A PARALLEL GAUSS-SEIDEL METHOD FOR CONVEX PROBLEMS WITH SEPARABLE STRUCTURE

XIN YANG, NAN WANG AND LINGLING Xu*

Jiangsu Provincial Key Laboratory for NSLSCS
School of Mathematical Sciences
Nanjing Normal University, Nanjing, China

ABSTRACT. The convergence of direct ADMM is not guaranteed when used to solve multi-block separable convex optimization problems. In this paper, we propose a Gauss-Seidel method which can be calculated in parallel while solving subproblems. First we divide the variables into different groups. In the inner group, we use Gauss-Seidel method solving the subproblem. Among the different groups, Jacobi-like method is used. The effectiveness of the algorithm is proved by some numerical experiments.

1. Introduction. In this paper, we consider a separable convex optimization problem with linear constraints:

$$
\min \left\{ \sum_{i=1}^{m} \theta_i(x_i) \mid \sum_{i=1}^{m} A_i x_i = b, x_i \in X_i \right\},
$$

where $\theta_i : \mathcal{R}^{n_i} \to (-\infty, +\infty]$ are closed proper convex function, $A_i \in \mathcal{R}^{l \times n_i}$, $b \in \mathcal{R}^{l}$, $X_i \subseteq \mathcal{R}^{n_i}$ are nonempty closed convex sets and $n_1 + n_2 + \cdots + n_m = n$. The solution set of (1) is assumed to be nonempty throughout this paper. This kind of problem has many applications in signal and imaging processing, traffic management, machine learning and so on.

The Lagrangian function of problem (1) is

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

It has been well understood theoretically that the augmented Lagrangian method (ALM) is a benchmark approach for the linearly constrained convex optimization problem (1). The augmented Lagrangian function of problem (1) is

$$
L_\beta(x_1, x_2, \cdots, x_m; \lambda) = \sum_{i=1}^{m} \theta_i(x_i) - \lambda^T (\sum_{i=1}^{m} A_i x_i - b) + \frac{\beta}{2} \left\| \sum_{i=1}^{m} A_i x_i - b \right\|^2,
$$

where $\lambda \in \mathcal{R}^{l}$ is the Lagrangian multiplier vector corresponding to the linear constraints and $\beta > 0$ is a penalty parameter. Given $\lambda^k \in \mathcal{R}^{l}$, the ALM for solving (1)
updates the primal and the dual variables via:

\[
\begin{cases}
(x_1^{k+1}, x_2^{k+1}, \ldots, x_m^{k+1}) = \arg \min \{L_\beta(x_1, x_2, \ldots, x_m; \lambda^k) \mid x_i \in X_i\}, \\
\lambda^{k+1} = \lambda^k - \beta (\sum_{i=1}^m A_i x_i^{k+1} - b).
\end{cases}
\] (4)

In general, the ALM is sufficiently efficient for solving (1) with \(m = 1\). However, for the cases of \(m \geq 2\), the ALM could lose its efficiency due to the coupling variables. Therefore, an important motivating question is that whether one can improve the performance of the ALM by fully exploiting the valuable structures hidden in \(\theta_i\).

A natural property of (1) is that each function only depends on its own variable \(x_i\), which provides the possibility to deal with them individually in designing numerical algorithms. A very popular algorithm is the alternating direction method of multipliers (ADMM) [14,15], which generates the iterative sequence according to the following scheme:

\[
\begin{cases}
x_1^{k+1} = \arg \min \{L_\beta(x_1, x_2^k, \lambda^k) \mid x_1 \in X_1\}, \\
x_2^{k+1} = \arg \min \{L_\beta(x_1^{k+1}, x_2, \lambda^k) \mid x_2 \in X_2\}, \\
\vdots \\
x_m^{k+1} = \arg \min \{L_\beta(x_1^{k+1}, x_2^{k+1}, \ldots, x_m; \lambda^k) \mid x_m \in X_m\}, \\
\lambda^{k+1} = \lambda^k - \beta (\sum_{i=1}^m A_i x_i^{k+1} - b).
\end{cases}
\] (5)

The direct extension of the classic ADMM to the \(m\)-block convex optimization problem consists of the iterations:

\[
\begin{cases}
x_1^{k+1} = \arg \min \{L_\beta(x_1, x_2^k, \ldots, x_m^k; \lambda^k) \mid x_1 \in X_1\}, \\
x_2^{k+1} = \arg \min \{L_\beta(x_1^{k+1}, x_2, \ldots, x_m^k; \lambda^k) \mid x_2 \in X_2\}, \\
\vdots \\
x_m^{k+1} = \arg \min \{L_\beta(x_1^{k+1}, x_2^{k+1}, \ldots, x_m; \lambda^k) \mid x_m \in X_m\}, \\
\lambda^{k+1} = \lambda^k - \beta (\sum_{i=1}^m A_i x_i^{k+1} - b).
\end{cases}
\] (6)

In [2] the authors showed that the global convergence can not be guaranteed if it is directly extended to \(m\)-block cases (\(m \geq 3\)). Now there exist several methods to assure the convergence of multi-block convex programming. For example, if one of the objective functions is strongly convex [3], the direct ADMM is convergent. Besides, a series of augmented Lagrangian based splitting methods in combination with the Gauss back or the Jacobian decompositions have received considerable attention in recent years, and many improved algorithms are based on this kind of decomposition [7,8,10–12].

Among these research, there are some papers based on prediction and correction method to make the directly extended ADMM convergent. For example, the Gauss-Seidel method and fully Jacobian method [1,10]. The Gauss-Seidel method use the directly extended ADMM to generate the prediction point, and then through a correction step to get the next iterate.

In this paper, we will give a parallel Gauss-Seidel method to improve the efficiency for solving large scale problems. First we divide the variables into some groups. Each group has two blocks. In every group we update the first variable, then update the other variable with the latest information of the first variable. The information between different groups is not shared. Hence we can update each
group independently at the same time, which ensures the sub-problems to be solved in parallel.

2. Notations and preliminaries. In this section, we first summarize some notations that will be used throughout this paper. We then give the variational inequality characterization of the primal-dual optimality conditions of (1) and recall some well-known results that will play central roles in the later analysis.

Let \( \mathbb{R}^n \) be an \( n \)-dimensional Euclidean space and \( I \) be the identity matrix. The superscript "\( T \)" represents the transpose operation for matrix and vector variables. Let \( \| x \| = \sqrt{x^T x} \) as the Euclidean-norm, \( \| x \|_A = \sqrt{x^T A x} \) as \( A \)-norm where \( A \) is a symmetric positive definite matrix.

As we know, solving the convex minimization problem (1) is equivalent to determining a solution to the following mixed variational inequality (MVI) problem:

\[
\begin{align*}
\theta_i(x_i) - \theta_i(x_i^*) + (x_i - x_i^*)^T (-A_i^T \lambda^*) & \geq 0, \quad \forall x_i \in X_i, \quad i = 1, \cdots, m, \\
(\lambda - \lambda^*)^T \left( \sum_{i=1}^m A_i x_i^* - b \right) & \geq 0, \quad \forall \lambda \in \mathbb{R}^l.
\end{align*}
\] (7)

For simplicity, we give some notations:

\[
u = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix}, \quad \omega = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \\ \lambda \end{pmatrix}, \quad F(\omega) = \begin{pmatrix} -A_1^T \lambda \\ -A_2^T \lambda \\ \vdots \\ -A_m^T \lambda \\ \sum_{i=1}^m A_i x_i^* - b \end{pmatrix},
\] (8)

and \( \theta(u) = \theta(x_1) + \theta(x_2) + \cdots + \theta(x_m) \).

Using these notations, (7) can be expressed by a simpler form:

\[
\theta(u) - \theta(u^*) + (\omega - \omega^*)^T F(\omega^*) \geq 0, \quad \forall \omega \in \mathcal{W},
\] (9)

where \( \mathcal{W} = X \times \mathbb{R}^m = \prod_{i=1}^m X_i \times \mathbb{R}^m \). Denote

\[
G = \begin{pmatrix}
Q_1 & -\beta A_1^T A_2 & -\beta A_1^T A_3 & -\beta A_1^T A_4 & \cdots & 0 \\
0 & Q_2 & -\beta A_2^T A_3 & -\beta A_2^T A_4 & \cdots & 0 \\
-\beta A_3^T A_1 & -\beta A_3^T A_2 & Q_3 & -\beta A_3^T A_4 & \cdots & 0 \\
-\beta A_4^T A_1 & -\beta A_4^T A_2 & 0 & Q_4 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & 0 & \cdots & \frac{1}{n} I
\end{pmatrix},
\] (10)

which appears in the correction step of our algorithm. Let \( \omega^k = (x_1^k, x_2^k, \cdots, x_m^k, \lambda^k) \), and \( \omega^* \) is the optimization solution of (MVI).

Assumption 1. Assume that the set of solutions of MVI(\( W, F, \theta \)), denoted by \( \mathcal{W}^* \) is nonempty.

3. Algorithm and Main Results. In this section, we will present the main algorithm and analyze the convergence of the proposed algorithm.
3.1. The proposed algorithm. First we describe our algorithm as follows:

Step1: Initialization: given \((x^0_1, x^0_2, \ldots, \lambda^0)\), choose \(\beta > 0, tol > 0\);

Step2: (Prediction) 
for \(k = 0, 1, 2 \cdots \) do 
for \(i = 1, 2, \ldots, \frac{m}{2} \) do 
generate \(\tilde{x}^k_i \) via:

\[
\begin{align*}
\tilde{x}^k_{2i-1} &= \arg \min_{x_{2i-1} \in X_{2i-1}} \left\{ L_\beta(x_{2i-1}, x^k_{2i-1}, \ldots, x^k_{2m}, \lambda^k) + \frac{1}{2}\|x_{2i-1} - x^k_{2i-1}\|^2_{Q_{2i-1}} \right\}, \\
\tilde{x}^k_{2i} &= \arg \min_{x_{2i} \in X_{2i}} \left\{ L_\beta(x_{2i}, x^k_{2i-1}, \ldots, x^k_{2m}, \lambda^k) + \frac{1}{2}\|x_{2i} - x^k_{2i}\|^2_{Q_{2i}} \right\}, \\
\tilde{\lambda}^k &= \lambda^k - \beta(\sum_{i=1}^{m} A_i \tilde{x}^k_i - b).
\end{align*}
\]

(11)

end for \(i\)

Step3: When \(\|\omega_k - \tilde{\omega}_k\| < tol\), stop;

Step4: (Correction) 
where \(\alpha_k = \frac{\phi(\omega_k, \tilde{\omega}_k)}{\|G(\omega_k - \tilde{\omega}_k)\|}\); \(\phi(\omega_k, \tilde{\omega}_k) = \|\omega_k - \tilde{\omega}_k\|^2_{G + GT}\).

end for \(k\)

Remark 1. When \(\tilde{\lambda}^k = \lambda^k\), from (11) we know that \(\sum_{i=1}^{m} A_i \tilde{x}^k_i = b\), so \((\tilde{x}^k_1, \tilde{x}^k_2, \ldots, \tilde{x}^k_m)\) is a feasible point of problem (1). And by (11) and (12), when \(\omega_k = \tilde{\omega}_k\), \((\tilde{x}^k_1, \tilde{x}^k_2, \ldots, \tilde{x}^k_m)\) is a solution of problem (1). Therefore, it is reasonable to take \(\|\omega_k - \tilde{\omega}_k\| < \varepsilon\) as the stopping criteria in the algorithm.

Remark 2. In the algorithm when \(m\) is odd, we can split the first \(m - 1\) blocks to \(\frac{m-1}{2}\) groups. The last block \(\tilde{x}^k_m\) can be computed directly from

\[
\tilde{x}^k_m = \arg \min_{x_m \in X_m} \left\{ L_\beta(x^k_1, x^k_2, \ldots, x^k_{m-1}, x_m, \lambda^k) + \frac{1}{2}\|x_m - x^k_m\|^2_{Q_m} \right\}.
\]

The proposed algorithm has a few advantages. First, we can get the global convergence by choosing suitable matrices \(Q_i\). Second, the \(x_i\)-subproblem will be strictly or strongly convex by adding the proximal term. So it has a unique solution. Third, there are multiple choices for matrices \(Q_i\) with which the subproblem become easier to solve. Specifically, the \(x_i\)-subproblem contains a quadratic term \(\frac{\beta}{2} x_i^TA_i^TA_i x_i\). One can choose \(Q_i = D_i - \beta A_i^TA_i\), where the matrix \(D_i\) is a simple matrix (e.g., a diagonal matrix), thereby leading to an easier subproblem.

Here we offer two commonly used choices of \(Q_i\):
1. \(Q_i = \tau_i I(\tau_i > 0)\): this is a standard proximal method;
2. \(Q_i = \tau_i I - \beta A_i^TA_i(\tau_i > 0)\): this corresponds to the prox-linear method.

In order to ensure the convergence of the algorithm, the matrices \(Q_1, Q_2, \cdots, Q_m\) are symmetric positive definite and needs to satisfy other conditions so that \(G + GT\) is symmetric positive definite. Next, we analyze the matrix \(G\) and derive the specific requirements for \(Q_i\).
For convenience, suppose the number of blocks is 4, and

$$G + G^T = \begin{pmatrix}
    Q_1 + Q_1^T & -\beta A_1^T A_2 & -2\beta A_1^T A_3 & -2\beta A_1^T A_4 & 0 \\
    -\beta A_2^T A_1 & Q_2 + Q_2^T & -2\beta A_2^T A_3 & -2\beta A_2^T A_4 & 0 \\
    -2\beta A_3^T A_1 & -2\beta A_3^T A_2 & Q_3 + Q_3^T & -\beta A_3^T A_4 & 0 \\
    0 & 0 & 0 & Q_4 + Q_4^T & 0 \\
\end{pmatrix}. \quad (13)$$

Let

$$H = \begin{pmatrix}
    Q_1 + Q_1^T & -\beta A_1^T A_2 & -2\beta A_1^T A_3 & -2\beta A_1^T A_4 \\
    -\beta A_2^T A_1 & Q_2 + Q_2^T & -2\beta A_2^T A_3 & -2\beta A_2^T A_4 \\
    -2\beta A_3^T A_1 & -2\beta A_3^T A_2 & Q_3 + Q_3^T & -\beta A_3^T A_4 \\
    -2\beta A_4^T A_1 & -2\beta A_4^T A_2 & -\beta A_4^T A_3 & Q_4 + Q_4^T \\
\end{pmatrix}, \quad (14)$$

then

$$G + G^T = \begin{pmatrix}
    H & 0 \\
    0 & \frac{1}{\beta} I \\
\end{pmatrix}.$$ 

Let $B_i = Q_i + Q_i^T - \tau \beta A_i^T A_i$, $\tau > 0$ be a parameter, then

$$H = B + M,$$

where $B, M$ are defined as follows

$$B = \begin{pmatrix}
    B_1 & 0 & 0 & 0 \\
    0 & B_2 & 0 & 0 \\
    0 & 0 & B_3 & 0 \\
    0 & 0 & 0 & B_4 \\
\end{pmatrix},$$

$$M = \begin{pmatrix}
    \tau \beta A_1^T A_1 & -\beta A_1^T A_2 & -2\beta A_1^T A_3 & -2\beta A_1^T A_4 \\
    -\beta A_2^T A_1 & \tau \beta A_2^T A_2 & -2\beta A_2^T A_3 & -2\beta A_2^T A_4 \\
    -2\beta A_3^T A_1 & -2\beta A_3^T A_2 & \tau \beta A_3^T A_3 & -\beta A_3^T A_4 \\
    -2\beta A_4^T A_1 & -2\beta A_4^T A_2 & -\beta A_4^T A_3 & \tau \beta A_4^T A_4 \\
\end{pmatrix}.$$ 

It is easy to see that $M$ can be decomposed as

$$\text{diag}(\sqrt{\beta} A_1, \cdots, \sqrt{\beta} A_4)^T \begin{pmatrix}
    \tau I & -I & -2I & -2I \\
    -I & \tau I & -2I & -2I \\
    -2I & -2I & \tau I & -I \\
    -2I & -2I & -I & \tau I \\
\end{pmatrix} \text{diag}(\sqrt{\beta} A_1, \cdots, \sqrt{\beta} A_4).$$ 

Let

$$\tilde{M} = \begin{pmatrix}
    \tau I & -I & -2I & -2I \\
    -I & \tau I & -2I & -2I \\
    -2I & -2I & \tau I & -I \\
    -2I & -2I & -I & \tau I \\
\end{pmatrix}.$$ 

When $\tau > 5$, $\tilde{M}$ is a strictly diagonally dominant matrix. The main diagonal element is greater than 0, so it is a positive definite matrix. In addition, $M$ is positive semi-definite matrix. If $B_i = Q_i + Q_i^T - \tau \beta A_i^T A_i > 0$ for $i = 1, 2, 3, 4$, then $B$ is a positive definite matrix. Hence $H = B + M$ is a positive definite matrix, and then $G + G^T$ is a positive definite matrix. Through the above analysis, choosing an appropriate $Q_i$ can ensure $G + G^T$ to be positive definite.

When the number of blocks exceeds 4, we can divide the blocks into $N$ groups and decompose $G + G^T$ by the above method. It is obvious that to make $M$ be a
positive semi-definite matrix, it is enough to make the following matrix $\widetilde{M}$ positive definite:

$$
\widetilde{M} = \begin{pmatrix}
\tau I & -I & -2I & -2I & \cdots & -2I & -2I \\
-I & \tau I & -2I & -2I & \cdots & -2I & -2I \\
-2I & -2I & \tau I & -I & \cdots & -2I & -2I \\
-2I & -2I & -I & \tau I & \cdots & -2I & -2I \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
-2I & -2I & -2I & -2I & \cdots & \tau I & -I \\
-2I & -2I & -2I & -2I & \cdots & -I & \tau I
\end{pmatrix}.
$$

According to the special structure of $\widetilde{M}$, we know that $\widetilde{M}$ is a strictly diagonally dominant matrix when $\tau > 2N - 3$, which leads to the positive definiteness of $G + G^T$.

### 3.2. Convergence Analysis

In this section, we will analyze the convergence of the proposed algorithm. Suppose that Assumption 1 always holds. In order to simplify the analysis process, we assume that the number of blocks is 4. The proof of convergence is similar when the number of blocks is more than 4. First we give the following two lemmas.

**Lemma 3.1.** Let $\overline{w}^k = (\overline{x}^k_1, \overline{x}^k_2, \overline{x}^k_3, \overline{x}^k_4, \overline{\lambda}^k) \in W$ be the sequence generated by Algorithm 1. Then we have

$$(\overline{w}^k - \omega^*)^T G (\omega^k - \overline{w}^k) \geq 0, \; \forall \omega^* \in W^*,$n

where $G$ is defined in (10).

**Proof.** Since $\overline{x}^k_i$ is the solution of the sub-problem (11), we have:

$$
\begin{align*}
\theta_1(x_1) - \theta_1(\overline{x}_1^k) + (x_1 - \overline{x}_1^k)^T \{-A_1^T \lambda^k + \beta A_1^T (A_1 x_1^k + \sum_{i=2}^{4} A_i x_i^k + Q_1(\overline{x}_1^k - x_1^k))\} &\geq 0, \\
\theta_2(x_2) - \theta_2(\overline{x}_2^k) + (x_2 - \overline{x}_2^k)^T \{-A_2^T \lambda^k + \beta A_2^T (\sum_{i=1}^{2} A_i x_i^k + \sum_{i=3}^{4} A_i x_i^k + Q_2(\overline{x}_2^k - x_2^k))\} &\geq 0, \\
\theta_3(x_3) - \theta_3(\overline{x}_3^k) + (x_3 - \overline{x}_3^k)^T \{-A_3^T \lambda^k + \beta A_3^T (\sum_{i=1,2} A_i x_i^k + A_4 x_3^k + Q_3(\overline{x}_3^k - x_3^k))\} &\geq 0, \\
\theta_4(x_4) - \theta_4(\overline{x}_4^k) + (x_4 - \overline{x}_4^k)^T \{-A_4^T \lambda^k + \beta A_4^T (\sum_{i=1}^{4} A_i x_i^k + Q_4(\overline{x}_4^k - x_4^k))\} &\geq 0, \\
(\lambda - \overline{\lambda}^k)^T (A_1 x_1^k + A_2 x_2^k + A_3 x_3^k + A_4 x_4^k - b + \frac{1}{\beta}(\overline{\lambda}^k - \lambda^k)) &\geq 0,
\end{align*}
$$

for any $\omega \in W$ according to the first-order optimality condition. By

$$
\overline{\lambda}^k = \lambda^k - \beta (\sum_{i=1}^{m} A_i \overline{x}_i^k - b),
$$
we get
\[
\begin{cases}
\theta_1(x_1) - \theta_1(\bar{x}_1^k) + (x_1 - \bar{x}_1^k)^T \{ - A_1^T \bar{x}_1^k - \beta A_1^T \left( \sum_{i=2}^{4} A_i(\bar{x}_i^k - x_i^k) \right) + Q_1(\bar{x}_1^k - x_1^k) \} \geq 0, \\
\theta_2(x_2) - \theta_2(\bar{x}_2^k) + (x_2 - \bar{x}_2^k)^T \{ - A_2^T \bar{x}_2^k - \beta A_2^T \left( \sum_{i=3}^{4} A_i(\bar{x}_i^k - x_i^k) \right) + Q_2(\bar{x}_2^k - x_2^k) \} \geq 0, \\
\theta_3(x_3) - \theta_3(\bar{x}_3^k) + (x_3 - \bar{x}_3^k)^T \{ - A_3^T \bar{x}_3^k - \beta A_3^T \left( \sum_{i \neq 3} A_i(\bar{x}_i^k - x_i^k) \right) + Q_3(\bar{x}_3^k - x_3^k) \} \geq 0, \\
\theta_4(x_4) - \theta_4(\bar{x}_4^k) + (x_4 - \bar{x}_4^k)^T \{ - A_4^T \bar{x}_4^k - \beta A_4^T \left( \sum_{i=1}^{2} A_i(\bar{x}_i^k - x_i^k) \right) + Q_4(\bar{x}_4^k - x_4^k) \} \geq 0,
\end{cases}
\]
\[
(\lambda - \bar{\lambda})^T (A_1\bar{x}_1^k + A_2\bar{x}_2^k + A_3\bar{x}_3^k + A_4\bar{x}_4^k - b + \frac{1}{\beta} (\bar{\lambda} - \lambda)) \geq 0.
\]

Using the notations (8) and (10), this mixed variational inequality can be written in the form of
\[
\theta(\omega) - \theta(\bar{\omega}) + (\omega - \bar{\omega})^T \{ F(\bar{\omega}) + G(\bar{\omega} - \omega) \} \geq 0, \quad \forall \omega \in W. \tag{15}
\]

Since \( F \) is a monotone mapping, for any \( \bar{\omega} \in W \) and \( \omega^* \in W^* \), the following inequality holds:
\[
\theta(\bar{\omega}) - \theta(\omega^*) + (\bar{\omega} - \omega^*)^T F(\bar{\omega}) \geq \theta(\bar{\omega}) - \theta(\omega^*) + (\bar{\omega} - \omega^*)^T F(\omega^*) \geq 0.
\]

Setting \( \omega = \omega^* \) in (15), we obtain
\[
(\bar{\omega} - \omega^*)^T G(\omega^* - \bar{\omega}) \geq 0.
\]

Lemma 3.2. In Algorithm 1, \( \alpha_k \) is bounded away from 0.

Proof. By the definition of \( \alpha_k \), we have
\[
\alpha_k = \frac{\| \omega_k - \bar{\omega}_k \|^2_{G + GT}}{\| G(\omega^* - \bar{\omega}) \|^2} = \frac{\left( \omega_k - \bar{\omega}_k \right)^T (G + GT)(\omega_k - \bar{\omega}_k)}{\left( \omega_k - \bar{\omega}_k \right)^T G(\omega_k - \bar{\omega}_k)} \geq \frac{\lambda_{\min}(G + GT)}{\lambda_{\max}(GTG)} > 0,
\]
the last inequality hold because \( G + GT \) and \( GTG \) are positive definite matrix. \( \Box \)

Next, we will prove that the sequence generated by Algorithm 1 is contractive respect to the set \( W \).

Theorem 3.3. Let \{\( \omega^k \)\} and \{\( \bar{\omega}^k \)\} be the sequences generated by Algorithm 1. Then for any \( \omega^* \in W \), we have
\[
\| \omega^{k+1} - \omega^* \|^2 \leq \| \omega^k - \omega^* \|^2 - \alpha_k \phi(\omega_k, \bar{\omega}_k).
\]

Proof. Since \( \omega^{k+1} = \omega^k - \alpha_k G(\omega^k - \bar{\omega}^k) \), and by the definition of
\[
\phi(\omega_k, \bar{\omega}_k) = \| \omega_k - \bar{\omega}_k \|^2_{G + GT} \quad \text{and} \quad \alpha_k = \frac{\phi(\omega_k, \bar{\omega}_k)}{\| G(\omega^k - \bar{\omega}) \|^2},
\]

we have
\[
\|\omega^{k+1} - \omega^*\|^2 = \|\omega^k - \omega^* - \alpha_k G(\omega^k - \bar{\omega}^k)\|^2 \\
= \|\omega^k - \omega^*\|^2 - 2\alpha_k (\omega^k - \omega^*^2)T G(\omega^k - \bar{\omega}^k) + \alpha_k^2\|G(\omega^k - \bar{\omega}^k)\|^2 \\
= \|\omega^k - \omega^*\|^2 - 2\alpha_k (\omega^k - \omega^*^2)T G(\omega^k - \bar{\omega}^k) - 2\alpha_k (\bar{\omega}^k - \omega^*)^2 G(\omega^k - \bar{\omega}^k) + \alpha_k^2\|G(\omega^k - \bar{\omega}^k)\|^2 \\
\leq \|\omega^k - \omega^*\|^2 - 2\alpha_k (\omega^k - \bar{\omega}^k)^2 G(\omega^k - \bar{\omega}^k) + \alpha_k^2\|G(\omega^k - \bar{\omega}^k)\|^2 \\
= \|\omega^k - \omega^*\|^2 - 2\alpha_k (\omega^k - \bar{\omega}^k)^2 + \alpha_k^2\|G(\omega^k - \bar{\omega}^k)\|^2 \\
= \|\omega^k - \omega^*\|^2 - \alpha_k^2\|G(\omega^k, \bar{\omega}^k)\|^2,
\]
thus we complete the proof. \(\square\)

Theorem 3.3 shows that the sequence \(\{\|\omega^{k+1} - \omega^*\|\}\) is non-increasing. Based on the lemmas and Theorem 3.3, we can get the global convergence result of the Algorithm 1.

**Theorem 3.4.** Let \(\{\omega^k\}\) and \(\{\bar{\omega}^k\}\) be the sequences generated by Algorithm 1. Then

1. The sequence \(\{\omega^k\}\) is bounded.
2. \(\lim_{k \to \infty} \|\omega^k - \bar{\omega}^k\| = 0\).
3. Any cluster point of \(\{\bar{\omega}^k\}\) is a solution point of (1).
4. The sequence \(\{\omega^k\}\) converges to some \(\omega^\infty \in \mathcal{W}^*\).

**Proof.** According to Theorem 3.3, the sequence \(\{\|\omega^{k+1} - \omega^*\|\}\) is non-increasing. Hence
\[
\|\omega^{k+1} - \omega^*\| \leq \|\omega^k - \omega^*\| \leq \cdots \leq \|\omega^0 - \omega^*\|.
\]
The first assertion holds.

From Theorem 3.3, we have
\[
\alpha_k\|\omega^k - \bar{\omega}^k\|^2 G(\omega^k, \bar{\omega}^k) \leq \|\omega^k - \omega^*\|^2 - \|\omega^{k+1} - \omega^*\|^2.
\]
Summing the inequality from 0 to \(n\), then we get
\[
\sum_{k=0}^{n} \alpha_k\|\omega^k - \bar{\omega}^k\|^2 G(\omega^k, \bar{\omega}^k) \leq \|\omega^0 - \omega^*\|^2 - \|\omega^{n+1} - \omega^*\|^2 \leq \|\omega^0 - \omega^*\|^2.
\]
Let \(n \to \infty\), then
\[
\sum_{k=0}^{\infty} \alpha_k\|\omega^k - \bar{\omega}^k\|^2 G(\omega^k, \bar{\omega}^k) \leq \|\omega^0 - \omega^*\|^2.
\]
and consequently we have
\[
\lim_{k \to \infty} \|\omega^k - \bar{\omega}^k\| = 0.
\]

So
\[
\lim_{k \to \infty} \|x_i^k - \bar{x}_i^k\| = 0, \ i = 1, \cdots, N,
\]
and
\[
\lim_{k \to \infty} \|\lambda^k - \bar{\lambda}^k\| = 0.
\]
The second assertion is proved.
Because \( \{\omega^k\} \) is bounded, then \( \{\tilde{\omega}^k\} \) is bounded, so there is at least one cluster point of \( \{\tilde{\omega}^k\} \), denoted by \( \omega^\infty \). Let \( \{\tilde{\omega}^k_j\} \subset W \), and \( \lim_{j \to \infty} \tilde{\omega}^k_j = \omega^\infty \). When \( j \to \infty \) in

\[
\theta(u) - \theta(\tilde{\omega}^k_j) + (\omega - \tilde{\omega}^k_j)^T \{F(\tilde{\omega}^k_j) + G(\tilde{\omega}^k_j - \omega^k_j)\} \geq 0,
\]

we have

\[
\theta(u) - \theta(\omega^\infty) + (\omega - \omega^\infty)^T F(\omega^\infty) \geq 0, \quad \forall \omega \in W.
\]

Thus any cluster point of \( \tilde{\omega}^k \) is a solution point of (1.1).

At last we prove that \( \{\tilde{\omega}^k\} \) converges to \( \omega^\infty \). From assertion 2, we have \( \lim_{j \to \infty} \omega^k_j = \omega^\infty \).

As \( \{\omega^k_j\} \) is a subsequence of \( \{\omega^k\} \), for any \( k \), there exists \( k_j \) and \( k_{j+1} \) satisfy

\[
k_j \leq k < k_{j+1}.
\]

Since the sequence \( \{\|\omega^{k+1} - \omega^*\|\} \) is non-increasing, then we get

\[
\|\omega^k_{j+1} - \omega^\infty\| \leq \|\omega^k - \omega^\infty\| \leq \|\omega^{k+1} - \omega^\infty\|,
\]

so \( \{\omega^k\} \) converges to \( \omega^\infty \).

4. Numerical Experiments. In this section, we present some numerical examples to illustrate the performance of our algorithm. All codes were written by Matlab(R2016a) and all numerical experiments were conducted on a laptop with an Intel Core i5 and 64-bit operating system.

4.1. The Basis Pursuit Problem. The basis pursuit problem is defined as :

\[
\min_x \quad \|x\|_1 \\
\text{s.t.} \quad Ax = b, \quad x \in X,
\]

where \( x \in \mathbb{R}^n \), \( A \in \mathbb{R}^{m \times n} \) and \( b \in \mathbb{R}^m (m < n) \).

According to the compressive sensing theory, the basis pursuit problem is deemed to recover the sparsity vector \( x \) from a small number of observations \( b \). In the experiment, we divide the data into \( N \)-block first, i.e. \( x = [x_1, x_2, \cdots, x_N]^T \), \( A = [A_1, A_2, \cdots, A_N] \). Therefore, the problem (16) can be rewritten as:

\[
\min_x \quad \sum_{i=1}^{N} \|x_i\|_1 \\
\text{s.t.} \quad \sum_{i=1}^{N} A_i x_i = b, \quad x \in X,
\]

which is in the form of problem (1) with \( \theta_i(x_i) = \|x_i\|_1 \).

We test the problem by our algorithm, JADMM [5] and BWLADMM(Block-wise linearized ADMM) [16]. First we set \( N = 4 \). A sparse vector \( x^* \in \mathbb{R}^n \) is randomly generated with \( k \) (\( k \ll n \)) nonzeros drawn from the standard Gaussian distribution. Matrix \( A \in \mathbb{R}^{m \times n} \) is also randomly generated from the standard Gaussian distribution, and it needs to be partitioned into 4 blocks correspondingly. The vector \( b \) is then computed by \( b = Ax^* + \varepsilon \), where \( \varepsilon \sim N(0, \sigma^2 I) \) is Gaussian noise with standard deviation \( \sigma \). The maximal number of iterations is set as 1000, and the stopping criterion is \( \|\omega^k - \tilde{\omega}^k\| < tol \). We set \( tol = 1e-3 \), \( \beta = 0.6 \) for all three algorithms, \( \alpha_k \) is computed by our algorithm in each iteration. To solve this...
problem (17), we need to solve several subproblems. For example, $x_1$-subproblem is as follows:

$$
\tilde{x}_k^1 = \arg \min_{x_1 \in X_1} \{ \|x_1\|_1 - (\lambda^k)^T (A_1 x_1) + \frac{\beta}{2} \| A_1 x_1 + \sum_{i=2}^4 A_i x_i^k - b \|^2 + \frac{1}{2} \| x_1 - x_k^1 \|^2_{Q_1} \} \tag{18}.
$$

Generally $A_1$ is not an identity matrix, so this subproblem can not be solved directly. If we choose $Q_1 = \tau_1 - \beta A_1^T A_1$, $x_1$-subproblem (18) can be converted into the following problem:

$$
\tilde{x}_k^1 = \arg \min_{x_1 \in X_1} \{ \|x_1\|_1 + (x_1 - x_k^1)^T A_1^T (A x_k^1 - b - \frac{\lambda^k}{\beta}) + \frac{\tau_1}{2} \|x_1 - x_k^1\|^2 \} \tag{19}.
$$

(19) admits a simple closed-form solution by shrinkage operator. Here we choose $\tau_1$ which satisfies $\tau_1 > \frac{2N-1}{2} \beta \|A_1^T A_1\|$ to assure the global convergence of the algorithm. Similarly, we set $P_i = \tau_i - \beta A_i^T A_i$, and $\tau_i > N \beta \|A_i^T A_i\|$ according to the algorithm JADMM. As for BWLADMM, we take the linearized version A strategy which divide the variables into two groups. Every group has $\frac{N}{2}$ variables. We set $k_1 > \frac{N}{2} \max\{ \rho(A_i^T A) | i = 1, \ldots, \frac{N}{2} \}$ and $k_2 > \frac{N}{2} \max\{ \rho(A_i^T A) | i = \frac{N}{2} + 1, \ldots, N \}$ according to the algorithm BWLADMM.

We compare our algorithm to JADMM and BWLADMM with different scales of matrix $A$. Matrix $A$ and sparse vector $x^*$ are randomly generated. The numerical results are recorded in Table 1, where the number of iterations is denoted by “Iter.”, the optimal objective function value is denoted by “Obj.”, and the true optimal objective function value is denoted by “Real $\|x^*\|_1$.”

### Table 1. The results of Algorithm 1, JADMM and BWLADMM with different scales of $A$

| Problem | Algorithm 1 | JADMM | BWLADMM | Real $\|x^*\|_1$ |
|---------|-------------|-------|---------|-----------------|
|         | Iter. | Obj. | Iter. | Obj. | Iter. | Obj. | Iter. | Obj. | Real $\|x^*\|_1$ |
| $m=20$, $n=100$ | 90 | 3.6410 | 239 | 3.6405 | 110 | 3.6381 | 3.6406 |
|         | 111 | 4.4316 | 263 | 4.4299 | 135 | 4.4310 | 4.4325 |
|         | 196 | 3.7944 | 346 | 3.7957 | 198 | 3.7950 | 3.7948 |
| $m=40$, $n=200$ | 158 | 8.2603 | 335 | 8.2617 | 170 | 8.2603 | 8.2641 |
|         | 198 | 7.2109 | 253 | 7.2104 | 222 | 7.2109 | 7.2093 |
|         | 277 | 8.3752 | 439 | 8.3811 | 343 | 8.3763 | 8.4107 |
| $m=80$, $n=400$ | 223 | 16.6467 | 336 | 16.6479 | 235 | 16.6454 | 16.6350 |
|         | 304 | 12.5305 | 631 | 12.5343 | 320 | 12.5316 | 13.0210 |
|         | 349 | 17.1994 | 519 | 17.1929 | 364 | 17.1981 | 17.1707 |
| $m=100$, $n=500$ | 240 | 20.3980 | 364 | 20.4009 | 265 | 20.3978 | 20.3860 |
|         | 319 | 16.7817 | 508 | 16.7750 | 325 | 16.7714 | 16.8788 |
|         | 388 | 17.9074 | 651 | 17.9105 | 450 | 17.9112 | 18.0729 |
| $m=200$, $n=1000$ | 356 | 42.4486 | 482 | 42.4570 | 364 | 42.4554 | 42.5060 |
|         | 358 | 45.2504 | 523 | 45.2505 | 372 | 45.2531 | 45.3523 |
|         | 411 | 36.1374 | 495 | 36.1408 | 432 | 36.1393 | 36.2120 |

As shown in Table 1, our algorithm and BWLADMM perform better than JADMM for most of the test problems. Besides, our algorithm and BWLADMM share a similar iteration times, and our algorithm has a slight advantage than BWLADMM. The main reason is that our algorithm and BWLADMM use the latest information of iteration variables which are mixture of alternating and parallel decomposition,
whereas JADMM is a fully-parallel algorithm which only use the previous step information. In our algorithm, the step size $\alpha_k$ at each iteration could be found towards the purpose of maximizing the contraction of the sequence, so the iteration times is a little less than BWLADMM in numerical performance.

4.2. **Separable Quadratic Programming Problem.** Consider the following separable quadratic programming problem:

$$\min_{\{x_i\}} \sum_{i=1}^{N} \left( \frac{1}{2} x_i^T H_i x_i + c_i^T x_i \right)$$

s.t. $\sum_{i=1}^{N} A_i x_i = b,$

where $H_i \in \mathbb{R}^{n \times n}$ are positive definite matrices, $c_i \in \mathbb{R}^n$ are column vectors, $A_i \in \mathbb{R}^{m \times n}$ are full-column-rank matrices ($i = 1, \ldots, N$), $b \in \mathbb{R}^m$ is a column vector.

In the numerical experiments, $H_1 = H_2 = \cdots = H_N = 2I$, $c_1 = c_2 = \cdots = c_N = -2 \times [1, 2, 3, \ldots, n]^T$, $A_1 = A_2 = \cdots = A_N = I$, $b = N \times [1, 2, 3, \ldots, n]^T$, where $I$ denotes the $n \times n$ identity matrix. It is easy to see that for each $i$, $x_i^* = x_1^* = x_2^* = \cdots = x_N^* = \lambda_0 = e$ is the initial point, where $e$ denotes the $n$-dimensional column vector with all elements being 1.

For this problem, we compare our algorithm with PPPCSA [1], JADMM [5], and BWLADMM [16]. Because PPPCSA is used to solve three-block convex problems with linear constraints, we first set $N = 3$. Then we set $N = 4, 6$ to show the result of our algorithm and JADMM, BWLADMM. $tol = 10^{-6}$ is used as the termination criterion precision.

4.2.1. The number of block with $N = 3$. In this case, $Q_1 = Q_2 = \frac{2x^3-3x+1}{2} \beta I$, $Q_3 = \frac{(2x^3-3x+1)}{2} \beta I$ in our algorithm to ensure $G + G^T$ be a positive definite matrix. We set parameter $\gamma = 1$ and $P_1 = P_2 = P_3 = 3 \beta I$ in JADMM. For BWLADMM, we use the linearized version A. This algorithm regroups the variables into two groups. Here we let the first group have one variable and the second group have two variables. For ensuring the convergence of algorithm, we set $k_1 = 2$, $k_2 = 3$. For each case with different $n$, we have tried many different step size $\alpha$ and penalty parameter $\beta$ in order to find a better numerical performance for PPPCSA. As a result, we set $\beta = 0.7$ for our algorithm, JADMM and BWLADMM, and for PPPCSA we set $\alpha = 0.42, \beta = 1.3$. All the four algorithms are terminated successfully with the given $tol$. The number of iterations is denoted by “Iter.” and the optimal objective function value is denoted by “Obj.” The numerical results are shown in Figure 1 and Table 2:

| Problem | Algorithm 1 | PPPCSA | JADMM | BWLADMM |
|---------|-------------|--------|--------|----------|
| Iter.   | Obj.        | Iter.  | Obj.   | Iter.    |
| 20      | 19          | -8.6100e+03 | 49     | -8.6100e+03 | 19     | -8.6100e+03 | 29     | -8.6100e+03 |
| 100     | 16          | -1.0150e+06 | 54     | -1.0150e+06 | 21     | -1.0150e+06 | 22     | -1.0150e+06 |
| 500     | 20          | -1.2538e+08 | 59     | -1.2538e+08 | 22     | -1.2538e+08 | 24     | -1.2538e+08 |
| 1000    | 22          | -1.0015e+09 | 61     | -1.0015e+09 | 23     | -1.0015e+09 | 25     | -1.0015e+09 |
Figure 1. The relation between iteration times and stop criteria of four algorithms with different $n$

Figure 1 and Table 2 show that all the three algorithms are able to find the optimal solution for different $n$. On the other hand, it verifies that our algorithm is convergent for 3-block linear constrained convex problems by selecting some appropriate matrices $Q_i$. In addition, it can be seen that under the same stop criteria, the iteration number of our algorithm is less than PPPCSA, JADMM, and BWLADMM.

4.2.2. The number of block with $N = 4, 6$. In order to ensure $G + G^T$ be positive definite matrix, we set $Q_i = (\frac{2 \times N - 3 + 1}{2}) \times \beta I$ in our algorithm. Let $\gamma = 1$ and $P_i = N \times \beta I$ in JADMM. In BWLADMM, we set every group have $\frac{N}{2}$ variables and set $k_1 = k_2 = \frac{N}{2} + 1$. The scale of the problem is $n = 300$. We set penalty parameter $\beta = 0.7$ for all the three algorithms. The three algorithms are terminated successfully with the specified termination precision. The numerical results are shown in Figure 2.

The number of iterations is 22, 23, 28 when $N = 4$, and 16, 20, 31 when $N = 6$ for our algorithm, JADMM and BWLADMM respectively. It can be seen that our algorithm performs better in terms of the number of iterations.

Besides the above examples, we also test some problems without correction step. The next iterate is obtained by the following formula directly. We compared the results of Algorithm 1, JADMM and BWLADMM. Figure 3 shows that a direct update performs better than the other three algorithms. Until now we can not get a theoretical convergence result about the direct algorithm. This will motivate us
to study more about the algorithm.

\[
\begin{align*}
    x_{2i-1}^{k+1} &= \arg\min_{x_{2i-1} \in \mathbb{R}} \left\{ L_{\beta}(x_1^k, x_2^k, \ldots, x_{2i-2}^k, \ldots, x_m^k, \lambda^k) + \frac{1}{2} \|x_{2i-1} - x_{2i-1}^k\|_2^2 \right\}, \\
    x_{2i}^{k+1} &= \arg\min_{x_{2i} \in \mathbb{R}} \left\{ L_{\beta}(x_1^k, x_2^k, \ldots, x_{2i-1}^k, x_{2i+1}^k, \ldots, x_m^k, \lambda^k) + \frac{1}{2} \|x_{2i} - x_{2i}^k\|_2^2 \right\}, \\
    \lambda^{k+1} &= \lambda^k - \frac{\gamma}{\beta} (\sum_{i=1}^m A_i x_i^{k+1} - b).
\end{align*}
\]

**Figure 2.** The results of Algorithm 1, JADMM and BWLADMM with different $N$

**Figure 3.** The comparison of Algorithm 1, JADMM, BWLADMM and DPGSM
5. Conclusion. In this paper, we proposed a method to solve the separable convex optimization problem with linear constraints. The main idea is that we divide the variables into different groups while solving the subproblems. Each group has two blocks of variables. In the inner group, we use Gauss-Seidel method solving the subproblem. Among the different groups, Jacobi-like method is used. It can be solved in parallel during each iteration. We prove the convergence of the algorithm. Numerical results show that our method is valid.

REFERENCES

[1] F. Bai and L. Xu, A partially parallel prediction-correction splitting method for convex optimization problems with separable structure, J. Oper. Res. Soc. China, 5 (2017), 529–544.
[2] C. Chen, B. He, X. Yuan and Y. Ye, The direct extension of ADMM for multi-block convex minimization problem is not necessarily convergent, Math. Program., 155 (2016), 57–79.
[3] X. Cai, D. Han, and X. Yuan, On the convergence of the direct extension of ADMM for three-block separable convex minimization models with one strongly convex function, Comput. Optim. Appl., 66 (2017), 39–73.
[4] W. Deng and W. Yin, On the global and linear convergence of the generalized alternating direction method of multipliers, J Sci. Comput., 66 (2016), 889–916.
[5] W. Deng, M. Lai, Z. Peng, W. Yin, Parallel multi-block admm with $o(1/k)$ convergence, J. Sci. Comput., 71 (2017), 712–736.
[6] J. Eckstein and D. Bertsekas, On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators, Math. Program., 55 (1992), 293–318.
[7] B. He, Parallel splitting augmented Lagrangian methods for monotone structured variational inequalities, Comput. Optim. Appl., 42 (2009), 195–212.
[8] B. He, M. Tao and X. Yuan, Alternating direction method with Gaussian back substitution for separable convex programming, SIAM J. Optim., 22 (2012), 313–340.
[9] B. He, M. Tao and X. Yuan, A splitting method for separable convex programming, IMA J. Numer. Anal., 35 (2015), 394–426.
[10] B. He, L. Hou and X. Yuan, On full Jacobian decomposition of the augmented Lagrangian method for separable convex programming, SIAM J. Optim., 25 (2015), 2274–2312.
[11] D. Han, X. Yuan and W. Zhang, An augmented-Lagrangian-based parallel splitting method for separable convex minimization with applications to image processing, Math. Comput., 83 (2014), 2263–2291.
[12] D. Han, X. Yuan, W. Zhang and X. Cai, An ADM-based splitting method for separable convex programming, Comput. Optim. Appl., 54 (2013), 343–369.
[13] X. Gao and S. Zhang, First-order algorithms for convex optimization with nonseparable objective and coupled constraints, J. Oper. Res. Soc. China, 5 (2016), 1–29.
[14] D. Gabay and B. Mercier, A dual algorithm for the solution of nonlinear variational problems via finiteelement approximations, Comput. Math. Appl., 2 (1976), 17–40.
[15] R. Glowinski and A. Marroco, Sur l’approximation par éléments nis dordre un, et la résolution par pénalisation-dualité d’une classe de problèmes de Dirichlet nonlinéaires, J. Equine. Vet. Sci., 2 (1975), 41–76.
[16] Z. Wu, X. Cai, D. Han, Linearized block-wise alternating direction method of multipliers for multiple-block convex programming, Journal of Industrial and Management Optimization, 14 (2018), 833–855.

Received March 2020; 1st revision August 2020; Final revision September 2020.

E-mail address: 180902062@njnu.edu.cn
E-mail address: nnsd.wangnan@163.com
E-mail address: xulingling@njnu.edu.cn