag{4.7}$$

where the inequalities come from (4.2) and (4.6). The assumptions in Lemma 2.1 are fulfilled ($\beta_k \equiv 0$ and $\xi_k \equiv 0$), and we conclude that

(i) $\lim_{k\to\infty}\|\mathbf{u}^k-\mathbf{u}^*\|_G$ exists and is finite; (ii) $\lim_{k\to\infty}\|\mathbf{u}^k-\tilde{\mathbf{u}}^k\|_N = 0$.

Since the matrix N is positive definite as $\mu \ge 1$, we have

$$\lim_{k\to\infty}\|\lambda^k-\tilde{\lambda}^k\| = 0 \quad\text{and}\quad \lim_{k\to\infty}\|x_i^k-\tilde{x}_i^k\| = 0,\quad i=1,2,\cdots,m. \tag{4.8}$$

By Lemma 3.1, we deduce that the sequence $\{\tilde{\mathbf{u}}^k\}$ converges to a solution of VIP (2.3). Consequently, the sequence $\{(\tilde{x}_1^k,\tilde{x}_2^k,\cdots,\tilde{x}_m^k)\}$ converges to a solution of (1.4). □

5. An inexact version of the sequential updating scheme of the Lagrange multiplier

In this section, we consider an inexact version of the sequential updating scheme of the Lagrange multiplier which allows the minimization subproblems in Algorithm 1 to be solved approximately subject to certain inexactness criteria. This consideration is necessary when the objective functions $\theta_i(x_i)$'s in (1.4) are in general forms, and thus it is not possible to obtain closed-form solutions for the $\tilde{x}_i^k$-subproblems in Algorithm 1. In the following, we present an inexact version of Algorithm 1 and establish its convergence rigorously.

Remark 5.1. To check the inexact criterion in Line 8 of Algorithm 2 which involves the unknown exact solution $\tilde{x}_i^k$, we can employ an existing algorithm in the literature that is suitable for the x_i-subproblem and seek an approximate solution $\hat{x}_i^k$ such that a certain error bound which can sufficiently guarantee the criterion in Line 8 of Algorithm 2 is satisfied. For example, according to [11, Chapter 6], there exists a function $r_i : \mathcal{R}^{n_i} \to \mathcal{R}_+$ such that

$$\|x_i-\tilde{x}_i^k\| \le C_i\|r_i(x_i)\|,\ \forall x_i\in\mathcal{R}^{n_i}, \tag{5.1}$$

where $C_i > 0$ represents a constant independent of $\tilde{x}_i^k$ and $r_i(x_i)$ is computable based on the generated iterates. Then, as long as an iterate $\hat{x}_i^k$ generated by iteratively solving the x_i-subproblem satisfies

$$\|r_i(\hat{x}_i^k)\| \le \hat{\epsilon}_k/C_i,$$

it sufficiently ensures the inexact criterion in Line 8 of Algorithm 2. For how to choose $r_i(x_i)$ in (5.1) for different scenarios, we refer the reader to, e.g., [11,31] for details.

Algorithm 2: An inexact version of Algorithm 1

```
Input: Choose $\mu \geq 1$, $\gamma \in (0,2)$, $\epsilon > 0$, a sequence $\{\hat{\epsilon}_k\} \in [0,1)$ satisfying
       $\sum_{k=0}^{\infty} \hat{\epsilon}_k < +\infty$, $\mathbf{u}^0 = (x_1^0, x_2^0, \cdots, x_m^0, \lambda^0) \in \mathcal{U}$,
       $\hat{\mathbf{u}}^0 = (\hat{x}_1^0, \hat{x}_2^0, \cdots, \hat{x}_m^0, \hat{\lambda}^0) \in \mathcal{U}$, $H \in \mathcal{S}_{++}^l$ and $G \in \mathcal{S}_{++}^{n+l}$. Set $k = 0$.
1  while $\max\left\{\max_{1\le i\le m} \|A_i x_i^k - A_i \hat{x}_i^k\|, \|\lambda^k - \hat{\lambda}^k\|\right\} > \epsilon$ do
2      for $i = 1, 2, \cdots, m$ do
3          if $i = 1$ then
4              $\hat{\lambda}^{k+\frac{i-1}{m}} = \lambda^k - H\left(\sum_{j=1}^m A_j x_j^k - b\right)$.
5          else
6              $\hat{\lambda}^{k+\frac{i-1}{m}} = \hat{\lambda}^{k+\frac{i-2}{m}} + \mu H(A_{i-1}x_{i-1}^k - A_{i-1}\hat{x}_{i-1}^k)$.
7          end
8          Find an $\hat{x}_i^k$ satisfying $\|\hat{x}_i^k - \tilde{x}_i^k\| < \hat{\epsilon}_k$,
9          where $\tilde{x}_i^k = \arg\min_{x_i \in \mathcal{X}_i} \left\{\theta_i(x_i) - (\lambda^{k+\frac{i-1}{m}})^T A_i x_i + \frac{\mu}{2}\|A_i x_i - A_i x_i^k\|_H^2\right\}$.
10     end
11     $\hat{\lambda}^k = \lambda^k - H\left(\sum_{i=1}^m A_i \hat{x}_i^k - b\right)$.
12     $\hat{\alpha}_k = (1-\hat{\epsilon}_k)\|\mathbf{u}^k - \hat{\mathbf{u}}^k\|_N^2 / \left(2\|G^{-1}M(\mathbf{u}^k - \hat{\mathbf{u}}^k)\|_G^2\right)$.
13     $\mathbf{u}^{k+1} = \mathbf{u}^k - \gamma\hat{\alpha}_k G^{-1}M(\mathbf{u}^k - \hat{\mathbf{u}}^k)$.
14     $k = k + 1$.
15 end
```

Remark 5.2. The requirements $\hat{\epsilon}_k \geq 0$ and $\sum_{k=0}^{\infty} \hat{\epsilon}_k < +\infty$ imply that $\lim_{k\to\infty} \hat{\epsilon}_k = 0$. Thus, with the given accuracy $\epsilon > 0$ in the inexact criteria, there exists an integer $K > 0$ such that $\hat{\epsilon}_k < \epsilon$ for all $k \geq K$. Without loss of generality, we just assume that $\hat{\epsilon}_k < \epsilon$ for all k's.

The following lemma, together with Lemma 3.1, justifies the rationale of the stopping criteria in Algorithm 2.

Lemma 5.1. *If the stopping rule in Line 1 of Algorithm* 2 *holds, then there exists a constant* $\nu > 0$ *such that*

$$\max\left\{\max_{1\le i\le m} \|A_i x_i^k - A_i \tilde{x}_i^k\|, \|\lambda^k - \tilde{\lambda}^k\|\right\} < \nu\epsilon.$$

Proof. If the stopping rule in Line 1 of Algorithm 2 holds, i.e.,

$$\max\left\{\max_{1\le i\le m} \|A_i x_i^k - A_i \hat{x}_i^k\|, \|\lambda^k - \hat{\lambda}^k\|\right\} < \epsilon,$$

it follows that $\|A_i x_i^k - A_i \hat{x}_i^k\| < \epsilon$ for all i's. By the triangle inequality, we have

$$\begin{aligned}\|A_i x_i^k - A_i \tilde{x}_i^k\| &\leq \|A_i x_i^k - A_i \hat{x}_i^k\| + \|A_i \tilde{x}_i^k - A_i \hat{x}_i^k\| \\ &\leq \epsilon + \|A_i\|\hat{\epsilon}_k \\ &\leq (1 + \|A_i\|)\epsilon, \quad \forall\, i.\end{aligned}$$

Recall the updates of $\tilde{\lambda}^k$ and $\hat{\lambda}^k$ in Algorithms 1 and 2, respectively. We have

$$\hat{\lambda}^k - \tilde{\lambda}^k = H\sum_{i=1}^m (A_i \tilde{x}_i^k - A_i \hat{x}_i^k),$$

and as a consequence,

(5.2)
$$\|\hat{\lambda}^k - \tilde{\lambda}^k\| \leq \left\| H \sum_{i=1}^{m} (A_i \tilde{x}_i^k - A_i \hat{x}_i^k) \right\| \leq \|H\| \left(\sum_{i=1}^{m} \|A_i\| \cdot \|\tilde{x}_i^k - \hat{x}_i^k\| \right) \leq \varsigma_1 \hat{\epsilon}_k,$$

where $\varsigma_1 := \|H\| \left(\sum_{i=1}^m \|A_i\| \right)$. Hence,

$$\|\lambda^k - \tilde{\lambda}^k\| \leq \|\lambda^k - \hat{\lambda}^k\| + \|\hat{\lambda}^k - \tilde{\lambda}^k\| \leq (1 + \varsigma_1)\, \epsilon.$$

The proof is completed by setting $\nu := \max\{\max_{1\leq i \leq m}(1 + \|A_i\|), 1 + \varsigma_1\}$. □

Compared to the α_k in Algorithm 1, the $\hat{\alpha}_k$ in Algorithm 2 is computed by replacing (the unknown) $\tilde{\mathbf{u}}^k$ with $\hat{\mathbf{u}}^k$ and by multiplying by a factor $(1 - \hat{\epsilon}_k)$. Accordingly, the lower boundedness of $\hat{\alpha}_k$ can be proved analogously to Lemma 4.2. Specifically, it can be easily proved that

$$\hat{\alpha}_k \geq (1 - \hat{\epsilon}_{\max})\alpha_{\min} =: \hat{\alpha}_{\min} > 0, \quad \forall\, k \geq 0,$$

where $\hat{\epsilon}_{\max} := \max_{k\geq 0} \hat{\epsilon}_k$ and $\alpha_{\min}$ is as in (4.6). Moreover, followed by the fact that

$$\|G^{-1}M(\mathbf{u}^k - \hat{\mathbf{u}}^k)\|_G^2 \geq \mathrm{eig}_{\min}(M^T G^{-1} M)\|\mathbf{u}^k - \hat{\mathbf{u}}^k\|^2,$$
$$\|\mathbf{u}^k - \hat{\mathbf{u}}^k\|_N^2 \leq \mathrm{eig}_{\max}(N)\|\mathbf{u}^k - \hat{\mathbf{u}}^k\|^2,$$

we have

(5.3)
$$\begin{aligned}\hat{\alpha}_k = \frac{(1-\hat{\epsilon}_k)\|\mathbf{u}^k - \hat{\mathbf{u}}^k\|_N^2}{2\|G^{-1}M(\mathbf{u}^k - \hat{\mathbf{u}}^k)\|_G^2} &\leq \frac{(1-\hat{\epsilon}_k)\mathrm{eig}_{\max}(N)}{2 \cdot \mathrm{eig}_{\min}(M^T G^{-1} M)} \\ &< \frac{\mathrm{eig}_{\max}(N)}{2 \cdot \mathrm{eig}_{\min}(M^T G^{-1} M)} =: \hat{\alpha}_{\max},\end{aligned}$$

which indicates that the sequence $\{\hat{\alpha}_k\} \subset [\hat{\alpha}_{\min}, \hat{\alpha}_{\max}]$ is bounded.

Lemma 5.2. *Let $(x_1^*, x_2^*, \cdots, x_m^*)$ be an arbitrary solution of (1.4) and λ^* be the corresponding Lagrange multiplier. Then the sequences $\{\mathbf{u}^k\}$ and $\{\hat{\mathbf{u}}^k\}$ generated by Algorithm 2 satisfy*

(5.4)
$$\begin{aligned}(\mathbf{u}^k - \mathbf{u}^*)M(\mathbf{u}^k - \hat{\mathbf{u}}^k) \geq & \frac{1}{2}\left\{(1-\hat{\epsilon}_k)\|\mathbf{u}^k - \hat{\mathbf{u}}^k\|_N^2 - (1 + \|N\|)(m + \varsigma_1^2)\hat{\epsilon}_k \right. \\ & \left. - \hat{\epsilon}_k \|M\|^2 \cdot \|\mathbf{u}^k - \mathbf{u}^*\|^2 \right\}.\end{aligned}$$

Proof. First, notice the identity

$$(\mathbf{u}^k - \mathbf{u}^*)^T M(\mathbf{u}^k - \hat{\mathbf{u}}^k) = (\mathbf{u}^k - \mathbf{u}^*)^T M(\mathbf{u}^k - \tilde{\mathbf{u}}^k) + (\mathbf{u}^k - \mathbf{u}^*)^T M(\tilde{\mathbf{u}}^k - \hat{\mathbf{u}}^k).$$

Applying the assertion in Lemma 4.1 to the first term of the right-hand side above, we obtain

(5.5)
$$(\mathbf{u}^k - \mathbf{u}^*)^T M(\mathbf{u}^k - \hat{\mathbf{u}}^k) \geq \frac{1}{2}\|\mathbf{u}^k - \tilde{\mathbf{u}}^k\|_N^2 + (\mathbf{u}^k - \mathbf{u}^*)^T M(\tilde{\mathbf{u}}^k - \hat{\mathbf{u}}^k).$$

Using (5.2), we get

(5.6)
$$\|\hat{\mathbf{u}}^k - \tilde{\mathbf{u}}^k\|^2 = \sum_{i=1}^{m} \|\hat{x}_i^k - \tilde{x}_i^k\|^2 + \|\hat{\lambda}^k - \tilde{\lambda}^k\|^2 \leq (m + \varsigma_1^2)\hat{\epsilon}_k^2.$$

For any two vectors a and b with the same dimension, the inequality

(5.7)
$$2a^T b \leq \varrho\|a\|^2 + \frac{1}{\varrho}\|b\|^2, \quad \forall \varrho > 0,$$

holds. Then, by setting $a := -M^T(\mathbf{u}^k - \mathbf{u}^*)$, $b := \tilde{\mathbf{u}}^k - \hat{\mathbf{u}}^k$ and $\varrho := \hat{\epsilon}_k$ in (5.7), we obtain

$$\begin{aligned} 2(\mathbf{u}^k - \mathbf{u}^*)^T M(\tilde{\mathbf{u}}^k - \hat{\mathbf{u}}^k) &\geq -\hat{\epsilon}_k \|M^T(\mathbf{u}^k - \mathbf{u}^*)\|^2 - \frac{1}{\hat{\epsilon}_k}\|\tilde{\mathbf{u}}^k - \hat{\mathbf{u}}^k\|^2 \\ &\geq -\hat{\epsilon}_k \|M\|^2 \cdot \|\mathbf{u}^k - \mathbf{u}^*\|^2 - \frac{1}{\hat{\epsilon}_k}\|\tilde{\mathbf{u}}^k - \hat{\mathbf{u}}^k\|^2 \\ &\geq -\hat{\epsilon}_k \|M\|^2 \cdot \|\mathbf{u}^k - \mathbf{u}^*\|^2 - (m + \varsigma_1^2)\hat{\epsilon}_k, \end{aligned} \tag{5.8}$$

where the last inequality follows from (5.6). Analogically, by setting $a := -N^{\frac{1}{2}}(\mathbf{u}^k - \hat{\mathbf{u}}^k)$, $b := N^{\frac{1}{2}}(\hat{\mathbf{u}}^k - \tilde{\mathbf{u}}^k)$ and $\varrho := \hat{\epsilon}_k$ in (5.7), we obtain

$$\begin{aligned} 2(\mathbf{u}^k - \hat{\mathbf{u}}^k)^T N(\hat{\mathbf{u}}^k - \tilde{\mathbf{u}}^k) &\geq -\hat{\epsilon}_k \|\mathbf{u}^k - \hat{\mathbf{u}}^k\|_N^2 - \frac{1}{\hat{\epsilon}_k}\|N\| \cdot \|\tilde{\mathbf{u}}^k - \hat{\mathbf{u}}^k\|^2 \\ &\geq -\hat{\epsilon}_k \|\mathbf{u}^k - \hat{\mathbf{u}}^k\|_N^2 - \|N\|(m + \varsigma_1^2)\hat{\epsilon}_k. \end{aligned}$$

Consequently,

$$\begin{aligned} \|\mathbf{u}^k - \tilde{\mathbf{u}}^k\|_N^2 &= \|\mathbf{u}^k - \hat{\mathbf{u}}^k\|_N^2 + 2(\mathbf{u}^k - \hat{\mathbf{u}}^k)^T N(\hat{\mathbf{u}}^k - \tilde{\mathbf{u}}^k) + \|\hat{\mathbf{u}}^k - \tilde{\mathbf{u}}^k\|_N^2 \\ &\geq (1 - \hat{\epsilon}_k)\|\mathbf{u}^k - \hat{\mathbf{u}}^k\|_N^2 - \|N\|(m + \varsigma_1^2)\hat{\epsilon}_k. \end{aligned} \tag{5.9}$$

Combining (5.5), (5.8) and (5.9), we obtain the assertion immediately. ☐

By Lemma 5.2, however, $-G^{-1}M(\mathbf{u}^k - \hat{\mathbf{u}}^k)$ cannot be guaranteed as a descent direction of the implicit merit function $\frac{1}{2}\|\mathbf{u}^k - \mathbf{u}^*\|_G^2$. Nevertheless, we are still capable of proving the global convergence of Algorithm 2.

We are now in the position of proving the global convergence of the proposed algorithm.

Theorem 5.3. *The sequence* $\{\mathbf{u}^k\}$ *generated by Algorithm* 2 *converges to a solution of VIP* (2.3)*.*

Proof. It follows from the scheme of $\mathbf{u}^{k+1}$ in Line 13 of Algorithm 2 that

$$\begin{aligned} \|\mathbf{u}^{k+1} - \mathbf{u}^*\|_G^2 &= \|\mathbf{u}^k - \mathbf{u}^*\|_G^2 + \gamma^2\hat{\alpha}_k^2 \|G^{-1}M(\mathbf{u}^k - \hat{\mathbf{u}}^k)\|_G^2 \\ &\quad - 2\gamma\hat{\alpha}_k(\mathbf{u}^k - \mathbf{u}^*)^T M(\mathbf{u}^k - \hat{\mathbf{u}}^k) \\ &\leq \|\mathbf{u}^k - \mathbf{u}^*\|_G^2 + \gamma^2\hat{\alpha}_k^2 \|G^{-1}M(\mathbf{u}^k - \hat{\mathbf{u}}^k)\|_G^2 - \gamma\hat{\alpha}_k \big[(1 - \hat{\epsilon}_k)\|\mathbf{u}^k - \hat{\mathbf{u}}^k\|_N^2 \\ &\qquad -(1 + \|N\|)(m + \varsigma_1^2)\hat{\epsilon}_k - \|M\|^2\hat{\epsilon}_k\|\mathbf{u}^k - \mathbf{u}^*\|\big] \\ &\leq \left(1 + \frac{\gamma\hat{\alpha}_{\max}\hat{\epsilon}_k\|M\|^2}{\mathrm{eig}_{\min}(G)}\right)\|\mathbf{u}^k - \mathbf{u}^*\|_G^2 - \frac{1}{2}\gamma(2-\gamma)\hat{\alpha}_{\min}\|\mathbf{u}^k - \hat{\mathbf{u}}^k\|_N^2 \\ &\qquad + \gamma\hat{\alpha}_{\max}(1 + \|N\|)(m + \varsigma_1^2)\hat{\epsilon}_k, \end{aligned} \tag{5.10}$$

where the inequalities are due to (5.4) and (5.3), respectively. Specializing $z_k := \|\mathbf{u}^k - \mathbf{u}^*\|_G^2$, $\beta_k := \frac{\gamma\hat{\alpha}_{\max}\hat{\epsilon}_k\|M\|^2}{\mathrm{eig}_{\min}(G)}$, $\xi_k := \gamma\hat{\alpha}_{\max}(1 + \|N\|)(m + \varsigma_1^2)\hat{\epsilon}_k$, and $\zeta_k := \frac{1}{2}\gamma(2-\gamma)\hat{\alpha}_{\min}\|\mathbf{u}^k - \hat{\mathbf{u}}^k\|_N^2$, respectively, we know that the assumptions in Lemma 2.1 are fulfilled and we conclude that

(i) $\lim_{k\to\infty} \|\mathbf{u}^k - \mathbf{u}^*\|_G$ exists and is finite; (ii) $\lim_{k\to\infty} \|\mathbf{u}^k - \hat{\mathbf{u}}^k\|_N = 0.$

Combining Line 8 of Algorithm 2 and (5.2), we have

$$\lim_{k\to\infty} \|\hat{\mathbf{u}}^k - \tilde{\mathbf{u}}^k\| = 0,$$

which indicates that the sequences $\{\mathbf{u}^k\}$, $\{\hat{\mathbf{u}}^k\}$ and $\{\tilde{\mathbf{u}}^k\}$ are all bounded and they share the same cluster points. Let $\check{\mathbf{u}}$ be a cluster point such that

$$\lim_{j\to\infty} \mathbf{u}^{k_j} = \lim_{j\to\infty} \hat{\mathbf{u}}^{k_j} = \lim_{j\to\infty} \tilde{\mathbf{u}}^{k_j} = \check{\mathbf{u}}. \tag{5.11}$$

It follows from the optimality condition that for $\tilde{x}_i^k$, there exists some $\tilde{\xi}_i^k \in \partial\theta_i(\tilde{x}_i^k)$ such that the following inequality holds:

$$(x_i' - \tilde{x}_i^k)^T \left\{ \tilde{\xi}_i^k - A_i^T \lambda^{k+\frac{i-1}{m}} + \mu A_i^T H A_i (\tilde{x}_i^k - x_i^k) \right\} \geq 0, \quad \forall x_i' \in \mathcal{X}_i. \tag{5.12}$$

Note that according to Line 11 of Algorithm 2, we have

$$(\lambda' - \hat{\lambda}^k)^T \left\{ \sum_{i=1}^m A_i \hat{x}_i^k - b + H^{-1}(\hat{\lambda}^k - \lambda^k) \right\} \geq 0, \quad \forall \lambda' \in \mathcal{R}^l. \tag{5.13}$$

Taking the limit in (5.12)-(5.13) and using the upper semicontinuity of the subdifferential operator and (5.11), it follows that for all $i = 1, \cdots, m$, there exist $\check{\xi}_i$ such that

$$(x_i' - \check{x}_i)^T \left\{ \check{\xi}_i - A_i^T \check{\lambda} \right\} \geq 0, \quad \forall x_i' \in \mathcal{X}_i,$$

and

$$(\lambda' - \check{\lambda})^T \left\{ \sum_{i=1}^m A_i \check{x}_i - b \right\} \geq 0, \quad \forall \lambda' \in \mathcal{R}^l,$$

which means that $\check{\mathbf{u}}$ is a solution point of VIP (2.3).

Setting $\mathbf{u}^* := \check{\mathbf{u}}$ in (5.10) and applying again Lemma 2.1, we deduce that $\lim_{k\to\infty} \|\mathbf{u}^k - \check{\mathbf{u}}\|_G$ exists and it is finite. Since $\check{\mathbf{u}}$ is the cluster point of $\{\mathbf{u}^k\}$, we eventually conclude that the sequence $\{\mathbf{u}^k\}$ converges to $\check{\mathbf{u}}$, which is a solution point of VIP (2.3). □

6. Numerical experiments

In this section, we test some applications of the abstract model (1.4) and empirically verify the efficiency of the proposed sequential updating scheme of the Lagrange multiplier. For these applications, all the decomposed subproblems have closed-form solutions. Thus, only Algorithm 1 ("SUSLM" for short), the exact version, is tested. We wrote our code by Matlab 7.9, and all experiments were conducted on a Lenovo personal computer with Intel Core (TM) CPU 2.30 GHZ and 8G memory.

6.1. **An image decomposition model.**

First, we test an image decomposition model. A brief introduction to the image decomposition problem is as follows. For a digital image $f \in \mathcal{R}^n$ (the two- or higher-dimensional image is stacked as a one-dimensional vector, e.g., in lexicographic order, and the pixel values of image are re-scaled into [0,1]), the goal of image decomposition is to separate $f = u + v$ where u is the cartoon part which contains the geometry or sketchy approximation of f, and v is the texture part which involves the oscillations or small scale repeated patterns of f (see, e.g., [1, 34, 36, 44, 46]). Mathematically, the cartoon part of an image can be described by a piecewise smooth (or piecewise constant) function, and the texture part is commonly oscillating. We test the scenario considered in [43], where a low patch-rank interpretation is used for the texture part. In [43], an n_1-by-n_2 ($n = n_1 \cdot n_2$) image v with textures is first partitioned orderly into a series of r-by-r ($r \ll n$) small square patches (nonoverlapping) under a certain boundary condition. Then these small patches are vectorized individually as a vector, denoted by $w_i \in \mathcal{R}^{r^2}$ with $i = 1, 2, \cdots, l$. Therein, $l = \lceil n/r^2 \rceil$, where $\lceil \cdot \rceil$ rounds a scalar as the nearest integer towards infinity. By further realigning those vector w_i's together, an r^2-by-l matrix, denoted by V, is obtained. The patch mapping $\mathcal{P} : \mathcal{R}^n \to \mathcal{R}^{r^2 \times l}$, which depicts the foregoing procedure of rearranging an image v as a matrix V, is defined as (see [43] for details)

$$V := \mathcal{P}v = [w_1, w_2, \cdots, w_l].$$

Generally, the texture part of a target image possesses numerous small scale repeated patterns. Thus, the matrix V, also the $\mathcal{P}v$, can be viewed as being of low rank. Accordingly, a low-rank based model for separating a target image f into cartoon and texture was developed in [43] as

$$\min_{u \in \mathcal{R}^n, v \in \mathcal{R}^n} \tau_1 \big\| |\nabla u| \big\|_1 + \tau_2 \|\mathcal{P}v\|_* + \frac{\tau_3}{2} \|K(u+v) - f\|^2, \tag{6.1}$$

where $\big\| |\nabla \cdot| \big\|_1$ denotes the total variation norm (see, e.g., [42]) and induces the cartoon part of f; $\| \cdot \|_*$ is the nuclear norm which is defined as the sum of all singular values of a matrix and induces the texture part of f; τ_i $(i = 1, 2, 3)$ are parameters to balance the three terms in the objective function; and $K : \mathcal{R}^n \to \mathcal{R}^n$ is a linear operator. Different K's correspond to image decomposition on different target images: (i) for clean images, $K = I$ where I is the identity matrix; (ii) for images with missing pixels (herein we consider the case of missing pixels with zero values), $K = S$ where S is a binary matrix (also the so-called "mask" in image inpainting) to represent missing pixels; (iii) for images with blurry pixels, $K = B$ where B is the convolutional matrix associated with a spatially invariant point spread function; (iv) for images with both blurry and missing pixels, $K = SB$ where S is a mask and B is a convolutional matrix.

The model (6.1) can be reformulated as a special case of the abstract model (1.4). Specifically, by introducing the auxiliary variables $x \in \mathcal{R}^n \times \mathcal{R}^n$, $y \in \mathcal{R}^{r^2 \times l}$ and $z \in \mathcal{R}^n$, the model (6.1) can be rewritten as

$$\min \left\{ \tau_1 \big\| |x| \big\|_1 + \tau_2 \|y\|_* + \frac{\tau_3}{2} \|Kz - f\|^2 \mid x = \nabla u,\ y = \mathcal{P}v,\ z = u + v \right\}. \tag{6.2}$$

Thus, the problem (6.2) is a special case of (1.4) with $m = 3$ and the following specifications:

- $x_1 := u$, $x_2 := v$, $x_3 := (x, y, z)$, $\mathcal{X}_i$ $(i = 1, 2, 3)$ are Euclidean spaces.
- $\theta_1(x_1) := 0$, $\theta_2(x_2) := 0$ and $\theta_3(x_3) := \tau_1 \big\| |x| \big\|_1 + \tau_2 \|y\|_* + \frac{\tau_3}{2} \|Kz - f\|^2$.
- The matrices in the linear constraint are

$$A_1 := \begin{pmatrix} \nabla \\ 0 \\ I \end{pmatrix}, \quad A_2 := \begin{pmatrix} 0 \\ \mathcal{P} \\ I \end{pmatrix}, \quad A_3 := \begin{pmatrix} -I & 0 & 0 \\ 0 & -I & 0 \\ 0 & 0 & -I \end{pmatrix},$$

  and the vector b is zero.

The proposed SUSLM is thus applicable to the model (6.1). The decomposed $\tilde{x}_1$-, $\tilde{x}_2$- and $\tilde{x}_3$-subproblems (i.e., the $\tilde{u}$-, $\tilde{v}$- and $(\tilde{x}, \tilde{y}, \tilde{z})$-subproblems) by implementing the SUSLM are further delineated as follows.

- The $\tilde{u}$-subproblem corresponds to the optimization problem

$$\tilde{u}^k = u^k + [\mu\beta(\nabla^T \nabla + I)]^{-1} (\nabla^T \lambda_1^{k + \frac{0}{3}} + \lambda_3^{k + \frac{0}{3}}), \tag{6.3}$$

which can be easily solved by the fast Fourier transform (FFT) if the periodic boundary condition is exploited to the derivative operator ∇, or by the discrete cosine transform (DCT) if the reflective boundary condition is exploited (see, e.g., [21, Chapter 7]).

- The $\tilde{v}$-subproblem is equivalent to the optimization problem

$$\tilde{v}^k = v^k + (\mathcal{P}^{-1}\lambda_2^{k+\frac{1}{3}} + \lambda_3^{k+\frac{1}{3}})/(2\mu\beta).$$

- The $\tilde{x}$-, $\tilde{y}$- and $\tilde{z}$-subproblems can be solved simultaneously as follows.
  - The $\tilde{x}$-subproblem can be solved explicitly by the shrinkage operator

$$\tilde{x}^k = \text{shrink}_{\frac{\tau_1}{\mu\beta}}(x^k - \lambda_1^{k+\frac{2}{3}}/(\mu\beta)),$$

where, for any $c > 0$, the mapping $\text{shrink}_c(\cdot)$ is defined as

$$\text{shrink}_c(g) := g - \min\{c, |g|\}\frac{g}{|g|}, \quad \forall g \in \mathcal{R}^n \times \mathcal{R}^n,$$

and $(\frac{g}{|g|})_i$ should be taken as 0 if $|g|_i = 0$.

  - The $\tilde{y}$-subproblem can be solved explicitly by the singular value decomposition (SVD)

$$\tilde{y}^k = \mathcal{D}_{\frac{\tau_2}{\mu\beta}}(y^k - \lambda_2^{k+\frac{2}{3}}/(\mu\beta)).$$

Here, for any $c > 0$, the mapping $\mathcal{D}_c(\cdot)$ is defined as

$$\mathcal{D}_c(M) := U\hat{\Sigma}V^T, \quad \forall\, M \in \mathcal{R}^{m\times n}, \tag{6.4}$$

where $U\Sigma V^T$ is the SVD of M, and $\hat{\Sigma}_{ij} = \max\{\Sigma_{ij} - c, 0\}$ for all $1 \le i \le m$ and $1 \le j \le n$.

  - The $\tilde{z}$-subproblem is involved in the matrix K:

$$\tilde{z}^k = (\tau_3 K^T K + \mu\beta)^{-1}(\tau_3 K^T f - \lambda_3^{k+\frac{2}{3}} + \mu\beta z^k). \tag{6.5}$$

If K is an identity, diagonal or downsampling matrix, then the $\tilde{z}$-subproblem (6.5) can be solved directly since its coefficient matrix is diagonal. If K is a convolutional matrix, then the $\tilde{z}$-subproblem (6.5) can be solved efficiently in the frequency domain as the $\tilde{u}$-subproblem (6.3).

For simplicity, we choose the matrices (see the `Input` in Algorithm 1) $G = I_{n+l}$ and $H = \beta I_l$ with $\beta > 0$ a scalar.

In [43], it is suggested that the model (6.1) be solved by the split Bregman method (SBM for short) proposed in [19]. To see the efficiency of SUSLM, we compare it with the SBM in [43] and report the numerical results. Comparisons with the alternating direction method with Gaussian back substitution (ADM-G) in [25] and the alternating proximal gradient method (APGM) in [32] are also reported. For the other methods to be compared, we also wrote the code by MATLAB 7.9, and their parameters were chosen exactly as suggested in the original papers. We refer the reader to [43] for the implementation details of the SBM. For the details of the ADM-G subproblems, since they are similar to those of the SUSLM, we omit them for succinctness. Now, we elaborate on the details of the APGM subproblems.

First, we need to reformulate the problem (6.2) as a special case of (1.4) with $m = 2$ and the following specifications:

- $x_1 := (u, v)$, $x_2 := (x, y, z)$, $\mathcal{X}_i$ $(i = 1, 2)$ are Euclidean spaces.
- $\theta_1(x_1) := 0$ and $\theta_2(x_2) := \tau_1 \big\| |x| \big\|_1 + \tau_2 \|y\|_* + \frac{\tau_3}{2} \|Kz - f\|^2$.
- The matrices in the linear constraint are

$$A_1 := \begin{pmatrix} \nabla & 0 \\ 0 & \mathcal{P} \\ I & I \end{pmatrix}, \quad A_2 := \begin{pmatrix} -I & 0 & 0 \\ 0 & -I & 0 \\ 0 & 0 & -I \end{pmatrix}$$

and $b = 0$. Then, the APGM iterative scheme for (6.2) is

$$\begin{cases} x_1^{k+1} = \arg\min \Big\{ \theta_1(x_1) + \frac{\beta}{2r_1} \|x_1 - x_1^k + r_1 A_1^T (A_1 x_1^k + A_2 x_2^k - b - \lambda^k/\beta)\|^2 \Big\}, \\ x_2^{k+1} = \arg\min \Big\{ \theta_2(x_2) + \frac{\beta}{2r_2} \|x_2 - x_2^k + r_2 A_2^T (A_1 x_1^{k+1} + A_2 x_2^k - b - \lambda^k/\beta)\|^2 \Big\}, \\ \lambda^{k+1} = \lambda^k - \beta(A_1 x_1^{k+1} + A_2 x_2^{k+1} - b), \end{cases}$$

where $r_i < 1/\mathrm{eig}_{\max}(A_i^T A_i)$ is the step size of the proximal gradient step and $\beta > 0$ is the penalty parameter. The x_i-subproblems can be further specified as

$$\begin{cases} u^{k+1} = u^k - r_1 \left[\nabla^T (\nabla u^k - x^k - \lambda_1^k/\beta) + (u^k + v^k - z^k - \lambda_3^k/\beta) \right], \\ v^{k+1} = v^k - r_1 \left[\mathcal{P}^{-1} (\mathcal{P} v^k - y^k - \lambda_2^k/\beta) + (u^k + v^k - z^k - \lambda_3^k/\beta) \right], \\ x^{k+1} = \mathrm{shrink}_{\frac{\tau_1 r_2}{\beta}} \left(x^k + r_2 (\nabla u^{k+1} - x^k - \lambda_1^k/\beta) \right), \\ y^{k+1} = \mathcal{D}_{\frac{\tau_2 r_2}{\beta}} \left(y^k + r_2 (\mathcal{P} v^{k+1} - y^k - \lambda_2^k/\beta) \|^2 \right), \\ z^{k+1} = (\beta I + \tau_2 r_3 K^T K)^{-1} \left[\tau_3 r_2 K^T f + \beta (z^k + r_2 (u^{k+1} + v^{k+1} - z^k - \lambda_3^k/\beta)) \right]. \end{cases}$$

We will test two scenarios for the model (6.1): $K = I$ and $K = S$. Some synthetic images and real images listed in Figures 1-2 will be tested. For the synthetic images in Figure 1, cartoon and texture parts are superposed with a ratio of 7:3. We adopt the periodic boundary condition for the images to be tested and thus the FFT will be implemented. Also, we will implement an efficient MATLAB Mex interface via a divide-and-conquer routine (dgesdd) implemented in LAPACK for the SVD in (6.4).

We follow [43, Theorem 3.3-3.6] to choose the trade-offs in model (6.1) as $\tau_1 \in [10^{-2}, 10^{-1}]$, $\tau_2 \in [10^{-3}, 10^{-2}]$ and $\tau_3 \equiv 1$. Recall that r is the size of the patch mapping $\mathcal{P}$. It can be easily estimated by the number of spikes of the target image f in the Fourier domain (see [43] for details). Empirically, as in [43], the integers in the interval [5,15] are preferable for the scalar r to render promising numerical results. We adopt $r = 11$ as the size of the patch mapping $\mathcal{P}$ throughout the numerical simulations.

To implement the SUSLM, some parameters should be specified. As we have mentioned, we simply take the penalty matrix $H = \beta I$. Numerically, the value of β is often tuned in order to result in better numerical results. We used the Boy image in Figure 1 to tune the value of β and we found that the choice of $\beta = 0.1$ works well for SUSLM. We thus fix $\beta = 0.1$ throughout the implementation of SUSLM. Then, for the proximity parameter μ, we found that $\mu \approx 1$ is good enough (see Figure 3) and we thus fix it as $\mu \equiv 1$ throughout. Finally, for the relaxation parameter γ, we take it as $\gamma \equiv 1.9$.

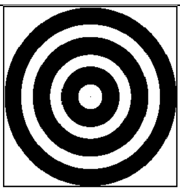
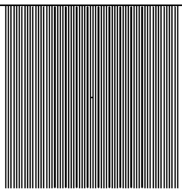
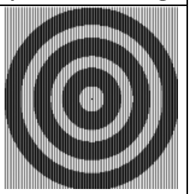
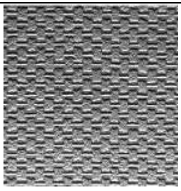
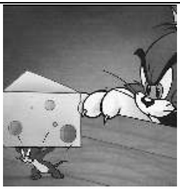
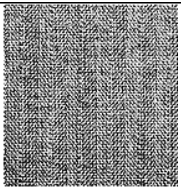
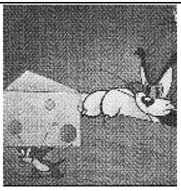
Image	cartoon	texture	synthetic image
"Circles" 256×256			
"Boy" 256×256			
"Tomjerry" 512×512			

FIGURE 1. Synthetic images.

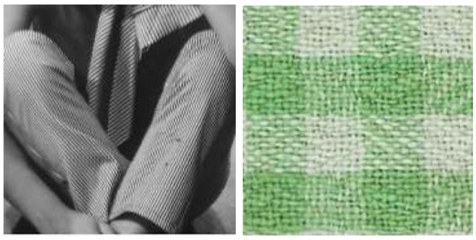


FIGURE 2. Real images. Left: 256×256 "Barbara.png". Right: 250×248 "Weave.jpg".

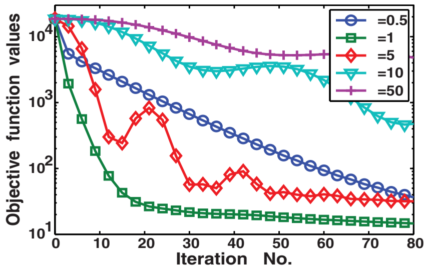


FIGURE 3. Performance of SUSLM with different values of the proximal factor μ.

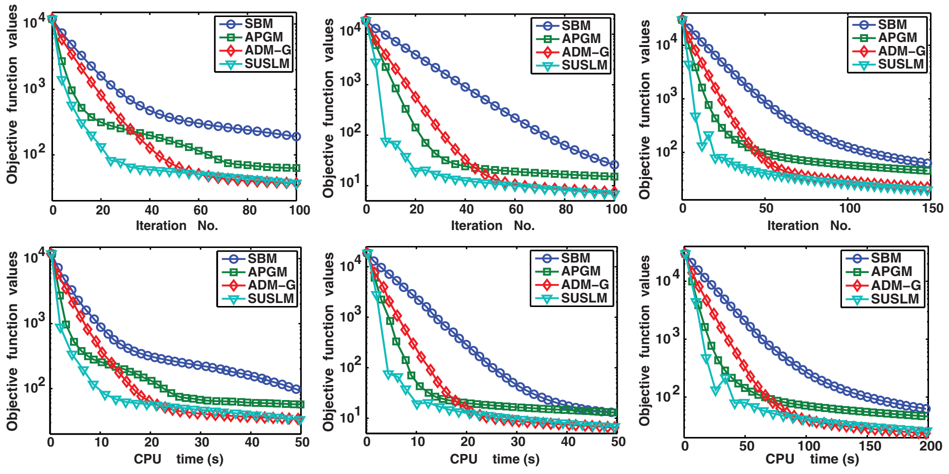


FIGURE 4. Evolutions of objective function values w.r.t. iterations and computing time. From left column to right column: for Circles, Boy and Tomjerry.

6.1.1. *The case of $K = I$.* For this case, the model (6.1) reduces to decomposing clean images without noise or blur.

We first test the synthetic images in Figure 1. The parameters (τ_1, τ_2, τ_3) in model (6.1) are fixed as $(\tau_1, \tau_2, \tau_3) = (0.05, 0.005, 1)$, as suggested by Theorems 3.3-3.6 in [43]. We take the penalty parameters $(\lambda_1, \lambda_2) = (1, 1)$ (as suggested in [43]) for the SBM; $\beta = 10$ and $\alpha = 1$ for the ADM-G; $\beta = 10$, $r_1 = 0.2$ and $r_2 = 1$ for the APGM. All the tested algorithms take zero as the initial point. We run all algorithms for some iterations (100 iterations for 256-by-256 images and 150 iterations for 512-by-512 images) and plot the evolutions of objective function values with respect to iterations and computing time in seconds in Figure 4. For synthetic images, the ground-truth of cartoons and textures are known (see Figure 1). We thus use the following criteria to measure the quality of the cartoons and textures decomposed by the tested algorithms:

$$\text{Error}(u) := \|u - u^*\| \quad \text{and} \quad \text{Error}(v) := \|v - v^*\|, \tag{6.6}$$

where u and v (pixel values are re-scaled in $[0, 1]$) are the decomposed cartoons and textures, respectively, and u^* and v^* are the corresponding ground-truths of synthetic images in Figure 1. The evolutions of Error(u) and Error(v) with respect to iterations and computing time in seconds are plotted in Figure 5. In Figure 6, we display the cartoons and textures decomposed by the SBM and SUSLM (corresponding to the best and worst decompositions among all tested algorithms).

We now test the two real images in Figure 2. Analogously, the parameters (τ_1, τ_2, τ_3) in model (6.1) are still fixed as $(\tau_1, \tau_2, \tau_3) = (0.05, 0.005, 1)$. The parameters (λ_1, λ_2) for the SBM; β and α for the ADM-G; β, r_1 and r_2 for the APGM and β for the SUSLM are identical to those of the synthetic image case. The initial points for all algorithms are taken as zero vectors. We run each algorithm for 150

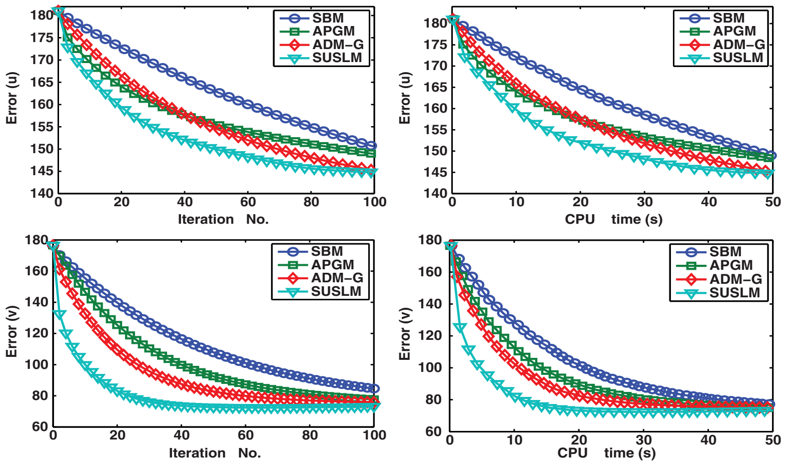


FIGURE 5. Evolutions of Error(u) and Error(v) w.r.t. iterations and computing time (test image: Circles).

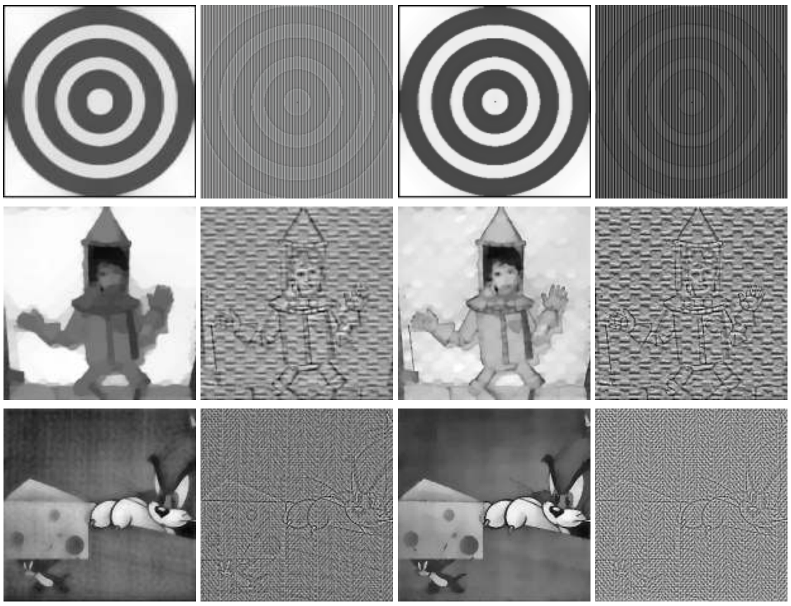


FIGURE 6. Decomposed cartoons and textures of synthetic images. From left column to right column: cartoons by SBM, textures by SBM, cartoons by SUSLM, textures by SUSLM.

iterations for both images and plot the evolutions of objective function values with respect to iterations and computing time in seconds in Figure 7. We display the decomposed cartoons and textures in Figure 8.

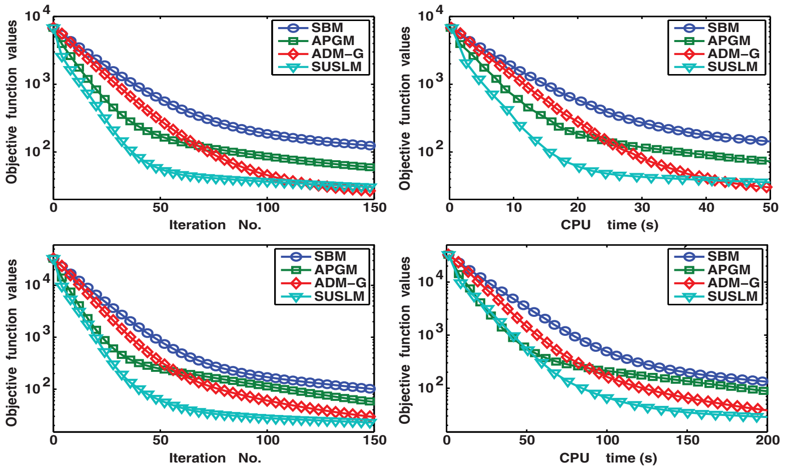


FIGURE 7. Evolutions of objective function values w.r.t. iterations and computing time. Top row: for Barbara. Bottom row: for Weave.

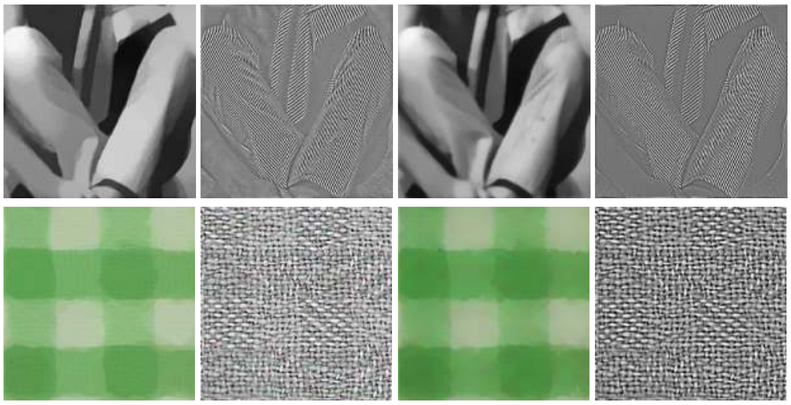


FIGURE 8. Decomposed cartoons and textures on real images. From left column to right column: cartoons by SBM, textures by SBM, cartoons by SUSLM, textures by SUSLM.

6.1.2. *The case of* $K = S$. We now test the model (6.1) with $K = S$; i.e., the image f to be decomposed has missing pixels. We test the 256-by-256 Barbara image in Figure 2 (with 9.12% missing pixels) and the 512-by-512 Tomjerry image in Figure 1 (with 12.76% missing pixels). Both corrupted images are listed in Figure 9. The signal-to-noise ratio (SNR), which is commonly used to measure the quality of reconstructed images, is defined by $\mathrm{SNR} = 20\log_{10}\|f^*\|/\|f - f^*\|$, where f^* is the true image and f is its approximation. For the corrupted images listed in Figure 9, the SNR values are 10.88dB for the corrupted Barbara image and 9.06dB for the corrupted Tomjerry image.

The parameters (τ_1, τ_2, τ_3) in (6.2) are chosen as $(0.08, 0.005, 1)$, as suggested by Theorems 3.3-3.6 in [43]. The penalty parameters are fixed as $(\lambda_1, \lambda_2) = (1, 1)$ for

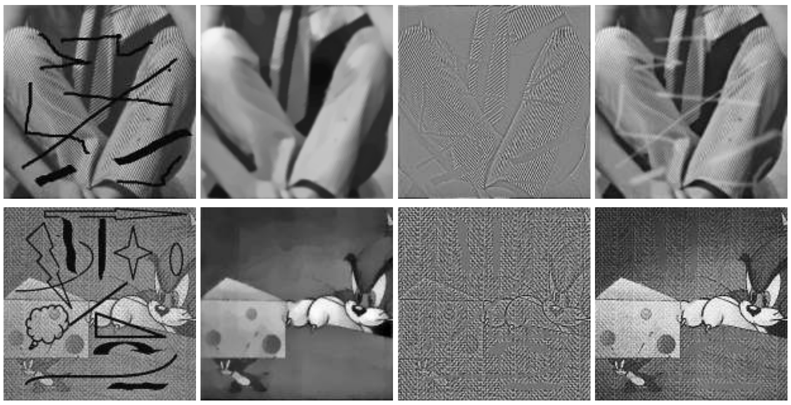


FIGURE 9. Decomposed cartoons and textures on images with missing pixels by SUSLM. From left column to right column: corrupted image, cartoon, texture, cartoon+texture.

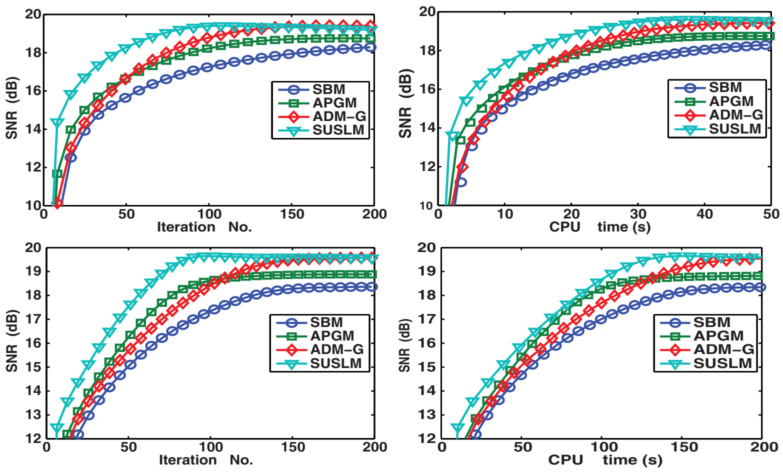


FIGURE 10. Evolutions of SNR w.r.t. iterations and computing time. Top row: for Barbara. Bottom row: for Tomjerry.

the SBM; $\beta = 10$ and $\alpha = 1$ for the ADM-G; $\beta = 10$, $r_1 = 0.1$ and $r_2 = 1$ for the APGM; and $\beta = 1$ for the SUSLM. All tested algorithms still take zero as the initial iterate. We illustrate the decomposed cartoons and textures by the SUSLM in Figure 9 at 200 iterations for test images. Visually, the target images with corruptions are basically separated as cartoons and textures. The reconstructions for the missing pixels regions in cartoons are superior to those in textures. Additionally, by superposing the decomposed cartoons and textures correspondingly, we derived the reconstructed images, i.e., "cartoon+texture", listed in Figure 9. The SNR of the reconstructed images in Figure 9 are 19.17dB for the Barbara image and 19.38dB for the Tomjerry image. We plot the evolutions of SNR with respect to iterations and computing time in seconds in Figure 10, which indicates the efficiency of the SUSLM.

6.2. **Application to an allocation problem.** In this subsection, we test an allocation problem arising from market mechanisms (see, e.g., [5,37,45]) and further verify the efficiency of the SUSLM.

We consider an economic system in which n resources are allocated by using m technological activities. The goal is to minimize the sum of cost functions of all the activities, denoted by θ_i $(i = 1, 2, \cdots, m)$. The amount of each resource is denoted by $b_j \geq 0$ $(j = 1, 2, \cdots, n)$, which justifies the nonnegativity and budget constraints. As [5], the allocation problem can be modeled as

$$\min \left\{ \sum_{i=1}^{m} \theta_i(x_i) \;\middle|\; \sum_{i=1}^{m} x_i = b,\ x_i \in \mathcal{R}_+^n,\ i = 1, 2, \cdots, m \right\}, \tag{6.7}$$

where $\theta_i : \mathcal{R}^n \to \mathcal{R}$ is the cost function of the i-th activity and the vector $b = (b_1, b_2, \cdots, b_n) \in \mathcal{R}^n$ represents all the resources.

The model (6.7) is a special case of (1.4) with

$$A_i := I \quad \text{and} \quad \mathcal{X}_i := \mathcal{R}_+^n, \quad i = 1, 2, \cdots, m.$$

Thus, the proposed SUSLM is applicable.

To specify the cost functions in (6.7), we follow the stencil functions listed in [7, Table 10.2]. Note that we skip 5 of them because the proximity functions of those scaled stencil functions are implicitly defined. More specifically, we list 12 stencil functions in Table 1 (labeled as in [7, Table 10.2]), denoted by $\phi(s)$, whose proximity functions possess closed-form solutions or can be efficiently computed by solving polynomial equations. Note that for a one-dimensional stencil function $\phi(s)$ in Table 1, it can be easily extended to an n-dimensional function $\Phi(\mathbf{s})$ whose proximity function can also be easily computed. Let us take the stencil function $\phi(s)$ listed as item (ii) in Table 1 as an illustrative example. Based on this $\phi(s)$, we can define

$$\Phi(\mathbf{s}) := \sum_{i=1}^{n} \phi_i(s_i), \tag{6.8}$$

where $\mathbf{s} = (s_1, s_2, \cdots, s_n)$, $\overline{\boldsymbol{\omega}} = (\overline{\omega}_1, \overline{\omega}_2, \cdots, \overline{\omega}_n)$, $\underline{\boldsymbol{\omega}} = (\underline{\omega}_1, \underline{\omega}_2, \cdots, \underline{\omega}_n)$ are vectors in $\mathcal{R}^n$, and

$$\phi_i(s_i) := \begin{cases} \overline{\omega}_i s_i, & \text{if } s_i \geq 0, \\ \underline{\omega}_i s_i, & \text{otherwise.} \end{cases}$$

Particularly, if $\overline{\boldsymbol{\omega}} = -\underline{\boldsymbol{\omega}} = \mathbf{1}$, we have $\Phi(\mathbf{s}) = \|\mathbf{s}\|_1$, which corresponds to the standard l_1-norm; otherwise, we have $\Phi(\mathbf{s}) = \|W\mathbf{s}\|_1$, which corresponds to the weighted l_1-norm (the diagonal entry of W is $\overline{\omega}_i$ if $s_i \geq 0$ and $\underline{\omega}_i$ if $s_i < 0$). Similarly, we can extend all the other functions listed in Table 1 to n-dimensional functions for the use of the cost functions $\{\theta_i\}_{i=1}^{12}$ in the model (6.7). The parametric vectors, e.g., $\overline{\omega}$, $\underline{\omega}$, κ, etc., are chosen randomly by following some uniform distributions (see the right column of Table 1 with $U(a, b)$ representing uniform distribution in the interval $[a, b]$). The resource amount vector b in model (6.7) is set as $b := n\mathbf{1}$.

For the allocation problem (6.7), we will test the proposed SUSLM, the ADM-G in [25], the APGM in [32], the scheme (1.8) (EADM) and the scheme (1.10) (SEADM). Recall that both (1.8) and (1.10) are not necessarily convergent, but empirically they could perform well. For the involved parameters of these iterative schemes, we chose $\beta = 1$, $r_1 = 1/6$ and $r_2 = 1/6$ for the APGM; $\beta = 1$ and $\alpha = 1$ for the ADM-G; $\beta = 1$, $\mu = 1$ and $\gamma = 1.8$ for the SUSLM; $\beta = 1$ for the SEADM and the EADM. All initial iterates are chosen as $\mathbf{1}$.

TABLE 1. The stencil function $\phi(s)$ for generating θ_i in (6.7).

Label	Stencil function $\phi(s) : \mathcal{R} \to (-\infty, +\infty]$	Parameters
i	$\begin{cases} 0, & \text{if } s \in [\underline{\omega}, \overline{\omega}] \\ +\infty, & \text{otherwise} \end{cases}$	$\overline{\omega} \sim U(10^2, 10^3)$ $\underline{\omega} \sim U(-10^2, -10^3)$
ii	$\begin{cases} \overline{\omega} s, & \text{if } s \geq 0 \\ \underline{\omega} s, & \text{otherwise} \end{cases}$	
v	$\kappa \lvert s \rvert^q$	$\omega \sim U(1,5)$ $\kappa \sim U(1,5)$ $\tau \sim U(1,5)$ $q \sim U(1,5)$
vi	$\begin{cases} \kappa s^2, & \text{if } \lvert s \rvert \leq \omega/\sqrt{2\kappa} \\ \omega\sqrt{2\kappa}\lvert s \rvert - \omega^2/2, & \text{otherwise} \end{cases}$	
vii	$\omega \lvert s \rvert + \tau \lvert s \rvert^2 + \kappa \lvert s \rvert^q$	
ix	$\begin{cases} \omega s, & \text{if } s \geq 0 \\ +\infty, & \text{otherwise} \end{cases}$	
x	$\begin{cases} -\omega s^{1/q}, & \text{if } s \geq 0 \\ +\infty, & \text{otherwise} \end{cases}$	
xi	$\begin{cases} \omega s^{-q}, & \text{if } s > 0 \\ +\infty, & \text{otherwise} \end{cases}$	$\omega \sim U(10^7, 10^8)$ $q \sim U(1,5)$
xiv	$\begin{cases} -\kappa \ln(s) + \tau s^2/2 + \alpha s, & \text{if } s > 0 \\ +\infty, & \text{otherwise} \end{cases}$	$\omega \sim U(1,5)$ $\kappa \sim U(1,5)$ $\tau \sim U(1,5)$ $q \sim U(1,5)$ $\alpha \sim U(1,5)$
xv	$\begin{cases} -\kappa \ln(s) + \alpha s + \omega s^{-1}, & \text{if } s > 0 \\ +\infty, & \text{otherwise} \end{cases}$	
xvi	$\begin{cases} -\kappa \ln(s) + \omega s^q, & \text{if } s > 0 \\ +\infty, & \text{otherwise} \end{cases}$	
xvii	$\begin{cases} -\underline{\kappa} \ln(s - \underline{\omega}) - \overline{\kappa} \ln(\overline{\omega} - s), & \text{if } s \in (\underline{\omega}, \overline{\omega}) \\ +\infty, & \text{otherwise} \end{cases}$	$\overline{\kappa} \sim U(10^{-3}, 10^{-1})$ $\underline{\kappa} \sim U(10^{-3}, 10^{-1})$ $\overline{\omega} \sim U(10^3, 10^4)$ $\underline{\omega} \sim U(-10^3, -10^4)$

To compare the efficiency of different algorithms for solving the model (6.7), we measure the quality of iterates by both the objective function value and the constraint violation. More specifically, for the iterate x_i^k, the constraint violation is measured by

$$\frac{\left\| \sum_{i=1}^n x_i^k - b \right\|}{1 + \|b\|}. \tag{6.9}$$

Note that the nonnegativity constraint is satisfied automatically for all the sequences generated by the algorithms to be tested.

In Figure 11, we plot the evolutions of the objective function values (Obj-Fun-Val for short) with respect to the computing time for the cases of n where $n = 100, 200, 300, 500, 800, 1000$. Similar plots with respect to the iterations can also be displayed, but we omit them for succinctness. The evaluations of the constraint violation with respect to the computing time are plotted in Figure 12. These figures show that both the EADM and SEADM, though theoretically lack convergence, perform well for the allocation problem (6.7). Furthermore, among the methods with provable convergence, the proposed SUSLM performs the best.

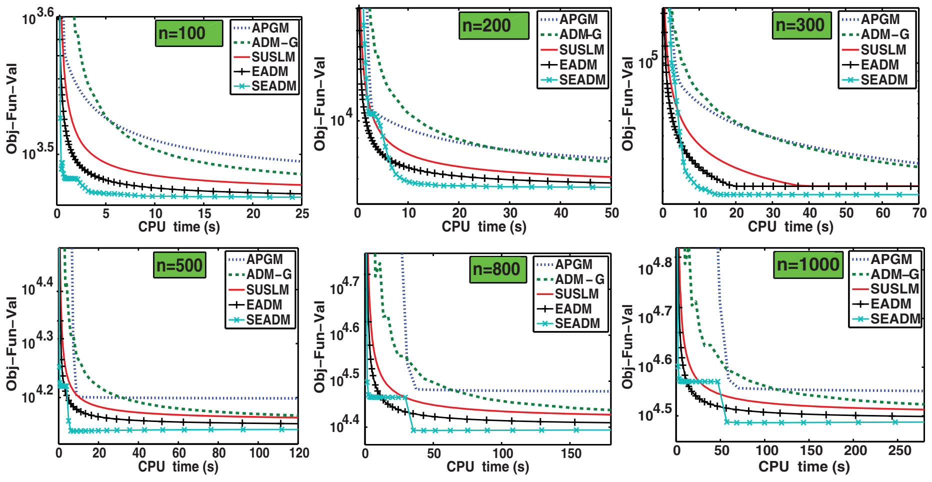


FIGURE 11. Evolutions of objective function values w.r.t. computing time for variant n's.

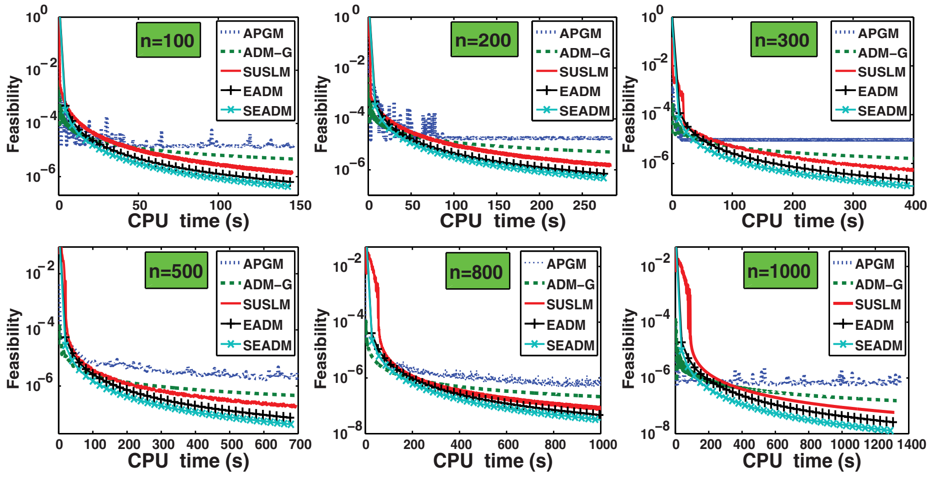


FIGURE 12. Evolutions of the constraint violation measured by (6.9) w.r.t. computing time for variant n's.

7. CONCLUDING REMARKS

In this paper we focused on how to split the augmented Lagrangian method (ALM) for a separable convex minimization problem with linear constraints and an objective function in the sum of m functions without coupled variables. A sequential updating scheme of the Lagrange multiplier was proposed. The new scheme splits the ALM subproblem at each iteration into m smaller minimization problems such that each of them only involves one function in the original objective. As most of the ALM-based splitting methods in the literature, the new scheme preserves the possibility of taking advantgae of the objective functions' properties individually. But it differs from the existing ALM-based splitting methods in that the Lagrange multiplier is updated timely (m times) once each block of primal variables are solved. We derive both the exact and inexact algorithms based on the new scheme. An image decomposition problem and a resource allocation problem have been tested to demonstrate the numerical efficiency of the proposed sequential updating scheme of the Lagrange multiplier.

Acknowledgments

The authors would like to thank the anonymous referees for their valuable comments, which helped the authors improve the presentation of this paper substantially.

References

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

[2] H. H. Bauschke and P. L. Combettes, *Convex analysis and monotone operator theory in Hilbert spaces*, with a foreword by Hédy Attouch. CMS Books in Mathematics/Ouvrages de Mathématiques de la SMC, Springer, New York, 2011. MR2798533 (2012h:49001)

[3] D. P. Bertsekas, *Constrained optimization and Lagrange multiplier methods*, Computer Science and Applied Mathematics, Academic Press, Inc. [Harcourt Brace Jovanovich, Publishers], New York-London, 1982. MR690767 (84k:90068)

[4] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*, Athena Scientific, 1996.

[5] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Found. Trends Mach. Learn. **3** (2010), 1–122.

[6] C. Chen, B. He, Y. Ye, and X. Yuan, *The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent*, Math. Program. **155** (2016), no. 1-2, Ser. A, 57–79, DOI 10.1007/s10107-014-0826-5. MR3439797

[7] P. L. Combettes and J.-C. Pesquet, *Proximal splitting methods in signal processing*, Fixed-point algorithms for inverse problems in science and engineering, Springer Optim. Appl., vol. 49, Springer, New York, 2011, pp. 185–212, DOI 10.1007/978-1-4419-9569-8_10. MR2858838 (2012i:90117)

[8] E. Corman and X. Yuan, *A generalized proximal point algorithm and its convergence rate*, SIAM J. Optim. **24** (2014), no. 4, 1614–1638, DOI 10.1137/130940402. MR3268621

[9] J. Douglas Jr. and H. H. Rachford Jr., *On the numerical solution of heat conduction problems in two and three space variables*, Trans. Amer. Math. Soc. **82** (1956), 421–439. MR0084194 (18,827f)

[10] J. Eckstein and W. Yao, *Understanding the convergence of the alternating direction method of multipliers: theoretical and computational perspectives*, Pac. J. Optim. **11** (2015), no. 4, 619–644. MR3420805

[11] F. Facchinei and J.-S. Pang, *Finite-dimensional variational inequalities and complementarity problems. Vols. I, II*, Springer Series in Operations Research, Springer-Verlag, New York, 2003. MR1955648 (2004g:90003a)

[12] D. Gabay, *Applications of the method of multipliers to variational inequalities*, In: *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems* (M. Fortin, R. Glowinski), North-Holland, Amsterdam, 1983, pp. 299–331.

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

[14] R. Glowinski, *Numerical methods for nonlinear variational problems*, Springer Series in Computational Physics, Springer-Verlag, New York, 1984. MR737005 (86c:65004)

[15] R. Glowinski, *On alternating direction methods of multipliers: a historical perspective*, Modeling, simulation and optimization for science and technology, Comput. Methods Appl. Sci., vol. 34, Springer, Dordrecht, 2014, pp. 59–82, DOI 10.1007/978-94-017-9054-3_4. MR3330832

[16] R. Glowinski and P. Le Tallec, *Augmented Lagrangian and operator-splitting methods in nonlinear mechanics*, SIAM Studies in Applied Mathematics, vol. 9, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1989. MR1060954 (91f:73038)

[17] R. Glowinski, T. Kärkkäinen, and K. Majava, *On the convergence of operator-splitting methods*, Numerical methods for scientific computing. Variational problems and applications, Internat. Center Numer. Methods Eng. (CIMNE), Barcelona, 2003, pp. 67–79. MR2427782 (2009j:65284)

[18] R. Glowinski and A. Marrocco, *Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité, d'une classe de problèmes de Dirichlet non linéaires* (French, with Loose English summary), Rev. Française Automat. Informat. Recherche Opérationnelle Sér. Rouge Anal. Numér. **9** (1975), no. R-2, 41–76. MR0388811 (52 #9645)

[19] T. Goldstein and S. Osher, *The split Bregman method for $L1$-regularized problems*, SIAM J. Imaging Sci. **2** (2009), no. 2, 323–343, DOI 10.1137/080725891. MR2496060 (2010e:65087)

[20] D. Han and X. Yuan, *A note on the alternating direction method of multipliers*, J. Optim. Theory Appl. **155** (2012), no. 1, 227–238, DOI 10.1007/s10957-012-0003-z. MR2983116

[21] P. C. Hansen, J. G. Nagy, and D. P. O'Leary, *Deblurring images*, Matrices, spectra, and filtering, Fundamentals of Algorithms, vol. 3, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2006. MR2271138 (2008d:94007)

[22] B. He, L.-Z. Liao, D. Han, and H. Yang, *A new inexact alternating directions method for monotone variational inequalities*, Math. Program. **92** (2002), no. 1, Ser. A, 103–118, DOI 10.1007/s101070100280. MR1892298 (2003b:90111)

[23] B. S. He, H. Liu, J. Lu, and X. M. Yuan, *Application of the strictly contractive Peaceman-Rachford splitting method to multi-block convex programming*, manuscript (2014).

[24] B. He, H. Liu, Z. Wang, and X. Yuan, *A strictly contractive Peaceman-Rachford splitting method for convex programming*, SIAM J. Optim. **24** (2014), no. 3, 1011–1040, DOI 10.1137/13090849X. MR3231988

[25] B. He, M. Tao, and X. Yuan, *Alternating direction method with Gaussian back substitution for separable convex programming*, SIAM J. Optim. **22** (2012), no. 2, 313–340, DOI 10.1137/110822347. MR2968856

[26] B. He and H. Yang, *Some convergence properties of a method of multipliers for linearly constrained monotone variational inequalities*, Oper. Res. Lett. **23** (1998), no. 3-5, 151–161, DOI 10.1016/S0167-6377(98)00044-3. MR1677664 (2000d:90089)

[27] B. He and X. Yuan, *On the $O(1/n)$ convergence rate of the Douglas-Rachford alternating direction method*, SIAM J. Numer. Anal. **50** (2012), no. 2, 700–709, DOI 10.1137/110836936. MR2914282

[28] M. R. Hestenes, *Multiplier and gradient methods*, J. Optimization Theory Appl. **4** (1969), 303–320. MR0271809 (42 #6690)

[29] M. Hong and Z. Q. Luo, *On the linear convergence of alternating direction method of multipliers*, Math. Program., to appear.

[30] P.-L. Lions and B. Mercier, *Splitting algorithms for the sum of two nonlinear operators*, SIAM J. Numer. Anal. **16** (1979), no. 6, 964–979, DOI 10.1137/0716071. MR551319 (81g:47070)

[31] Z.-Q. Luo and P. Tseng, *On the convergence rate of dual ascent methods for linearly constrained convex minimization*, Math. Oper. Res. **18** (1993), no. 4, 846–867, DOI 10.1287/moor.18.4.846. MR1251683 (95a:90070)

[32] S. Q. Ma, *Alternating proximal gradient method for convex minimization*, preprint (2012).

[33] B. Martinet, *Régularisation d'inéquations variationnelles par approximations successives* (French), Rev. Française Informat. Recherche Opérationnelle **4** (1970), no. Ser. R-3, 154–158. MR0298899 (45 #7948)

[34] Y. Meyer, *Oscillating patterns in image processing and nonlinear evolution equations*, The fifteenth Dean Jacqueline B. Lewis memorial lectures. University Lecture Series, vol. 22, American Mathematical Society, Providence, RI, 2001. MR1852741 (2002j:43001)

[35] J.-J. Moreau, *Proximité et dualité dans un espace hilbertien* (French), Bull. Soc. Math. France **93** (1965), 273–299. MR0201952 (34 #1829)

[36] S. Osher, A. Solé, and L. Vese, *Image decomposition and restoration using total variation minimization and the H^{-1} norm*, Multiscale Model. Simul. **1** (2003), no. 3, 349–370 (electronic), DOI 10.1137/S1540345902416247. MR2030155 (2004k:49004)

[37] M. Patriksson, *A survey on the continuous nonlinear resource allocation problem*, European J. Oper. Res. **185** (2008), no. 1, 1–46, DOI 10.1016/j.ejor.2006.12.006. MR2358684 (2008g:91153)

[38] D. W. Peaceman and H. H. Rachford Jr., *The numerical solution of parabolic and elliptic differential equations*, J. Soc. Indust. Appl. Math. **3** (1955), 28–41. MR0071874 (17,196d)

[39] M. J. D. Powell, *A method for nonlinear constraints in minimization problems*, Optimization (Sympos., Univ. Keele, Keele, 1968), Academic Press, London, 1969, pp. 283–298. MR0272403 (42 #7284)

[40] H. Robbins and D. Siegmund, *A convergence theorem for non negative almost supermartingales and some applications*, Optimizing methods in statistics (Proc. Sympos., Ohio State Univ., Columbus, Ohio, 1971), Academic Press, New York, 1971, pp. 233–257. MR0343355 (49 #8097)

[41] R. T. Rockafellar, *Monotone operators and the proximal point algorithm*, SIAM J. Control Optimization **14** (1976), no. 5, 877–898. MR0410483 (53 #14232)

[42] L. I. Rudin, S. Osher, and E. Fatemi, *Nonlinear total variation based noise removal algorithms*, Phys. D **60** (1992), no. 1-4, 259–268. MR3363401

[43] H. Schaeffer and S. Osher, *A low patch-rank interpretation of texture*, SIAM J. Imaging Sci. **6** (2013), no. 1, 226–262, DOI 10.1137/110854989. MR3032953

[44] J.-L. Starck, M. Elad, and D. L. Donoho, *Image decomposition via the combination of sparse representations and a variational approach*, IEEE Trans. Image Process. **14** (2005), no. 10, 1570–1582, DOI 10.1109/TIP.2005.852206. MR2483314

[45] H. Uzawa, *Market mechanisms and mathematical programming*, Econometrica **28** (1960), 872–881. MR0136447 (24 #B2481)

[46] L. A. Vese and S. J. Osher, *Modeling textures with total variation minimization and oscillating patterns in image processing*, Special issue in honor of the sixtieth birthday of Stanley Osher, J. Sci. Comput. **19** (2003), no. 1-3, 553–572, DOI 10.1023/A:1025384832106. MR2028858 (2004k:49006)

LSEC, Institute of Computational Mathematics, Chinese Academy of Sciences, P.O. Box 2719, Beijing 100190, People's Republic of China
E-mail address: `dyh@lsec.cc.ac.cn`

School of Mathematical Sciences, Jiangsu Key Laboratory for NSLSCS, Nanjing Normal University, Nanjing 210023, People's Republic of China
E-mail address: `handeren@njnu.edu.cn`

Department of Mathematics, Hong Kong Baptist University, Kowloon, Hong Kong, People's Republic of China
E-mail address: `xmyuan@hkbu.edu.hk`

School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu 611731, People's Republic of China
E-mail address: `wxzh1984@126.com`