Noname manuscript No.
(will be inserted by the editor)

A parallel splitting ALM-based algorithm for separable convex programming

Shengjie Xu · Bingsheng He

Received: date / Accepted: date

Abstract The augmented Lagrangian method (ALM) provides a benchmark for tackling the canonical convex minimization problem with linear constraints. We consider a special case where the objective function is the sum of m individual subfunctions without coupled variables. The recent study reveals that the direct extension of ALM for separable convex programming problems is not necessarily convergent, even though it is empirically efficient for many applications. It has thus inspired a number of variants. In this paper, for parallel computing, we propose a novel operator splitting method for solving the separable convex minimization problem. We first show that a fully Jacobian decomposition of the regularized ALM can contribute a descent direction yielding the contraction of proximity to the solution set; then, we generate the new iterate via a simple correction step with a constant step size. We establish the convergence analysis for the proposed method, and then illustrate its numerical efficiency by the latent variable Gaussian graphical model selection problem arising in statistical learning.

Keywords Convex programming · Augmented Lagrangian method · Jacobian decomposition · Parallel computing · Operator splitting methods

Mathematics Subject Classification (2010) 90C25 · 90C06 · 90C33

Shengjie Xu
Department of Mathematics, Harbin Institute of Technology, Harbin, China, and Department of Mathematics, Southern University of Science and Technology, Shenzhen, China.
E-mail: xsjnsu@163.com

Bingsheng He
Department of Mathematics, Nanjing University, Nanjing, China.
E-mail: hebma@nju.edu.cn

1 Introduction

Our discussion starts with the canonical convex minimization problem with linear constraints:

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

where $\theta(x) : \Re^n \to \Re$ is a closed proper convex (but not necessarily smooth) function; $\mathcal{X} \subset \Re^n$ is a nonempty closed convex set; $\mathcal{A} \in \Re^{l\times n}$ and $b \in \Re^l$. Among algorithms for solving (1.1), the augmented Lagrangian method (ALM) in [23,28] turns out to be a fundamental tool in both theoretical and algorithmic aspects, and its associated iterative scheme reads as

$$\text{(ALM)} \begin{cases} x^{k+1} = \ \arg\min\big\{\mathcal{L}_\beta(x, \lambda^k) \mid x \in \mathcal{X}\big\}, \\ \lambda^{k+1} = \ \lambda^k - \beta(\mathcal{A}x^{k+1} - b), \end{cases} \tag{1.2}$$

where

$$\mathcal{L}_\beta(x, \lambda) = \theta(x) - \lambda^T(\mathcal{A}x - b) + \frac{\beta}{2}\|\mathcal{A}x - b\|_2^2$$

denotes the augmented Lagrangian function of (1.1); $\lambda \in \Re^l$ is the Lagrange multiplier and $\beta > 0$ is a penalty parameter for the linear constraints. As analyzed in [29], the classic ALM is indeed an application of the proximal point algorithm (PPA) in [25] to the dual of (1.1). Throughout our discussion, the penalty parameter β is assumed to be fixed for simplification.

In this paper, we focus on a special case of (1.1), where its objective function is the sum of m subfunctions without coupled variables:

$$\min\Big\{\sum_{i=1}^m \theta_i(x_i) \mid \sum_{i=1}^m A_i x_i = b;\ x_i \in \mathcal{X}_i,\ i = 1, 2, \ldots, m\Big\}. \tag{1.3}$$

Here, $\theta_i(x_i) : \Re^{n_i} \to \Re$ $(i = 1, \ldots, m)$ are closed proper convex (could be nonsmooth) functions; $\mathcal{X}_i \subset \Re^{n_i}$ $(i = 1, \ldots, m)$ are nonempty closed convex sets; $\sum_{i=1}^m n_i = n$; $A_i \in \Re^{l\times n_i}$ $(i = 1, \ldots, m)$ and $b \in \Re^l$. Obviously, the aim model (1.3) corresponds to (1.1) with $x = (x_1; \ldots; x_m)$, $\mathcal{A} = (A_1, \ldots, A_m)$ and $\theta(x) = \sum_{i=1}^m \theta_i(x_i)$. The model (1.3) finds a lot of applications in, e.g., [4,30] for some image reconstruction problems, [31,35] for the matrix recovery model and the Potts based image segmentation problem, and [5,6,32] for a number of problems arising in distributed optimization and statistical learning. We also refer to, e.g., [24,34], for more applications in other communities. Throughout our discussion, the solution set of (1.3) is assumed to be nonempty and the matrices A_i $(i = 1, \ldots, m)$ are assumed to be full column-rank. In addition, we assume each x_i-subproblem

$$x_i^* = \arg\min\{\theta_i(x_i) + \frac{\beta}{2}\|A_i x_i - q_i\|_2^2 \ \mid\ x_i \in \mathcal{X}_i\}, \tag{1.4}$$

with any given $q_i \in \Re^l$ and $\beta > 0$, has a closed-form solution or can be easily solved with a high precision (see, e.g., [26], for some specific examples).

Apply the generic ALM (1.2) straightforwardly to (1.3); the resulting iterative scheme reads as

$$\begin{cases} (x_1^{k+1},\dots,x_m^{k+1}) = \arg\min\big\{\mathcal{L}_\beta(x_1,x_2,\dots,x_m,\lambda^k)\,|\,x_i\in\mathcal{X}_i,\ i=1,\dots,m\big\},\\ \qquad\qquad \lambda^{k+1} = \lambda^k - \beta(\sum_{i=1}^m A_i x_i^{k+1} - b),\end{cases}$$

where

$$\mathcal{L}_\beta(x_1,x_2,\dots,x_m,\lambda) = \sum_{i=1}^m \theta_i(x_i) - \lambda^T(\sum_{i=1}^m A_i x_i - b) + \frac{\beta}{2}\|\sum_{i=1}^m A_i x_i - b\|_2^2. \tag{1.5}$$

However, note there are coupled terms in the quadratic term $\|\sum_{i=1}^m A_i x_i - b\|_2^2$. The x_i-subproblems ($i=1,\dots,m$) generally could not be tackled simultaneously. Exploiting the separable structure of the objective function $\theta(x)$, a common-used operator splitting strategy for x-subproblem is the following Gaussian-decomposition iterative scheme:

$$\begin{cases} \begin{cases} x_1^{k+1} = \arg\min\big\{\mathcal{L}_\beta(x_1,x_2^k,\dots,x_{m-1}^k,x_m^k,\lambda^k)\mid x_1\in\mathcal{X}_1\big\},\\ x_2^{k+1} = \arg\min\big\{\mathcal{L}_\beta(x_1^{k+1},x_2,\dots,x_{m-1}^k,x_m^k,\lambda^k)\mid x_2\in\mathcal{X}_2\big\},\\ \qquad\qquad \vdots \\ x_m^{k+1} = \arg\min\big\{\mathcal{L}_\beta(x_1^{k+1},x_2^{k+1},\dots,x_{m-1}^{k+1},x_m,\lambda^k)\mid x_m\in\mathcal{X}_m\big\},\end{cases}\\ \lambda^{k+1} = \lambda^k - \beta(\sum_{i=1}^m A_i x_i^{k+1} - b).\end{cases} \tag{1.6}$$

It is also named the direct extension of ADMM (denoted by D-ADMM). Clearly, in (1.6), each x_i-subproblem possesses the favorable composition (1.4) and is treated individually. In particular, for the case $m=2$, the method (1.6) corresponds to the classical alternating direction method of multipliers (ADMM) in [9,10], and it is indeed an application of the Douglas-Rachford splitting method (see [8]). For the case $m\geq 3$, the D-ADMM has been shown to be empirically efficient for various applications, see, e.g., [27,31] and references cited therein. However, a counterexample in [7] reveals that the D-ADMM is not necessarily convergent. It has immediately inspired a number of interesting works such as [11,13,17,18,19,22,24,30,31,32]. Especially, the ADMM with Gaussian back substitution (abbreviated as GS-ADMM) proposed in [12,17,19], which needs only an additional back substitution procedure based on the D-ADMM, turns out to be a convergent method that performs competitively.

We now review another common-used operator splitting strategy (for parallel computing) for the x-subproblem in the generic ALM scheme (1.2). Applying the full Jacobian-decomposition of x-subproblem to (1.2), it immediately leads to the following direct extension of ALM (abbreviated as D-ALM):

$$
\text{(D-ALM)}\quad \begin{cases} \begin{cases} x_1^{k+1} = \arg\min\big\{\mathcal{L}_\beta(x_1, x_2^k, \ldots, x_{m-1}^k, x_m^k, \lambda^k) \mid x_1 \in \mathcal{X}_1\big\}, \\ x_2^{k+1} = \arg\min\big\{\mathcal{L}_\beta(x_1^k, x_2, \ldots, x_{m-1}^k, x_m^k, \lambda^k) \mid x_2 \in \mathcal{X}_2\big\}, \\ \qquad\qquad \vdots \\ x_m^{k+1} = \arg\min\big\{\mathcal{L}_\beta(x_1^k, x_2^k, \ldots, x_{m-1}^k, x_m, \lambda^k) \mid x_m \in \mathcal{X}_m\big\}, \end{cases} \\ \lambda^{k+1} \;=\; \lambda^k - \beta(\sum_{i=1}^m A_i x_i^{k+1} - b). \end{cases}
$$

In fact, as shown in [36], there are many applications in the medical image processing field such as the Potts-model based image segmentation problem and its variants, can be solved efficiently by the D-ALM. However, a simple example in [14] demonstrates that the D-ALM is not convergent in theory. For convergence guarantee, in [14], the authors first regard the D-ALM as a predictor and then do an additional correction step $w^{k+1} := w^k - \alpha(w^k - w^{k+1})$, where $w = (x_1; x_2; \ldots; x_m; \lambda)$. And this simple variant (denoted by P-ALM) is convergent when $\alpha \in (0,\ 2(1 - \sqrt{m/(m+1)}))$. It is easy to verify that, as the increase of m, the step size α would be tiny (for example, if $m = 3$, then $\alpha \in (0, 0.268)$), which would adversely affect the convergence efficiency in the real computations. In [18], by adding extra regularization terms to the D-ALM, another operator splitting ALM-based method is proposed, and it reads as

$$
\begin{cases} x_1^{k+1} \;=\; \arg\min\big\{\mathcal{L}_\beta(x_1, x_2^k, \ldots, x_{m-1}^k, x_m^k, \lambda^k) \mid x_1 \in \mathcal{X}_1\big\}, \\ \begin{cases} x_2^{k+1} \;=\; \arg\min\big\{\mathcal{L}_\beta(x_1^{k+1}, x_2, \ldots, x_{m-1}^k, x_m^k, \lambda^k) \\ \qquad\qquad\qquad + \dfrac{\tau}{2}\beta\|A_2(x_2 - x_2^k)\|_2^2 \mid x_2 \in \mathcal{X}_2\big\}, \\ \qquad\qquad \vdots \\ x_m^{k+1} \;=\; \arg\min\big\{\mathcal{L}_\beta(x_1^{k+1}, x_2^k, \ldots, x_{m-1}^k, x_m, \lambda^k) \\ \qquad\qquad\qquad + \dfrac{\tau}{2}\beta\|A_m(x_m - x_m^k)\|_2^2 \mid x_m \in \mathcal{X}_m\big\}, \end{cases} \\ \lambda^{k+1} \;=\; \lambda^k - \beta(\sum_{i=1}^m A_i x_i^{k+1} - b). \end{cases}
$$

This algorithm (denoted by PS-ADMM) is convergent provided $\tau > m - 2$. Clearly, it would correspond to a huge regularization term (or equivalently, a tiny step size) as the increase of m. Recent works such as [15,16,33] have demonstrated that relaxation of regularization term allows a bigger step size to reduce required convergence iterations potentially. Hence, it is necessary to reduce such a regularization factor τ.

The primary purpose of the paper is to present a novel parallel splitting ALM-based algorithm for the separable convex programming problem (1.3). More concretely, our new method begins with the following fully parallel regularized ALM-based scheme:

$$\left\{\begin{array}{l}\left\{\begin{array}{lcl}\tilde{x}_1^k &=& \arg\min\big\{\mathcal{L}_\beta(x_1, x_2^k, \ldots, x_{m-1}^k, x_m^k, \lambda^k) \\ && \qquad\qquad + \dfrac{\tau}{2}\beta\|A_1(x_1 - x_1^k)\|_2^2 \mid x_1 \in \mathcal{X}_1\big\}, \\ \tilde{x}_2^k &=& \arg\min\big\{\mathcal{L}_\beta(x_1^k, x_2, \ldots, x_{m-1}^k, x_m^k, \lambda^k) \\ && \qquad\qquad + \dfrac{\tau}{2}\beta\|A_2(x_2 - x_2^k)\|_2^2 \mid x_2 \in \mathcal{X}_2\big\}, \\ && \vdots \\ \tilde{x}_m^k &=& \arg\min\big\{\mathcal{L}_\beta(x_1^k, x_2^k, \ldots, x_{m-1}^k, x_m, \lambda^k) \\ && \qquad\qquad + \dfrac{\tau}{2}\beta\|A_m(x_m - x_m^k)\|_2^2 \mid x_m \in \mathcal{X}_m\big\},\end{array}\right. \\ \\ \quad\tilde{\lambda}^k \;=\; \lambda^k - \beta(\sum_{i=1}^m A_i x_i^k - b),\end{array}\right. \tag{1.7}$$

where $\tau > (m-4)/4$. As can be seen easily, the scheme (1.7) possesses a fully parallel splitting structure for both x_i-subproblems ($i = 1, \ldots, m$) and λ-subproblem. It thus belongs to the category of parallel operator splitting schemes. Furthermore, as will be shown in Subsect. 3.1, the scheme (1.7) can provide a descent direction yielding the contraction of proximity to the solution set of (1.3). It is thus inspired to generate the new iterate w^{k+1} via

$$w^{k+1} = w^k + \alpha d(w^k, \tilde{w}^k) := w^k - \alpha M(w^k - \tilde{w}^k), \tag{1.8}$$

where $\alpha \in (0,1)$ and the correction matrix M will be defined in (2.11). Note that the step size α is a constant. The correction step (1.8), as the initialization of next iteration, could be handled easily. Moreover, we give a more practical and succinct scheme (3.7) to replace the method (1.7)-(1.8) in Subsect. 3.3.

The novel algorithm (1.7)-(1.8) enjoys great advantages in two folds: first, compared with the P-ALM, the constant step size $\alpha \in (0,1)$ can be taken more broadly; second, it reduces the regularization factor τ from $(m-2)$ to $(m-4)/4$ compared with the PS-ADMM, so it provides a wider choice of the regularization term to potentially accelerate the convergence. We also show the global convergence for the proposed method.

The rest of the paper is organized as follows. In Sect. 2, we summarize some preliminaries and fundamental matrices, which will be useful for further analysis. Then, we propose the novel method in Sect. 3 and establish its convergence analysis in Sect. 4. The numerical efficiency of the proposed method is further reported in Sect. 5. Some conclusions are drawn in Sect. 6.

2 Preliminaries

In this section, we summarize some preliminary results that will be used throughout our discussion. The analysis of the paper is based on the following elementary lemma (its proof can be found in, e.g., [2]).

Lemma 1 *Let $\mathcal{X} \subset \Re^n$ be a closed convex set, $\theta(x)$ and $\varphi(x)$ be convex functions. If $\varphi(x)$ is differentiable on an open set which contains $\mathcal{X}$, and the solution set of the convex minimization problem* $\min\{\theta(x) + \varphi(x) \mid x \in \mathcal{X}\}$ *is nonempty, then we have*

$$x^* \in \arg\min\{\theta(x) + \varphi(x) \mid x \in \mathcal{X}\}$$

if and only if

$$x^* \in \mathcal{X}, \quad \theta(x) - \theta(x^*) + (x - x^*)^T \nabla\varphi(x^*) \geq 0, \quad \forall\, x \in \mathcal{X}.$$

2.1 A variational characterization of (1.3)

To begin with, using the similar techniques in, e.g., [21], we show how to stand for the optimality condition of (1.3) in the variational inequality context. By attaching the Lagrange multiplier $\lambda \in \Re^l$ to the linear constraints, the Lagrange function of (1.3) reads as

$$L(x_1, x_2, \ldots, x_m, \lambda) = \sum_{i=1}^{m} \theta_i(x_i) - \lambda^T (\sum_{i=1}^{m} A_i x_i - b), \tag{2.1}$$

and it is defined on the set $\Omega := \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_m \times \Re^l$. Suppose the pair $(x_1^*, x_2^*, \ldots, x_m^*, \lambda^*) \in \Omega$ is a saddle point of (2.1). Then, for any $\lambda \in \Re^l$ and $x_i \in \mathcal{X}_i\,(i = 1, \ldots, m)$, we have

$$L(x_1^*, x_2^*, \ldots, x_m^*, \lambda) \leq L(x_1^*, x_2^*, \ldots, x_m^*, \lambda^*) \leq L(x_1, x_2, \ldots, x_m, \lambda^*).$$

In view of Lemma 1, it is equivalent to finding $(x_1^*; \ldots; x_m^*; \lambda^*)$ such that

$$\begin{cases} \theta_1(x_1) - \theta_1(x_1^*) + (x_1 - x_1^*)^T (-A_1^T \lambda^*) \geq 0, & \forall\, x_1 \in \mathcal{X}_1, \\ \qquad\qquad \vdots \\ \theta_m(x_m) - \theta_m(x_m^*) + (x_m - x_m^*)^T (-A_m^T \lambda^*) \geq 0, & \forall\, x_m \in \mathcal{X}_m, \\ (\lambda - \lambda^*)^T (\sum_{i=1}^m A_i x_i^* - b) \geq 0, & \forall\, \lambda \in \Re^l. \end{cases} \tag{2.2}$$

Moreover, if we set

$$\begin{aligned} &x = \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix}, \quad \theta(x) = \textstyle\sum_{i=1}^m \theta_i(x_i), \quad \mathcal{A} = (A_1, A_2, \ldots, A_m), \\ &w = \begin{pmatrix} x_1 \\ \vdots \\ x_m \\ \lambda \end{pmatrix} \quad \text{and} \quad F(w) = \begin{pmatrix} -A_1^T \lambda \\ \vdots \\ -A_m^T \lambda \\ \sum_{i=1}^m A_i x_i - b \end{pmatrix}, \end{aligned} \tag{2.3}$$

the inequalities (2.2) can be compactly rewritten as

$$\mathrm{VI}(\Omega, F, \theta): \quad w^* \in \Omega, \quad \theta(x) - \theta(x^*) + (w - w^*)^T F(w^*) \geq 0, \quad \forall\, w \in \Omega. \tag{2.4}$$

In the later sections, we denote by Ω^* the solution set of (2.4), which is assumed to be nonempty. On the one hand, since the operator $F(w)$ in (2.3) is affine with a skew-symmetric matrix, we obtain

$$(w-\bar{w})^T(F(w)-F(\bar{w})) \equiv 0, \quad \forall\, w,\ \bar{w} \in \Omega, \tag{2.5}$$

which means F is monotone. On the other hand, the objective function $\theta(x)$ is not necessarily differentiable. With this regard, we also name (2.4) the mixed monotone variational inequality. In the following, we say (2.4) the variational inequality (VI) for short.

2.2 A variational characterization of (1.7)

The associated VI-structure of the parallel splitting scheme (1.7) can be deduced by the following fundamental lemma.

Lemma 2 *Let* $\{\tilde{w}^k\}$ *be the sequence generated by* (1.7) *for solving* (1.3). *Then, we have*

$$\theta(x)-\theta(\tilde{x}^k)+(w-\tilde{w}^k)^T F(\tilde{w}^k) \geq (w-\tilde{w}^k)^T Q(w^k-\tilde{w}^k), \quad \forall\, w \in \Omega, \tag{2.6}$$

where

$$Q = \begin{pmatrix} (1+\tau)\beta A_1^T A_1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & (1+\tau)\beta A_m^T A_m & 0 \\ -A_1 & \cdots & -A_m & \frac{1}{\beta} I_l \end{pmatrix}. \tag{2.7}$$

Proof To begin with, for the x_i-subproblem in (1.7), it follows from Lemma 1 that $\tilde{x}_i^k \in \mathcal{X}_i$, and $\tilde{x}_i^k$ satisfies

$$\begin{aligned}&\theta_i(x_i)-\theta_i(\tilde{x}_i^k)+(x_i-\tilde{x}_i^k)^T \\ &\quad \Big\{-A_i^T\big(\lambda^k-\beta(A_i\tilde{x}_i^k+\sum_{j\neq i} A_j x_j^k - b)\big)+\tau\beta A_i^T A_i(\tilde{x}_i^k-x_i^k)\Big\} \geq 0,\ \forall\, x_i \in \mathcal{X}_i.\end{aligned}$$

Since $\tilde{\lambda}^k = \lambda^k - \beta(\mathcal{A}x^k - b)$ in (1.7), we obtain

$$\begin{aligned}&-A_i^T\big(\lambda^k-\beta(A_i\tilde{x}_i^k+\sum_{j\neq i} A_j x_j^k - b)\big)+\tau\beta A_i^T A_i(\tilde{x}_i^k-x_i^k) \\ &\quad = -A_i^T[\lambda^k-\beta(\mathcal{A}x^k-b)]+\beta A_i^T A_i(\tilde{x}_i^k-x_i^k)+\tau\beta A_i^T A_i(\tilde{x}_i^k-x_i^k) \\ &\quad = -A_i^T\tilde{\lambda}^k+(1+\tau)\beta A_i^T A_i(\tilde{x}_i^k-x_i^k).\end{aligned}$$

Therefore, the x_i-subproblem in (1.7) satisfies

$$\begin{aligned}&\theta_i(x_i)-\theta_i(\tilde{x}_i^k) \\ &\quad +(x_i-\tilde{x}_i^k)^T\big\{-A_i^T\tilde{\lambda}^k+(1+\tau)\beta A_i^T A_i(\tilde{x}_i^k-x_i^k)\big\} \geq 0, \quad \forall\, x_i \in \mathcal{X}_i.\end{aligned} \tag{2.8}$$

For the λ-subproblem in (1.7), it is easy to see that $\mathcal{A}x^k - b + \frac{1}{\beta}(\tilde{\lambda}^k - \lambda^k) = 0$, which is also equivalent to

$$(\lambda - \tilde{\lambda}^k)^T\{\mathcal{A}\tilde{x}^k - b - \mathcal{A}(\tilde{x}^k - x^k) + \frac{1}{\beta}(\tilde{\lambda}^k - \lambda^k)\} = 0, \quad \forall\, \lambda \in \Re^l. \tag{2.9}$$

Using the notations in (2.3), the assertion of the lemma follows immediately by adding (2.8) and (2.9). $\square$

2.3 Some basic matrices

We list some fundamental matrices which will be useful for further discussion. Recall the matrix Q defined in (2.7). Setting $S = Q^T + Q$, we get

$$\begin{aligned} S = Q^T + Q &= \begin{pmatrix} 2(1+\tau)\beta A_1^T A_1 & \cdots & 0 & -A_1^T \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 2(1+\tau)\beta A_m^T A_m & -A_m^T \\ -A_1 & \cdots & -A_m & \frac{2}{\beta} I_l \end{pmatrix} \\ &= \mathrm{diag}(\begin{pmatrix} A_1^T \\ \vdots \\ A_m^T \\ I_l \end{pmatrix}) \begin{pmatrix} 2(1+\tau)\beta I_l & \cdots & 0 & -I_l \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 2(1+\tau)\beta I_l & -I_l \\ -I_l & \cdots & -I_l & \frac{2}{\beta} I_l \end{pmatrix} \mathrm{diag}(\begin{pmatrix} A_1 \\ \vdots \\ A_m \\ I_l \end{pmatrix}). \end{aligned} \tag{2.10}$$

Note that the matrices A_i $(i = 1, \ldots, m)$ are assumed to be full column-rank. Clearly, the matrix S is symmetric, and $S \succ 0$ provided $\tau > (m-4)/4$. Throughout, $\tau > (m-4)/4$ is required to guarantee the positive definiteness of the matrix S. Furthermore, we define $M = Q^{-T}S$, and thus

$$\begin{aligned} M = &\frac{1}{1+\tau}\mathrm{diag}((A_1^T A_1)^{-1}, \ldots,\ (A_m^T A_m)^{-1},\ I_l) \\ &\begin{pmatrix} (2\tau+1)A_1^T A_1 & -A_1^T A_2 & \cdots & -A_1^T A_m & \frac{1}{\beta} A_1^T \\ -A_2^T A_1 & (2\tau+1)A_2^T A_2 & \cdots & -A_2^T A_m & \frac{1}{\beta} A_2^T \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ -A_m^T A_1 & -A_m^T A_2 & \cdots & (2\tau+1)A_m^T A_m & \frac{1}{\beta} A_m^T \\ -(1+\tau)\beta A_1 & -(1+\tau)\beta A_2 & \cdots & -(1+\tau)\beta A_m & 2(1+\tau) I_l \end{pmatrix}. \end{aligned} \tag{2.11}$$

Note that these inverses $(A_i^T A_i)^{-1}$ $(i = 1, \ldots, m)$ are just for algebraically showing the correction step explicitly, and they can be completely avoided (see Subsect. 3.3). We also set

$$H = QS^{-1}Q^T, \tag{2.12}$$

and

$$\bar{M} = \begin{pmatrix} \frac{2\tau+1}{1+\tau}I_l & -\frac{1}{1+\tau}I_l & \cdots & -\frac{1}{1+\tau}I_l & \frac{1}{(1+\tau)\beta}I_l \\ -\frac{1}{1+\tau}I_l & \frac{2\tau+1}{1+\tau}I_l & \cdots & -\frac{1}{1+\tau}I_l & \frac{1}{(1+\tau)\beta}I_l \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ -\frac{1}{1+\tau}I_l & -\frac{1}{1+\tau}I_l & \cdots & \frac{2\tau+1}{1+\tau}I_l & \frac{1}{(1+\tau)\beta}I_l \\ -\beta I_l & -\beta I_l & \cdots & -\beta I_l & 2I_l \end{pmatrix}. \tag{2.13}$$

3 Algorithm

Based on the framework of contraction method (see, e.g., in [3]), we present the novel algorithm (1.7)-(1.8) in this section.

3.1 A descent direction of the distance function induced by (1.7)

Recall the VI-structure of (1.7). The predictor $\tilde{w}^k$ generated by (1.7) satisfies

$$\theta(x) - \theta(\tilde{x}^k) + (w - \tilde{w}^k)^T F(\tilde{w}^k) \geq (w - \tilde{w}^k)^T Q(w^k - \tilde{w}^k), \quad \forall\, w \in \Omega. \tag{3.1}$$

Without loss of generality, we assume $w^k \neq \tilde{w}^k$ throughout our discussion. Otherwise, it follows from (2.4) that $\tilde{w}^k$ would be an optimal solution. Let w^* be an arbitrary solution of (2.4). Setting $w = w^*$ in (3.1), we have

$$\begin{aligned} (\tilde{w}^k - w^*)^T Q(w^k - \tilde{w}^k) &\geq \theta(\tilde{x}^k) - \theta(x^*) + (\tilde{w}^k - w^*)^T F(\tilde{w}^k) \\ &\overset{(2.5)}{=} \theta(\tilde{x}^k) - \theta(x^*) + (\tilde{w}^k - w^*)^T F(w^*) \geq 0. \end{aligned}$$

Using $\tilde{w}^k - w^* = (\tilde{w}^k - w^k) + (w^k - w^*)$ and $w^T Q w = \frac{1}{2} w^T (Q^T + Q) w$, the above inequality can be rewritten as

$$(w^k - w^*)^T Q(w^k - \tilde{w}^k) \geq (w^k - \tilde{w}^k)^T Q(w^k - \tilde{w}^k) \overset{(2.10)}{=} \frac{1}{2}\|w^k - \tilde{w}^k\|_S^2. \tag{3.2}$$

Taking the nonsingularity of the matrices Q and H (see (2.7) and (2.12)) into consideration, the inequality (3.2) can be further reformulated as

$$\begin{aligned} &\Big\langle \nabla\big(\frac{1}{2}\|w - w^*\|_H^2\big)\Big|_{w=w^k}, \ -H^{-1}Q(w^k - \tilde{w}^k)\Big\rangle \\ &\qquad = -(w^k - w^*)^T Q(w^k - \tilde{w}^k) \leq -\frac{1}{2}\|w^k - \tilde{w}^k\|_S^2. \end{aligned}$$

Therefore, for any fixed $w^* \in \Omega^*$,

$$d(w^k, \tilde{w}^k) := -H^{-1}Q(w^k - \tilde{w}^k) = -Q^{-T}S(w^k - \tilde{w}^k) = -M(w^k - \tilde{w}^k)$$

is a descent direction of the unknown distance function $\|w - w^*\|_H^2$ at the point w^k. According to the same analysis in [20], such a direction is able to yield the contraction of proximity to the solution set of (1.3) if an appropriate step size is taken. Hence, we generate the new iterate w^{k+1} via

$$w^{k+1} = w^k + \alpha_k d(w^k, \tilde{w}^k) = w^k - \alpha_k M(w^k - \tilde{w}^k). \tag{3.3}$$

It remains to find a step size α_k to make the update w^{k+1} closer to Ω^*.

3.2 A befitting step size α_k of the correction step (3.3)

We now turn to choose an appropriate step size α_k in the correction step (3.3). Since

$$\begin{aligned}
&\|w^{k+1} - w^*\|_H^2 \\
&\quad = \|w^k - \alpha Q^{-T} S(w^k - \tilde{w}^k) - w^*\|_H^2 \\
&\quad = \|w^k - w^*\|_H^2 - 2\alpha(w^k - w^*)^T Q(w^k - \tilde{w}^k) + \alpha^2 \|w^k - \tilde{w}^k\|_S^2 \\
&\quad \overset{(3.2)}{\leq} \|w^k - w^*\|_H^2 - \alpha\|w^k - \tilde{w}^k\|_S^2 + \alpha^2 \|w^k - \tilde{w}^k\|_S^2,
\end{aligned}$$

we have

$$\begin{aligned}
&\|w^k - w^*\|_H^2 - \|w^{k+1} - w^*\|_H^2 \\
&\quad \geq \alpha\|w^k - \tilde{w}^k\|_S^2 - \alpha^2\|w^k - \tilde{w}^k\|_S^2 =: q(\alpha).
\end{aligned}$$

Maximizing the quadratic term $q(\alpha)$, it follows that

$$\alpha_k^* = \frac{\|w^k - \tilde{w}^k\|_S^2}{2\|w^k - \tilde{w}^k\|_S^2} = \frac{1}{2}.$$

Meanwhile, note that $q(\alpha)$ is a lower bounded quadratic contraction function. Refer to the similar technique in [14], we can introduce a relaxation factor $\gamma \in (0, 2)$ and thus $\alpha_k := \gamma\alpha_k^* \in (0, 1)$, which means a fixed step size $\alpha_k \in (0, 1)$ can be taken in (3.3). Consequently, the correction step (3.3), as an update process, can be tackled easily.

3.3 The practical computing scheme for the new method (1.7)-(1.8)

Since the matrix M defined in (2.11) contains some inverses $(A_i^T A_i)^{-1}$ $(i = 1, \ldots, m)$, the corrector (3.3) is relatively complicated. Refer to the similar analysis in [22], these inverses are just for algebraically showing the correction step explicitly, and they can be completely avoided in computation empirically. More concretely, note that the variables $(A_1 x_1^k, \ldots, A_m x_m^k, \lambda^k)$ is essentially required in each iteration of (1.7). Using the same technique in, e.g., [19,22], the correction step (3.3) can be replaced with

$$\begin{pmatrix} A_1 x_1^{k+1} \\ \vdots \\ A_m x_m^{k+1} \\ \lambda^{k+1} \end{pmatrix} = \begin{pmatrix} A_1 x_1^k \\ \vdots \\ A_m x_m^k \\ \lambda^k \end{pmatrix} - \alpha \bar{M} \begin{pmatrix} A_1 x_1^k - A_1 \tilde{x}_1^k \\ \vdots \\ A_m x_m^k - A_m \tilde{x}_m^k \\ \lambda^k - \tilde{\lambda}^k \end{pmatrix} \tag{3.4}$$

where the matrix $\bar{M}$ is defined in (2.13), and it can be concretely written as

$$\begin{cases} A_i x_i^{k+1} = A_i x_i^k - \alpha\Big\{\dfrac{2\tau+1}{1+\tau}A_i(x_i^k - \tilde{x}_i^k) - \dfrac{1}{1+\tau}\big[\displaystyle\sum_{j\neq i} A_j(x_j^k - \tilde{x}_j^k) \\ \qquad\qquad\qquad\qquad - \dfrac{1}{\beta}(\lambda^k - \tilde{\lambda}^k)\big]\Big\}, \quad i = 1,\ldots,m, \\ \lambda^{k+1} = \lambda^k - \alpha\big\{-\beta\mathcal{A}(x^k - \tilde{x}^k) + 2(\lambda^k - \tilde{\lambda}^k)\big\}. \end{cases} \tag{3.5}$$

We now give a succinct representation of (3.5) for easy-implementation. Recall $\tilde{\lambda}^k = \lambda^k - \beta(\mathcal{A}x^k - b)$ in (1.7). By substituting it into (3.5), it follows that

$$\begin{aligned} & A_i x_i^{k+1} \\ &= A_i x_i^k - \alpha\Big\{\frac{2\tau+1}{1+\tau}A_i(x_i^k - \tilde{x}_i^k) - \frac{1}{1+\tau}\big[\sum_{j\neq i} A_j(x_j^k - \tilde{x}_j^k) - \frac{1}{\beta}(\lambda^k - \tilde{\lambda}^k)\big]\Big\} \\ &= A_i x_i^k - \alpha\Big\{\frac{2\tau+1}{1+\tau}A_i(x_i^k - \tilde{x}_i^k) - \frac{1}{1+\tau}\big[\sum_{j\neq i} A_j(x_j^k - \tilde{x}_j^k) - (\mathcal{A}x^k - b)\big]\Big\} \\ &= A_i x_i^k - \alpha\Big\{2A_i(x_i^k - \tilde{x}_i^k) + \frac{1}{1+\tau}\big(\mathcal{A}\tilde{x}^k - b\big)\Big\}, \end{aligned}$$

and

$$\begin{aligned} \lambda^{k+1} &= \lambda^k - \alpha\big\{-\beta\mathcal{A}(x^k - \tilde{x}^k) + 2(\lambda^k - \tilde{\lambda}^k)\big\} \\ &= \lambda^k - \alpha\big\{-\beta\mathcal{A}(x^k - \tilde{x}^k) + 2\beta(\mathcal{A}x^k - b)\big\} \\ &= \lambda^k - \alpha\beta\big\{\mathcal{A}x^k + \mathcal{A}\tilde{x}^k - 2b\big\}. \end{aligned}$$

Therefore, the practical correction step (3.5) can be reformulated concisely as

$$\begin{cases} \begin{cases} \text{for} \quad i = 1,\ldots,m, \quad \text{do:} \\ A_i x_i^{k+1} \;=\; A_i x_i^k - \alpha\big\{2A_i(x_i^k - \tilde{x}_i^k) + \frac{1}{1+\tau}(\mathcal{A}\tilde{x}^k - b)\big\}, \end{cases} \\ \quad \lambda^{k+1} \;=\; \lambda^k - \alpha\beta(\mathcal{A}x^k + \mathcal{A}\tilde{x}^k - 2b). \end{cases} \tag{3.6}$$

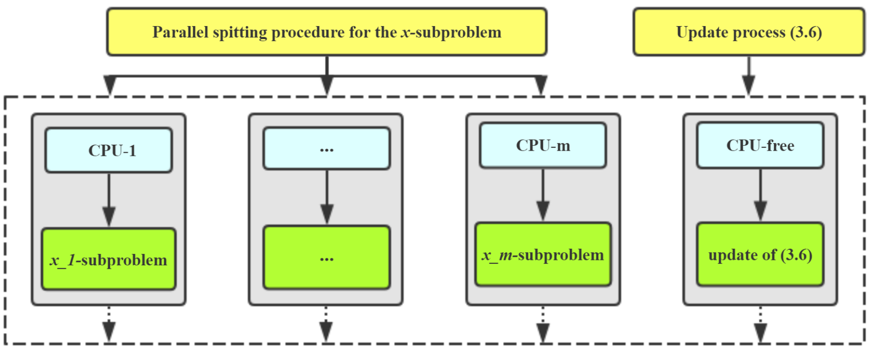

Fig. 1 An ideal parallel implementation mechanism of the proposed algorithm (3.7)

It further leads to the following more practical and succinct scheme to replace the proposed algorithm (1.7)-(1.8):

$$
\begin{cases}
\text{(Parallel)}\begin{cases}
\tilde{x}_1^k = \arg\min\big\{\mathcal{L}_\beta(x_1, x_2^k, \ldots, x_{m-1}^k, x_m^k, \lambda^k) \\
\qquad\qquad + \frac{\tau}{2}\beta\|A_1(x_1 - x_1^k)\|_2^2 \mid x_1 \in \mathcal{X}_1\big\}, \\
\tilde{x}_2^k = \arg\min\big\{\mathcal{L}_\beta(x_1^k, x_2, \ldots, x_{m-1}^k, x_m^k, \lambda^k) \\
\qquad\qquad + \frac{\tau}{2}\beta\|A_2(x_2 - x_2^k)\|_2^2 \mid x_2 \in \mathcal{X}_2\big\}, \\
\qquad \vdots \\
\tilde{x}_m^k = \arg\min\big\{\mathcal{L}_\beta(x_1^k, x_2^k, \ldots, x_{m-1}^k, x_m, \lambda^k) \\
\qquad\qquad + \frac{\tau}{2}\beta\|A_m(x_m - x_m^k)\|_2^2 \mid x_m \in \mathcal{X}_m\big\},
\end{cases} \\
\text{(Update)}\begin{cases}
\text{for} \quad i = 1, \ldots, m, \quad \text{do:} \\
A_i x_i^{k+1} = A_i x_i^k - \alpha\big\{2A_i(x_i^k - \tilde{x}_i^k) + \frac{1}{1+\tau}(\mathcal{A}\tilde{x}^k - b)\big\}, \\
\lambda^{k+1} = \lambda^k - \alpha\beta(\mathcal{A}x^k + \mathcal{A}\tilde{x}^k - 2b),
\end{cases}
\end{cases}
\tag{3.7}
$$

in which $\alpha \in (0,1)$ and $\tau > (m-4)/4$.

4 Convergence

We establish the convergence analysis of the proposed method (1.7)-(1.8) in this section. The technique of analysis is motivated by the work in [19, 22]. As we have mentioned, $\tau > (m-4)/4$ is required to guarantee the positive definiteness of the matrices S and H.

To show the global convergence of the novel algorithm (1.7)-(1.8), we first prove a lemma, followed by a theorem.

Lemma 3 *Let $\{\tilde{w}^k\}$ and $\{w^k\}$ be the sequences generated by the algorithm (1.7)-(1.8) for (1.3). Then, for any constant $\alpha \in (0,1)$, we have*

$$\|w^k - w^*\|_H^2 - \|w^{k+1} - w^*\|_H^2 \ \geq\ \alpha(1-\alpha)\|w^k - \tilde{w}^k\|_S^2. \tag{4.1}$$

Proof It follows from (3.3) that

$$
\begin{aligned}
&\|w^{k+1} - w^*\|_H^2 \\
&= \|w^k - \alpha Q^{-T}S(w^k - \tilde{w}^k) - w^*\|_H^2 \\
&= \|w^k - w^*\|_H^2 - 2\alpha(w^k - w^*)^T Q(w^k - \tilde{w}^k) + \alpha^2\|w^k - \tilde{w}^k\|_S^2 \\
&\overset{(3.2)}{\leq} \|w^k - w^*\|_H^2 - \alpha(1-\alpha)\|w^k - \tilde{w}^k\|_S^2,
\end{aligned}
$$

and the proof is complete. □

Theorem 1 *Let $\{\tilde{w}^k\}$ and $\{w^k\}$ be the sequences generated by the method (1.7)-(1.8) for solving (1.3). Then, for any $\alpha \in (0,1)$, we obtain $\lim_{k\to\infty} w^k = w^\infty$ where w^∞ belongs to Ω^*.*

Proof First of all, it follows from (4.1) that the generated sequence $\{w^k\}$ satisfies

$$\begin{aligned}\|w^{k+1}-w^*\|_H^2 &\le \|w^k-w^*\|_H^2 - \alpha(1-\alpha)\|w^k-\tilde{w}^k\|_S^2 \\ &\le \|w^k-w^*\|_H^2 \le \cdots \le \|w^0-w^*\|_H^2,\end{aligned}$$

which means the sequence $\{w^k\}$ is bounded. Furthermore, summing (4.1) over $k=0,1,\ldots,\infty$, we obtain

$$\begin{aligned}\sum_{k=0}^{\infty}\alpha(1-\alpha)\|w^k-\tilde{w}^k\|_S^2 &\le \sum_{k=0}^{\infty}(\|w^k-w^*\|_H^2 - \|w^{k+1}-w^*\|_H^2) \\ &\le \|w^0-w^*\|_H^2,\end{aligned}$$

and thus

$$\lim_{k\to\infty}\|w^k-\tilde{w}^k\|_S^2 = 0. \tag{4.2}$$

This indicates the sequence $\{\tilde{w}^k\}$ is also bounded. Let w^∞ be a cluster point of $\{\tilde{w}^k\}$, where $\{\tilde{w}^{k_j}\}$ is a subsequence which converges to w^∞. Then, it follows from (2.6) that

$$\theta(x)-\theta(\tilde{x}^{k_j}) + (w-\tilde{w}^{k_j})^T F(\tilde{w}^{k_j}) \ge (w-\tilde{w}^{k_j})^T Q(w^{k_j}-\tilde{w}^{k_j}), \quad \forall\, w\in\Omega.$$

Note the matrix Q is not singular. It follows from the continuity of $\theta(x)$ and $F(w)$ that

$$w^\infty\in\Omega, \quad \theta(x)-\theta(x^\infty)+(w-w^\infty)^T F(w^\infty)\ge 0, \quad \forall\, w\in\Omega,$$

which means w^∞ is a solution of the $\mathrm{VI}(\Omega,F,\theta)$ (2.4). On the one hand, according to (4.2), we have $\lim_{k\to\infty} w^{k_j} = w^\infty$. On the other hand, it follows from (4.1) that

$$\|w^{k+1}-w^\infty\|_H^2 \le \|w^k-w^\infty\|_H^2, \tag{4.3}$$

which means it is impossible that the sequence $\{w^k\}$ has more than one cluster point. Therefore, we have $\lim_{k\to\infty} w^k = w^\infty$ and the proof is complete. □

Remark 1 Refer to the same techniques in, e.g., [19,22], we can easily show a worst-case $\mathcal{O}(1/N)$ convergence rate measured by iteration complexity for the novel method (1.7)-(1.8) by using the fundamental Lemma 2. Since the techniques are completely similar, the analysis of the convergence rate is omitted here.

5 Numerical experiments

We report the numerical results of the proposed algorithm (1.7)-(1.8) (practical scheme is (3.7)) in this section. All codes are written in a Python 3.9 and implemented in a laptop with Intel Core CPU 2.20 GHz (i7-8750H) and 16 GB memory. The preliminary numerical results affirmatively demonstrate that the proposed method can perform more efficiently than some known parallel spitting algorithms.

5.1 The test model

The latent variable Gaussian graphical model selection problem (abbreviated as LVGGMS) in [6] is a typical three-blocks separable convex minimization model arising in statistical learning. As stated in [1], the mathematical form of the LVGGMS reads as

$$\begin{array}{ll} \min & F(X,Y,Z) := \langle X, C\rangle - \log\det(X) + \nu\|Y\|_1 + \mu \mathrm{tr}(Z) \\ \text{s.t.} & X - Y + Z = 0, \;\; Z \succeq 0, \end{array} \tag{5.1}$$

where $C \in \Re^{n\times n}$ is the covariance matrix obtained from the observation, ν and μ stand for two given positive weight parameters, $\|Y\|_1 = \sum_{i,j}|Y_{i,j}|$ and $\mathrm{tr}(\cdot)$ represents the trace of a matrix.

We first show the implementation of the proposed scheme (3.7) for LVGGMS. Since the update (3.6) is simple, we only focus on the detail of the resulting x_i-subproblems ($i = 1,2,3$). Let

$$\begin{aligned} \mathcal{L}_\beta(X,Y,Z,\varLambda) =& \langle X, C\rangle - \log\det(X) + \nu\|Y\|_1 + \mu\mathrm{tr}(Z) \\ & - \langle \varLambda, X - Y + Z\rangle + \frac{\beta}{2}\|X - Y + Z\|_F^2 \end{aligned}$$

be the augmented Lagrangian function of (5.1). The corresponding scheme of the x-subproblem in (3.7) then reads as

$$\begin{cases} \tilde{X}^k = \arg\min\big\{\mathcal{L}_\beta(X,Y^k,Z^k,\varLambda^k) + \frac{\tau\beta}{2}\|X - X^k\|_F^2 \;\big|\; X \in \Re^{n\times n}\big\}, \\ \tilde{Y}^k = \arg\min\big\{\mathcal{L}_\beta(X^k,Y,Z^k,\varLambda^k) + \frac{\tau\beta}{2}\|Y - Y^k\|_F^2 \;\big|\; Y \in \Re^{n\times n}\big\}, \\ \tilde{Z}^k = \arg\min\big\{\mathcal{L}_\beta(X^k,Y^k,Z,\varLambda^k) + \frac{\tau\beta}{2}\|Z - Z^k\|_2^2 \;\big|\; Z \succeq 0,\, Z \in \Re^{n\times n}\big\}. \end{cases} \tag{5.2}$$

For the X-subproblem in (5.2), according to the first-order optimal condition, it is equivalent to solve the nonlinear equation system

$$C - X^{-1} + \beta\big(X - Y^k + Z^k - \frac{1}{\beta}\varLambda^k\big) + \tau\beta(X - X^k) = 0.$$

Multiplying X to both sides, it follows that

$$(1+\tau)\beta X^2 + (C - \tau\beta X^k - \beta Y^k + \beta Z^k - \varLambda^k)X - I = 0. \tag{5.3}$$

Let $UDU^T = C - \tau\beta X^k - \beta Y^k + \beta Z^k - \varLambda^k$ be an eigenvalue decomposition. Plugging it back into (5.3) and setting $P = U^T X U$, we have

$$(1+\tau)\beta PP + DP - I = 0,$$

and thus obtain

$$P_{ii} = \frac{1}{2\beta(1+\tau)}\Big(- D_{ii} + \sqrt{D_{ii}^2 + 4(1+\tau)\beta}\,\Big).$$

Therefore, we can derive that $\tilde{X}^k = U\,\mathrm{diag}(P)\,U^T$ is a solution of (5.3). For the Y-subproblem in (5.2), since

$$\begin{aligned}
\tilde{Y}^k &= \arg\min_{Y} \big\{\nu\|Y\|_1 + \frac{\beta}{2}\|X^k - Y + Z^k - \frac{1}{\beta}\Lambda^k\|_F^2 + \frac{\tau}{2}\beta\|Y - Y^k\|_F^2\big\} \\
&= \arg\min_{Y} \big\{\|Y\|_1 + \frac{\beta(1+\tau)}{2\nu}\big\|Y - \frac{1}{1+\tau}(X^k + \tau Y^k + Z^k - \frac{1}{\beta}\Lambda^k)\big\|_F^2\big\},
\end{aligned}$$

the Y-subproblem can be solved via the soft shrinkage operator (see [31]) immediately. Finally, for the Z-subproblem in (5.2), note that

$$\begin{aligned}
\tilde{Z}^k &= \arg\min_{Z\succeq 0} \big\{\mu\mathrm{tr}(Z) + \frac{\beta}{2}\big\|X^k - Y^k + Z - \frac{1}{\beta}\Lambda^k\big\|_F^2 + \frac{\tau}{2}\beta\|Z - Z^k\|_2^2\big\} \\
&= \arg\min_{Z\succeq 0} \big\{\big\|Z - \frac{1}{1+\tau}\big(-X^k + Y^k + \tau Z^k + \frac{1}{\beta}(\Lambda^k - \mu I)\big)\big\|_2^2\big\}.
\end{aligned}$$

Let $VD_1V^T = \frac{1}{1+\tau}\big(-X^k + Y^k + \tau Z^k + \frac{1}{\beta}\Lambda^k - \frac{\mu}{\beta}I\big)$ be an eigenvalue decomposition. Then, we can easily verify that $\tilde{Z}^k = V\max(D_1, 0)\,V^T$ is a solution of the Z-subproblem (where $\max(D_1, 0)$ is taken component-wise).

5.2 Numerical results

We evaluate the numerical performance of the proposed method (3.7) in this subsection. Some parallel splitting algorithms including the parallel augmented Lagrange method (denoted by P-ALM) proposed in [14] and the splitting ADMM-like method (denoted by PS-ADMM) proposed in [18] are compared. In addition, as a reference, the numerical results of the direct extension of ADMM (denoted by D-ADMM) is also listed.

In our experiments, we take $\nu = 0.005$ and $\mu = 0.05$ (refer to [1]), and the covariance matrix C is randomly generated according to S. Boyd's Homepage (see http://web.stanford.edu/~boyd/papers/admm/covsel/covsel_example.html). Moreover, as stated in [1], the following three stop stopping conditions are taken:

$$\begin{aligned}
\mathrm{IER}(k) &:= \max\big\{\|X^k - X^{k+1}\|_\infty, \|Y^k - Y^{k+1}\|_\infty, \|Z^k - Z^{k+1}\|_\infty\big\} < \mathrm{Tol}, \\
\mathrm{OER}(k) &:= \tfrac{F(X^k,Y^k,Z^k) - F^*}{F^*} < \mathrm{Tol}, \\
\mathrm{CER}(k) &:= \|X^k - Y^k + Z^k\|_F < \mathrm{Tol},
\end{aligned}$$

where F^* denotes the approximate optimal objective function value obtained by running the GS-ADMM after 1,000 iterations. When the proposed method (3.7) is applied for solving (5.1), it is convergent provided $\tau > (3-4)/4$, and we consider three special cases: $\tau = 0$, $\tau = 1/3$ and $\tau = 1$. Moreover, the initial values are chosen as $(X_0, Y_0, Z_0, \Lambda_0) = (I_n, 2I_n, I_n, \mathbf{0}_{n\times n})$ in our experiments. To report the numerical results, "It." and "CPU" in the following tables stand for the iteration numbers and computing time in seconds, respectively.

Table 1 Numerical results of the novel method (3.7) for LVGGMS with $\mathrm{IER}(k) < 10^{-9}$

τ	β	It.	CPU(s)	IER	OER	CER
	0.05	344	9.75	9.7460e-10	1.0311e-11	7.5727e-8
	0.08	232	6.69	9.6085e-10	4.8423e-12	4.5428e-8
	0.09	210	6.47	9.9953e-10	3.3951e-12	4.1487e-8
	0.10	193	5.61	9.6146e-10	2.4104e-12	3.5610e-8
	0.11	178	5.26	9.5083e-10	1.8495e-12	3.1826e-8
	0.12	165	4.85	9.6006e-10	1.2734e-12	2.9308e-8
	0.13	154	4.66	9.4603e-10	1.4926e-12	2.6521e-8
$\tau = 0$	0.14	144	4.15	9.6551e-10	4.0919e-13	2.5027e-8
	0.15	135	3.91	9.9361e-10	9.7852e-12	2.3969e-8
	0.16	128	3.85	9.2450e-10	2.2960e-11	2.0924e-8
	0.17	121	3.52	9.4837e-10	3.7283e-11	2.0286e-8
	0.18	117	3.45	7.7855e-10	5.5935e-11	1.5566e-8
	0.19	113	3.18	6.8999e-10	9.4191e-11	1.2886e-8
	0.20	109	3.13	7.9165e-10	9.6042e-11	1.2085e-8
	0.25	150	4.39	9.6473e-10	9.6111e-11	3.4638e-9
	0.05	206	5.92	9.8697e-10	7.7880e-12	1.4254e-8
	0.08	140	4.14	9.3143e-10	2.3608e-12	8.0790e-9
	0.09	126	3.72	9.7631e-10	2.3915e-12	7.4708e-9
	0.10	115	3.42	9.5894e-10	3.4595e-12	6.5608e-9
	0.11	106	3.08	9.4971e-10	2.2397e-12	5.8720e-9
	0.12	98	2.80	9.6024e-10	5.8159e-12	5.3776e-9
	0.13	92	2.65	9.9286e-10	2.8222e-11	4.7183e-9
$\tau = \frac{1}{3}$	0.14	97	2.81	9.9905e-10	5.5752e-11	1.9197e-9
	0.15	114	3.35	9.6421e-10	6.9626e-11	9.4530e-10
	0.16	128	3.62	8.8635e-10	7.4129e-11	7.2717e-10
	0.17	140	4.20	9.0621e-10	8.5593e-11	6.6003e-10
	0.18	151	4.49	9.5548e-10	1.0016e-10	6.3327e-10
	0.19	162	4.74	9.5650e-10	1.0979e-10	5.8506e-10
	0.20	173	4.90	9.2823e-10	1.1551e-10	5.2899e-10
	0.25	223	6.52	9.6228e-10	1.6419e-10	4.1703e-10
	0.05	156	4.43	9.3712e-10	3.3488e-12	1.6634e-8
	0.08	103	3.04	9.1796e-10	3.0028e-12	1.0454e-8
	0.09	94	2.67	8.9077e-10	3.8441e-11	8.4775e-9
	0.10	111	3.20	9.8091e-10	6.9759e-11	2.2004e-9
	0.11	132	3.92	9.1714e-10	8.1184e-11	1.5264e-9
	0.12	150	4.36	9.0231e-10	9.4049e-11	1.2776e-9
	0.13	166	4.79	9.6545e-10	1.1519e-10	1.2104e-9
$\tau = 1$	0.14	182	5.24	9.5517e-10	1.2772e-10	1.0825e-9
	0.15	197	5.81	9.8515e-10	1.4569e-10	1.0230e-9
	0.16	212	6.19	9.8362e-10	1.5894e-10	9.4454e-10
	0.17	227	6.66	9.6333e-10	1.6857e-10	8.6156e-10
	0.18	241	7.26	9.9416e-10	1.8737e-10	8.3316e-10
	0.19	256	7.78	9.4939e-10	1.9123e-10	7.4871e-10
	0.20	270	7.83	9.5726e-10	2.0534e-10	7.1357e-10
	0.25	338	9.86	9.9542e-10	2.7775e-10	5.8496e-10

Table 2 Numerical results of the new method (3.7) for LVGGMS with OER(k) $< 10^{-9}$

τ	β	It.	CPU(s)	OER	IER	CER
$\tau = 0$	0.05	144	4.21	8.7198e-10	5.5963e-7	6.6726e-5
	0.08	113	3.18	2.1037e-10	3.2138e-7	1.8706e-5
	0.09	109	3.24	4.5469e-10	2.0799e-7	9.5445e-6
	0.10	109	3.07	9.8421e-10	1.6405e-7	6.5112e-6
	0.11	78	2.24	7.9163e-10	3.2814e-6	3.3773e-5
	0.12	98	2.77	3.3436e-10	1.3798e-7	4.3306e-6
	0.13	102	2.92	5.7360e-10	6.2578e-8	1.7714e-6
	0.14	96	2.65	8.3647e-10	6.3240e-8	1.6784e-6
	0.15	89	2.55	7.1812e-10	1.4656e-7	1.8801e-6
	0.16	90	2.57	8.3937e-10	1.7300e-7	9.9504e-7
	0.17	90	2.52	8.5395e-10	1.2672e-7	6.0071e-7
	0.18	94	2.74	7.7539e-10	4.1692e-8	2.3006e-7
	0.19	91	2.52	5.7388e-10	5.7868e-8	2.0498e-7
	0.20	79	2.24	6.2931e-10	2.0364e-7	7.0520e-7
	0.25	130	3.72	8.9520e-10	8.5439e-9	3.0995e-8
$\tau = \frac{1}{3}$	0.05	93	2.60	8.9340e-10	4.9865e-7	8.5843e-6
	0.08	77	2.22	4.6772e-10	2.1281e-7	1.9835e-6
	0.09	72	2.04	7.6146e-10	1.9433e-7	1.5404e-6
	0.10	73	2.07	4.1285e-10	9.6933e-8	6.7221e-7
	0.11	70	1.98	3.4314e-10	1.1518e-7	4.8610e-7
	0.12	67	1.97	4.6752e-10	8.4018e-8	3.7552e-7
	0.13	65	1.81	1.5487e-10	1.1548e-7	3.0907e-7
	0.14	82	2.39	9.3108e-10	1.5173e-8	2.5732e-8
	0.15	97	2.84	8.9237e-10	1.1692e-8	1.1546e-8
	0.16	109	3.18	9.2070e-10	1.0411e-8	8.6309e-9
	0.17	120	3.53	9.4610e-10	9.4787e-9	6.9899e-9
	0.18	131	3.74	9.1105e-10	8.2480e-9	5.5382e-9
	0.19	141	4.08	9.4327e-10	7.7995e-9	4.8375e-9
	0.20	151	4.35	9.4841e-10	7.2345e-9	4.1833e-9
	0.25	199	5.76	9.4629e-10	5.2964e-9	2.3286e-9
$\tau = 1$	0.05	79	2.19	4.4095e-10	3.5054e-7	6.8296e-6
	0.08	55	1.57	2.4838e-11	3.0057e-6	5.8229e-6
	0.09	59	1.67	2.7272e-10	6.4738e-7	2.0767e-6
	0.10	94	2.63	9.7135e-10	1.2826e-8	2.9113e-8
	0.11	113	3.22	9.2463e-10	9.8931e-9	1.6637e-8
	0.12	129	3.66	9.8080e-10	8.9059e-9	1.2776e-8
	0.13	145	4.25	9.3613e-10	7.4561e-9	9.4578e-9
	0.14	160	4.78	9.3631e-10	6.6617e-9	7.6573e-9
	0.15	174	5.20	9.8511e-10	6.3426e-9	6.6831e-9
	0.16	189	5.41	9.3168e-10	5.5066e-9	5.3626e-9
	0.17	203	6.08	9.4039e-10	5.1362e-9	4.6589e-9
	0.18	217	6.39	9.3748e-10	4.7653e-9	4.0484e-9
	0.19	230	6.98	9.8712e-10	4.6891e-9	3.7509e-9
	0.20	244	7.39	9.6792e-10	4.3266e-9	3.2695e-9
	0.25	311	9.50	9.8476e-10	3.4069e-9	2.0259e-9

Table 3 Numerical results of the novel method (3.7) for LVGGMS with IER(k) $< 10^{-9}$

τ	β	It.	CPU(s)	CER	IER	OER
	0.05	487	14.22	9.9285e-10	1.2929e-11	1.4414e-13
	0.08	311	9.17	9.8582e-10	2.0804e-11	1.1631e-13
	0.09	279	8.46	9.5679e-10	2.2917e-11	9.4975e-14
	0.10	252	7.56	9.8231e-10	2.6334e-11	8.4516e-14
	0.11	230	6.66	9.6789e-10	2.8712e-11	7.4265e-14
	0.12	211	6.29	9.9267e-10	3.2306e-11	7.4265e-14
	0.13	195	5.76	9.9202e-10	3.5177e-11	7.8449e-14
$\tau = 0$	0.14	182	5.30	9.2271e-10	3.5426e-11	8.2633e-14
	0.15	169	5.02	9.8864e-10	4.0898e-11	6.5897e-14
	0.16	159	4.76	9.1457e-10	4.0539e-11	3.5940e-13
	0.17	149	4.47	9.7014e-10	4.6083e-11	1.5732e-13
	0.18	141	4.16	9.3460e-10	4.7360e-11	9.8867e-13
	0.19	134	3.94	8.7264e-10	6.1258e-11	3.2007e-13
	0.20	127	3.66	9.3805e-10	1.1921e-10	3.3415e-12
	0.25	162	4.66	9.3297e-10	2.6086e-10	2.5317e-11
	0.05	257	7.80	9.5358e-10	6.6164e-11	5.3847e-13
	0.08	165	5.00	9.4608e-10	1.0857e-10	3.4308e-13
	0.09	147	4.59	9.6820e-10	1.2606e-10	3.7049e-13
	0.10	133	3.83	9.1763e-10	1.3381e-10	4.3325e-13
	0.11	121	3.51	9.4235e-10	1.5232e-10	7.4286e-13
	0.12	111	3.14	9.1923e-10	1.6399e-10	1.4037e-12
	0.13	103	2.99	9.7409e-10	1.8570e-10	5.5575e-12
$\tau = \frac{1}{3}$	0.14	101	3.10	9.6060e-10	4.8906e-10	2.7459e-11
	0.15	114	3.50	9.4530e-10	9.6421e-10	6.9626e-11
	0.16	126	3.66	9.4308e-10	1.1486e-9	9.6578e-11
	0.17	137	4.04	9.3989e-10	1.2886e-9	1.2263e-10
	0.18	147	4.17	9.7658e-10	1.4703e-9	1.5563e-10
	0.19	157	4.63	9.6689e-10	1.5763e-9	1.8303e-10
	0.20	167	4.97	9.2920e-10	1.6249e-9	2.0489e-10
	0.25	211	6.16	9.8489e-10	2.2575e-9	3.9376e-10
	0.05	193	5.44	9.8963e-10	5.5599e-11	2.2489e-13
	0.08	122	3.46	9.0785e-10	7.9808e-11	5.2320e-13
	0.09	108	3.06	9.9591e-10	1.0585e-10	3.7273e-12
	0.10	117	3.47	8.8647e-10	3.9514e-10	2.7586e-11
	0.11	136	4.01	9.2408e-10	5.5605e-10	4.8723e-11
	0.12	153	4.68	9.1998e-10	6.5068e-10	6.7349e-11
	0.13	168	4.87	9.9534e-10	7.9470e-10	9.4401e-11
$\tau = 1$	0.14	183	5.29	9.9053e-10	8.7447e-10	1.1669e-10
	0.15	198	5.71	9.4294e-10	9.0854e-10	1.3410e-10
	0.16	212	6.39	9.4454e-10	9.8362e-10	1.5894e-10
	0.17	225	7.02	9.9152e-10	1.1075e-9	1.9448e-10
	0.18	239	7.14	9.5035e-10	1.1328e-9	2.1422e-10
	0.19	252	7.74	9.5911e-10	1.2138e-9	2.4605e-10
	0.20	265	7.44	9.5597e-10	1.2794e-9	2.7654e-10
	0.25	327	9.91	9.7005e-10	1.6432e-9	4.6490e-10

Table 4 Toned values of parameters of the above mentioned operator spiting methods for LVGGMS

Method	Other parameters	β		
		IER	OER	CER
New method	regularization factor $\tau = 1/3$	$\beta = 0.13$	$\beta = 0.13$	$\beta = 0.14$
D-ADMM	—	$\beta = 0.15$	$\beta = 0.14$	$\beta = 0.15$
PS-ADMM	regularization factor $\tau = 1.001$	$\beta = 0.07$	$\beta = 0.07$	$\beta = 0.08$
P-ALM	relaxation factor $\gamma = 1.95$	$\beta = 0.10$	$\beta = 0.09$	$\beta = 0.11$

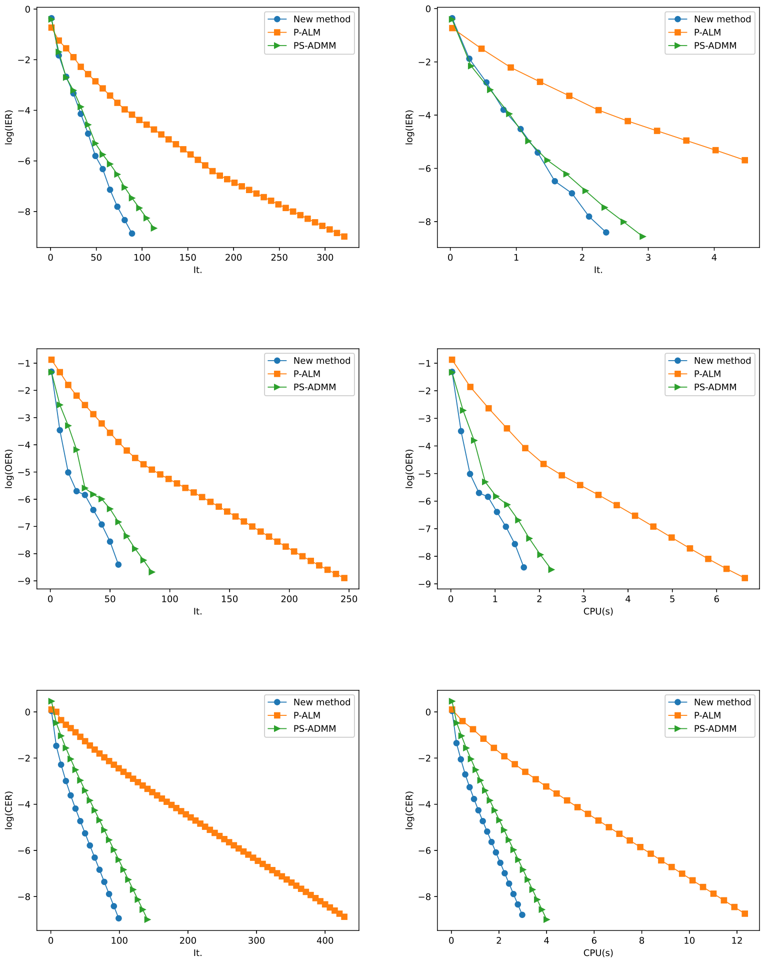

Fig. 2 First column: convergence curves of the IER, OER, CER with iteration numbers; second column: convergence curves of the IER, OER, CER with CPU times

Table 5 Numerical results of LVGGMS

Methods	It.	CPU(s)	IER	OER	CER
$\mathbf{IER}(k) < 10^{-9}$					
D-ADMM	67	1.86	9.9590e-10	3.8455e-11	1.6910e-8
P-ALM	323	9.37	9.8331e-10	1.0771e-10	1.0001e-7
D-ALM	div	—	—	—	—
PS-ADMM	121	3.18	9.7294e-10	5.8391e-12	3.1763e-8
New method	92	2.65	9.9286e-10	2.8222e-11	4.7183e-9
$\mathbf{OER}(k) < 10^{-9}$					
D-ADMM	37	1.02	1.1205e-6	3.1529e-10	2.3095e-5
P-ALM	252	7.24	2.8307e-8	9.9871e-10	3.0854e-6
D-ALM	div	—	—	—	—
PS-ADMM	91	2.54	3.0337e-8	9.5242e-10	1.0527e-6
New method	65	1.81	1.1548e-7	1.5487e-10	3.0907e-7
$\mathbf{CER}(k) < 10^{-9}$					
D-ADMM	79	2.20	4.6506e-11	3.9099e-13	8.2403e-10
P-ALM	436	12.77	2.0315e-8	5.4454e-12	9.7584e-10
D-ALM	div	—	—	—	—
PS-ADMM	143	4.06	3.6798e-11	1.2920e-12	8.7526e-10
New method	101	3.10	4.8906e-10	2.7459e-11	9.6060e-10

In Tables 1-3, we report the numerical results of the proposed method (3.7) for LVGGMS when various stop stopping conditions are adopted. As can be seen easily, both the iteration numbers and the CPU times of the new algorithm have a similar changing pattern with the change of β for the various regularization parameters τ. In particular, the cases where $\tau = 1/3$ and $\beta \in (0, 12, 0.14)$ perform more stably and efficiently, and they need only a reduced required iterations and CPU times in terms of the feasibility errors IER, OER and CER.

To show the comparisons between the novel method (3.7) and other operator spitting methods, we list first some efficient parameters for the above mentioned splitting algorithms. Note that the D-ADMM enjoys a Gaussian decomposition structure but not a Jacobian decomposition. The D-ADMM is thus not a parallel algorithm. Using the same parameter adjustment strategy as the novel method (3.7), the toned parameters of the mentioned methods for solving LVGGMS are given in Table 4.

In Table 5, we report the numerical performance of these five algorithms for LVGGMS. As shown in Table 5, we can find that the direct extension of ADMM performs more efficiently than other algorithms, even though its convergence is unclear; and the proposed method (3.7) performs competitively with the D-ADMM. Moreover, compared with the parallel algorithms P-ALM and PS-ADMM, the novel method need fewer required iterations and CPU times to reach different stopping criteria, which verifies empirically our theoretical motivation of the proposed algorithm (our purpose is to propose a more efficient parallel spitting method). In addition, from Table 5, we can easily find that the D-ALM is divergent for solving LVGGMS, which can be further illustrated that the direct extension of ALM is not necessarily convergent.

To further visualize the numerical comparison between the new method and the mentioned parallel splitting algorithms, in Fig. 2, we plot their respective feasibility errors with respect to iterations and CPU times. Computational results in Fig. 2 demonstrate that the proposed method has steeper convergence curves both in iterations and computing times under the various stopping criteria.

6 Conclusions

We propose a novel parallel splitting ALM-based method for the separable convex programming problem with linear constraints, where the objective function can be expressed as the sum of some individual subfunctions without coupled variables. By adding a simple update (1.8) to the fully parallel regularized ALM (1.7), convergence of the novel method can be guaranteed provided a reduced parameter of the regularization terms, which potentially results in fewer required iterations in the real computations. The numerical performance on LVGGMS affirmatively illustrates that the proposed method has an acceleration effect compared with some known parallel splitting algorithms.

Acknowledgements Funding was provided by National Natural Science Foundation of China (Grant No.11871029).

References

1. Bai, J., Li, J., Xu, F., Zhang, H.: Generalized symmetric ADMM for separable convex optimization. Computational Optimization and Applications **70**(1), 129–170 (2018). DOI 10.1007/s10589-017-9971-0
2. Beck, A.: First-order Methods in Optimization. SIAM, Philadelphia (2017)
3. Blum, E., Oettli, W.: Mathematische Optimierung. Econometrics Oper. Rese. 20, Springer-Verlag, Berlin (1975)
4. Bose, N., Boo, K.: High-resolution image reconstruction with multisensors. International Journal of Imaging Systems and Technology **9**(4), 294–304 (1998). DOI 10.1002/(SICI)1098-1098(1998)9:4⟨294::AID-IMA11⟩3.0.CO;2-X
5. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends Mach. Learn. **3**(1), 1–122 (2010). DOI 10.1561/2200000016
6. Chandrasekaran, V., Parrilo, P.A., Willsky, A.S.: Latent variable graphical model selection viaconvex optimization. The Annals of Statistics **40**(4), 1935–1967 (2012). DOI 10.1214/11-AOS949
7. Chen, C., He, B., Ye, Y., Yuan, X.: The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent. Mathematical Programming **155**(1-2), 57–79 (2016). DOI 10.1007/s10107-014-0826-5
8. Facchinei, F., Pang, J.S.: Finite-dimensional variational inequalities and complementarity problems. Vol. I. Springer Ser. Oper. Res., Springer-Verlag, New York (2003)
9. Gabay, D., Mercier, B.: A dual algorithm for the solution of nonlinear variational problems via finite element approximation. Computers & mathematics with applications **2**(1), 17–40 (1976). DOI 10.1016/0898-1221(76)90003-1

10. Glowinski, R.: Numerical Methods for Nonlinear Variational Problems. Springer-Verlag, New York, Berlin, Heidelberg, Tokyo (1984)
11. Hager, W.W., Zhang, H.: Convergence rates for an inexact ADMM applied to separable convex optimization. Computational Optimization and Applications **77**(3), 729–754 (2020). DOI 10.1007/s10589-017-9971-0
12. He, B.: My 20 years research on alternating directions method of multipliers. Oper. Res. Trans. **22**, 1–31 (2018)
13. He, B.: Study on the splitting methods for separable convex optimization in a unified algorithmic framework. Anal. Theory Appl. **36**, 262–282 (2020). DOI 10.4208/ata.OA-SU13
14. He, B., Hou, L., Yuan, X.: On full Jacobian decomposition of the augmented Lagrangian method for separable convex programming. SIAM Journal on Optimization **25**(4), 2274–2312 (2015). DOI 10.1137/130922793
15. He, B., Ma, F., Yuan, X.: Optimal proximal augmented Lagrangian method and its application to full Jacobian splitting for multi-block separable convex minimization problems. IMA Journal of Numerical Analysis **40**(2), 1188–1216 (2020). DOI 10.1093/imanum/dry092
16. He, B., Ma, F., Yuan, X.: Optimally linearizing the alternating direction method of multipliers for convex programming. Computational Optimization and Applications **75**(2), 361–388 (2020). DOI 10.1007/s10589-019-00152-3
17. He, B., Tao, M., Yuan, X.: Alternating direction method with Gaussian back substitution for separable convex programming. SIAM Journal on Optimization **22**(2), 313–340 (2012). DOI 10.1137/110822347
18. He, B., Tao, M., Yuan, X.: A splitting method for separable convex programming. IMA Journal of Numerical Analysis **35**(1), 394–426 (2015). DOI 10.1093/imanum/drt060
19. He, B., Tao, M., Yuan, X.: Convergence rate analysis for the alternating direction method of multipliers with a substitution procedure for separable convex programming. Mathematics of Operations Research **42**(3), 662–691 (2017). DOI 10.1287/moor.2016.0822
20. He, B., Yuan, X.: Convergence analysis of primal-dual algorithms for a saddle-point problem: from contraction perspective. SIAM Journal on Imaging Sciences **5**(1), 119–149 (2012). DOI 10.1137/100814494
21. He, B., Yuan, X.: On the $O(1/n)$ convergence rate of the Douglas-Rachford alternating direction method. SIAM Journal on Numerical Analysis **50**(2), 700–709 (2012). DOI 10.1137/110836936
22. He, B., Yuan, X.: A class of ADMM-based algorithms for three-block separable convex programming. Computational Optimization and Applications **70**(3), 791–826 (2018). DOI 10.1007/s10589-018-9994-1
23. Hestenes, M.R.: Multiplier and gradient methods. Journal of Optimization Theory and Applications **4**(5), 303–320 (1969). DOI 10.1007/BF00927673
24. Kiwiel, K.C., Rosa, C.H., Ruszczynski, A.: Proximal decomposition via alternating linearization. SIAM Journal on Optimization **9**(3), 668–689 (1999). DOI 10.1137/S1052623495288064
25. Martinet, B.: Régularisation d'inéquations variationnelles par approximations successives. Rev. Fr. Inform. Rech. Oper. **4**(R3), 154–158 (1970)
26. Parikh, N., Boyd, S.: Proximal algorithms. Foundations and Trends in optimization **1**(3), 127–239 (2014). DOI 10.1561/2400000003
27. Peng, Y., Ganesh, A., Wright, J., Xu, W., Ma, Y.: Rasl: Robust alignment by sparse and low-rank decomposition for linearly correlated images. IEEE transactions on pattern analysis and machine intelligence **34**(11), 2233–2246 (2012)
28. Powell, M.J.: A method for nonlinear constraints in minimization problems. In: Optimization, R. Fletcher, ed., pp. 283–298. Academic Press, New York (1969)
29. Rockafellar, R.T.: Monotone operators and the proximal point algorithm. SIAM Journal on Control and Optimization **14**(5), 877–898 (1976). DOI 10.1137/0314056
30. Setzer, S., Steidl, G., Teuber, T.: Deblurring Poissonian images by split Bregman techniques. Journal of Visual Communication and Image Representation **21**(3), 193–199 (2010). DOI 10.1016/j.jvcir.2009.10.006

31. Tao, M., Yuan, X.: Recovering low-rank and sparse components of matrices from incomplete and noisy observations. SIAM Journal on Optimization **21**(1), 57–81 (2011). DOI 10.1137/100781894
32. Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., Knight, K.: Sparsity and smoothness via the fused lasso. Journal of the Royal Statistical Society: Series B (Statistical Methodology) **67**(1), 91–108 (2005). DOI 10.1111/j.1467-9868.2005.00490.x
33. Xu, S., Yuan, J., He, B.: Proximal augmented Lagrangian method for convex optimization with linear inequality constraints. Optimization Online (2019)
34. Yuan, J., Bae, E., Tai, X.C.: A study on continuous max-flow and min-cut approaches. In: 2010 ieee computer society conference on computer vision and pattern recognition, pp. 2217–2224. IEEE (2010)
35. Yuan, J., Bae, E., Tai, X.C., Boykov, Y.: A continuous max-flow approach to *potts* model. In: European conference on computer vision, pp. 379–392. Springer (2010)
36. Yuan, J., Fenster, A.: Modern convex optimization to medical image analysis. arXiv preprint arXiv:1809.08734 (2018)