Parallel Multi-Block ADMM with $o(1/k)$ Convergence

Wei Deng[1] · **Ming-Jun Lai**[2] · **Zhimin Peng**[3] · **Wotao Yin**[3]

Received: 17 April 2015 / Revised: 20 May 2016 / Accepted: 1 November 2016 /
Published online: 12 November 2016

Abstract This paper introduces a parallel and distributed algorithm for solving the following minimization problem with linear constraints:

$$\begin{aligned} \text{minimize} \quad & f_1(\mathbf{x}_1) + \cdots + f_N(\mathbf{x}_N) \\ \text{subject to} \quad & A_1\mathbf{x}_1 \ + \cdots + A_N\mathbf{x}_N = c, \\ & \mathbf{x}_1 \in \mathcal{X}_1, \ \ \ldots, \ \mathbf{x}_N \in \mathcal{X}_N, \end{aligned}$$

where $N \geq 2$, f_i are convex functions, A_i are matrices, and $\mathcal{X}_i$ are feasible sets for variable $\mathbf{x}_i$. Our algorithm extends the alternating direction method of multipliers (ADMM) and decomposes the original problem into N smaller subproblems and solves them in parallel at each iteration. This paper shows that the classic ADMM can be extended to the N-block Jacobi fashion and preserve convergence in the following two cases: (i) matrices A_i are mutually near-orthogonal and have full column-rank, *or* (ii) proximal terms are added to the N subproblems (but without any assumption on matrices A_i). In the latter case, certain proximal terms can let the subproblem be solved in more flexible and efficient ways. We show that $\|\mathbf{x}^{k+1} - \mathbf{x}^k\|_M^2$ converges at a rate of $o(1/k)$ where M is a symmetric positive semi-definte matrix. Since the parameters used in the convergence analysis are conservative, we introduce a strategy for automatically tuning the parameters to substantially accelerate

✉ Ming-Jun Lai
mjlai@uga.edu

Wei Deng
wei.deng@rice.edu

Zhimin Peng
zhimin.peng@math.ucla.edu

Wotao Yin
wotaoyin@math.ucla.edu

[1] Department of Computational and Applied Mathematics, Rice University, Houston, TX 77005, USA

[2] Department of Mathematics, University of Georgia, Athens, GA 30602, USA

[3] Department of Mathematics, University of California, Los Angeles, CA 90095, USA

our algorithm in practice. We implemented our algorithm (for the case ii above) on Amazon EC2 and tested it on basis pursuit problems with >300 GB of distributed data. This is the first time that successfully solving a compressive sensing problem of such a large scale is reported.

Keywords Alternating direction method of multipliers · ADMM · Parallel and distributed computing · Convergence rate

1 Introduction

Let $N \geq 2$ be an integer. We consider the following convex optimization problem with N blocks of variables:

$$\min_{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_N} \sum_{i=1}^{N} f_i(\mathbf{x}_i) \quad \text{s.t.} \sum_{i=1}^{N} A_i\mathbf{x}_i = c, \tag{1.1}$$

where $\mathbf{x}_i \in \mathbb{R}^{n_i}$, $A_i \in \mathbb{R}^{m\times n_i}$, $c \in \mathbb{R}^m$, and $f_i : \mathbb{R}^{n_i} \to (-\infty, +\infty]$ are closed proper convex functions, $i = 1, 2, \ldots, N$. If an individual block is subject to constraint $\mathbf{x}_i \in \mathcal{X}_i$, where $\mathcal{X}_i \subseteq \mathbb{R}^{n_i}$ is a nonempty closed convex set, it can be incorporated in the objective function f_i using the indicator function:

$$I_{\mathcal{X}_i}(\mathbf{x}_i) = \begin{cases} 0 & \text{if } \mathbf{x}_i \in \mathcal{X}_i, \\ +\infty & \text{otherwise.} \end{cases} \tag{1.2}$$

In this study, we do not impose any additional assumptions such as strict convexity or differentiability on the objective functions f_i.

Such optimization problems arise from a broad spectrum of applications including signal and image processing, compressive sensing, statistics and machine learning. See [1–4,15, 16,29,31,35] for example.

In this paper, we focus on *parallel and distributed* optimization algorithms—Jacobi ADMM (Algorithm 3) and Jacobi-Proximal ADMM (Algorithm 4) below – for solving (1.1). Since both of the objective function and constraints of (1.1) are sums of terms on individual $\mathbf{x}_i$'s (they are *separable*), the problem can be decomposed into N smaller subproblems, which is solved in a parallel and distributed manner.

1.1 Literature Review

A simple distributed algorithm for solving (1.1) is *dual decomposition* [13], which is essentially a *dual ascent method* or *dual subgradient method* [34] as follows. Consider the Lagrangian for problem (1.1):

$$\mathcal{L}(\mathbf{x}_1, \ldots, \mathbf{x}_N, \lambda) = \sum_{i=1}^{N} f_i(\mathbf{x}_i) - \lambda^\top \left(\sum_{i=1}^{N} A_i\mathbf{x}_i - c \right) \tag{1.3}$$

where $\lambda \in \mathbb{R}^m$ is the Lagrange multiplier or the dual variable. The method of dual decomposition iterates as follows: for $k \geq 1$,

$$\begin{cases} (\mathbf{x}_1^{k+1}, \mathbf{x}_2^{k+1}, \ldots, \mathbf{x}_N^{k+1}) = \arg\min_{\{\mathbf{x}_i\}} \mathcal{L}(\mathbf{x}_1, \ldots, \mathbf{x}_N, \lambda^k), \\ \lambda^{k+1} = \lambda^k - \alpha_k \left(\sum_{i=1}^{N} A_i\mathbf{x}_i^{k+1} - c \right), \end{cases} \tag{1.4}$$

where $\alpha_k > 0$ is a step-size. Since all the $\mathbf{x}_i$'s are separable in the Lagrangian function (1.3), the $\mathbf{x}$-update step reduces to solving N individual $\mathbf{x}_i$-subproblems:

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x_i}} f_i(\mathbf{x}_i) - \langle \lambda^k, A_i \mathbf{x}_i \rangle, \quad \text{for } i = 1, 2, \ldots, N, \tag{1.5}$$

and thus they can be carried out in parallel. With suitable choice of α_k and certain assumptions, dual decomposition is guaranteed to converge to an optimal solution [34]. However, the convergence of such subgradient method often tends to be slow in practice. Its convergence rate for general convex problems is $O(1/\sqrt{k})$. The dual smoothing method [30] can be applied under certain conditions and improve the rate to $O(1/k)$.

Another effective distributed approach is based on the *alternating direction method of multipliers* (ADMM). ADMM was introduced in [14,16] to solve the special case of problem (1.1) with two blocks of variables ($N = 2$). It utilizes the *augmented Lagrangian* for (1.1):

$$\mathcal{L}_\rho(\mathbf{x}_1, \ldots, \mathbf{x}_N, \lambda) = \sum_{i=1}^N f_i(\mathbf{x}_i) - \lambda^\top \left(\sum_{i=1}^N A_i \mathbf{x}_i - c \right) + \frac{\rho}{2} \left\| \sum_{i=1}^N A_i \mathbf{x}_i - c \right\|_2^2, \tag{1.6}$$

which incorporates a quadratic penalty of the constraints (with a parameter $\rho > 0$) into the Lagrangian. In each iteration, the augmented Lagrangian is minimized over $\mathbf{x}_1$ and $\mathbf{x}_2$ separately, one after the other, followed by a dual update for λ. The iterative scheme of ADMM is outlined below:

$$\begin{cases} \mathbf{x}_1^{k+1} = \arg\min_{\mathbf{x}_1} \mathcal{L}_\rho(\mathbf{x}_1, \mathbf{x}_2^k, \lambda^k), \\ \mathbf{x}_2^{k+1} = \arg\min_{\mathbf{x}_2} \mathcal{L}_\rho(\mathbf{x}_1^{k+1}, \mathbf{x}_2, \lambda^k), \\ \lambda^{k+1} = \lambda^k - \rho(A_1 \mathbf{x}_1^{k+1} + A_2 \mathbf{x}_2^{k+1} - c). \end{cases} \tag{1.7}$$

To solve the problem (1.1) with $N \geq 3$ using ADMM, one can first convert the multi-block problem into an equivalent two-block problem via variable splitting [2]:

$$\begin{aligned} \min_{\{\mathbf{x}_i\},\{\mathbf{z}_i\}} & \sum_{i=1}^N f_i(\mathbf{x}_i) + I_{\mathcal{Z}}(\mathbf{z}_1, \ldots, \mathbf{z}_N) \\ \text{s.t.} & A_i \mathbf{x}_i - \mathbf{z}_i = \frac{c}{N}, \quad \forall i = 1, 2, \ldots, N, \end{aligned} \tag{1.8}$$

where $I_{\mathcal{Z}}$ is a indicator function defined by (1.2), and the convex set $\mathcal{Z}$ is given by

$$\mathcal{Z} = \left\{ (\mathbf{z}_1, \ldots, \mathbf{z}_N) : \sum_{i=1}^N \mathbf{z}_i = 0 \right\}.$$

The variables in (1.8) can be grouped into two blocks: $\mathbf{x} := (\mathbf{x}_1, \ldots, \mathbf{x}_N)$ and $\mathbf{z} := (\mathbf{z}_1, \ldots, \mathbf{z}_N)$, so that ADMM can directly apply. The augmented Lagrangian for (1.8) is given by

$$\mathcal{L}_\rho(\mathbf{x}, \mathbf{z}, \lambda) = \sum_{i=1}^N f_i(\mathbf{x}_i) + I_{\mathcal{Z}}(\mathbf{z}) - \sum_{i=1}^N \lambda_i^\top \left(A_i \mathbf{x}_i - \mathbf{z}_i - \frac{c}{N} \right) + \frac{\rho}{2} \sum_{i=1}^N \left\| A_i \mathbf{x}_i - \mathbf{z}_i - \frac{c}{N} \right\|_2^2.$$

Since all the $\mathbf{x}_i$'s are now fully decoupled, the resulting $\mathbf{x}$-subproblem decomposes into N individual $\mathbf{x}_i$-subproblems, which can be carried out in parallel.

Algorithm 1: Variable Splitting ADMM (VSADMM)

```
Initialize $\mathbf{x}^0,\ \lambda^0,\ \rho > 0$;
for $k = 0,\ 1, \ldots$ do
    Update $\mathbf{z}_i$ then $\mathbf{x}_i$ for $i = 1, \ldots, N$ in parallel by:
    $\mathbf{z}_i^{k+1} = \left(A_i\mathbf{x}_i^k - \frac{c}{N} - \frac{\lambda_i^k}{\rho}\right) - \frac{1}{N}\left\{\sum_{j=1}^N A_j\mathbf{x}_j^k - \frac{c}{N} - \frac{\lambda_j^k}{\rho}\right\}$;
    $\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} f_i(\mathbf{x}_i) + \frac{\rho}{2}\left\|A_i\mathbf{x}_i - \mathbf{z}_i^{k+1} - \frac{c}{N} - \frac{\lambda_i^k}{\rho}\right\|_2^2$;
    Update $\lambda_i^{k+1} = \lambda_i^k - \rho\left(A_i\mathbf{x}_i^{k+1} - \mathbf{z}_i^{k+1} - \frac{c}{N}\right), \forall i = 1, \ldots, N$.
```

The resulting $\mathbf{z}$-subproblem is a simple quadratic problem:

$$\mathbf{z}^{k+1} = \arg\min_{\{\mathbf{z}:\sum_{i=1}^N \mathbf{z}_i=0\}} \sum_{i=1}^N \frac{\rho}{2}\left\|A_i\mathbf{x}_i - \mathbf{z}_i - \frac{c}{N} - \frac{\lambda_i^k}{\rho}\right\|_2^2, \tag{1.9}$$

which admits a closed-form solution.

The distributed ADMM approach based on (1.8), by introducing splitting variables, *substantially increases the number of variables and constraints* in the problem, especially when N is large. Thus, it is clear that this approach will not be very efficient. Another extension is to replace the two-block alternating minimization scheme by a sweep of Gauss–Seidel update, namely, update $\mathbf{x}_i$ for $i = 1, 2, \ldots, N$ sequentially as follows:

$$\begin{aligned}\mathbf{x}_i^{k+1} &= \arg\min_{\mathbf{x}_i} \mathcal{L}_\rho\left(\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_{i-1}^{k+1}, \mathbf{x}_i, \mathbf{x}_{i+1}^k, \ldots, \mathbf{x}_N^k, \lambda^k\right)\\ &= \arg\min_{\mathbf{x}_i}\ f_i(\mathbf{x}_i) + \frac{\rho}{2}\left\|\sum_{j<i} A_j\mathbf{x}_j^{k+1} + A_i\mathbf{x}_i + \sum_{j>i} A_j\mathbf{x}_j^k - c - \frac{\lambda^k}{\rho}\right\|_2^2.\end{aligned} \tag{1.10}$$

Algorithm 2: Gauss–Seidel ADMM

```
Initialize $\mathbf{x}^0,\ \lambda^0,\ \rho > 0$;
for $k = 0,\ 1, \ldots$ do
    Update $\mathbf{x}_i$ for $i = 1, \ldots, N$ sequentially by:
    $\mathbf{x}_i^{k+1} = \min_{\mathbf{x}_i} f_i(\mathbf{x}_i) + \frac{\rho}{2}\left\|\sum_{j<i} A_j\mathbf{x}_j^{k+1} + A_i\mathbf{x}_i + \sum_{j>i} A_j\mathbf{x}_j^k - c - \frac{\lambda^k}{\rho}\right\|_2^2$; Update
    $\lambda^{k+1} = \lambda^k - \rho\left(\sum_{i=1}^N A_i\mathbf{x}_i^{k+1} - c\right)$.
```

Such *Gauss–Seidel ADMM* (Algorithm 2) has been considered lately, e.g., in [22,25]. However, it has been shown that the algorithm may not converge for $N \geq 3$ [5]. Although lack of convergence guarantee, some empirical studies show that Algorithm 2 is still very effective at solving many practical problems (see, e.g., [31,35,36]). Recent work has shown that additional assumptions (involving at least one strongly convex objective function) can guarantee convergence for $N = 3$ with certain modifications to the algorithm [6,11,18,26,27]. A disadvantage of Gauss–Seidel ADMM is that the blocks are updated one after another, which is not amenable for parallelization.

1.2 Jacobi-Type ADMM

The sequential nature of Gauss–Seidel ADMM motivates us to consider using a Jacobi-type scheme that updates all the N blocks in parallel:

$$\begin{aligned}\mathbf{x}_i^{k+1} &= \arg\min_{\mathbf{x}_i} \mathcal{L}_\rho\left(\mathbf{x}_1^k, \dots, \mathbf{x}_{i-1}^k, \mathbf{x}_i, \mathbf{x}_{i+1}^k, \dots, \mathbf{x}_N^k, \lambda^k\right)\\ &= \arg\min_{\mathbf{x}_i} f_i(\mathbf{x}_i) + \frac{\rho}{2}\left\| A_i\mathbf{x}_i + \sum_{j\neq i} A_j\mathbf{x}_j^k - c - \frac{\lambda^k}{\rho}\right\|_2^2, \ \forall i = 1, \dots, N. \quad (1.11)\end{aligned}$$

We present it in Algorithm 3 below.

The parallelization comes with a cost: this scheme is more likely to diverge than the Gauss–Seidel scheme when using the same parameter ρ. In fact, it may diverge even in the two-block case; see [21] for such an example. To guarantee its convergence, either additional assumptions or modifications to the algorithm must be made.

Algorithm 3: Jacobi ADMM

```
Initialize $\mathbf{x}^0$, $\lambda^0$, $\rho > 0$;
for $k = 0, 1, \dots$ do
    Update $\mathbf{x}_i$ for $i = 1, \dots, N$ in parallel by:
    $\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} f_i(\mathbf{x}_i) + \frac{\rho}{2}\left\|A_i\mathbf{x}_i + \sum_{j\neq i} A_j\mathbf{x}_j^k - c - \frac{\lambda^k}{\rho}\right\|_2^2$;
    Update $\lambda^{k+1} = \lambda^k - \rho\left(\sum_{i=1}^N A_i\mathbf{x}_i^{k+1} - c\right)$.
```

In Sect. 4, we show that if matrices A_i are mutually near-orthogonal and have full column-rank, then Algorithm 3 converges globally. For general cases, a few variants of Jacobi ADMM with additional corrections were proposed in [20,21].

In this paper, we propose *Jacobi-Proximal ADMM* (Algorithm 4). Compared with Algorithm 3, we make no assumption to the problem data but add a proximal term $\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_i^k\|_{P_i}^2$ for each $\mathbf{x}_i$-subproblem and a damping parameter $\gamma > 0$ for the update of λ. Here $P_i \succeq 0$ is a symmetric and positive semi-definite matrix and we let $\|\mathbf{x}_i\|_{P_i}^2 := \mathbf{x}_i^\top P_i\mathbf{x}_i$. We will show how to choose P_i that ensure the convergence of Algorithm 4.

Algorithm 4: Jacobi-Proximal ADMM

```
Initialize: $\mathbf{x}_i^0$ $(i = 1, 2, \dots, N)$ and $\lambda^0$;
for $k = 0, 1, \dots$ do
    Update $\mathbf{x}_i$ for $i = 1, \dots, N$ in parallel by:
    $\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} f_i(\mathbf{x}_i) + \frac{\rho}{2}\left\|A_i\mathbf{x}_i + \sum_{j\neq i} A_j\mathbf{x}_j^k - c - \frac{\lambda^k}{\rho}\right\|_2^2 + \frac{1}{2}\left\|\mathbf{x}_i - \mathbf{x}_i^k\right\|_{P_i}^2$;
    Update $\lambda^{k+1} = \lambda^k - \gamma\rho(\sum_{i=1}^N A_i\mathbf{x}_i^{k+1} - c)$.
```

The proposed algorithm has a few advantages. First of all, as we will show, it enjoys global convergence as well as an $o(1/k)$ convergence rate of $\|\mathbf{x}^{k+1} - \mathbf{x}^k\|_M^2$ under certain conditions on P_i and γ. Secondly, when the $\mathbf{x}_i$-subproblem is not strictly convex, adding the proximal term can make the subproblem strictly or strongly convex, making it to have a unique solution. Thirdly, there are multiple choices for matrices P_i with which the subproblems become easier to solve. Specifically, the $\mathbf{x}_i$-subproblem contains a quadratic term $\frac{\rho}{2}\mathbf{x}_i^\top A_i^\top A_i\mathbf{x}_i$. When $A_i^\top A_i$ is ill-conditioned or computationally expensive to invert, one can let $P_i = D_i - \rho A_i^\top A_i$, which cancels the quadratic term $\frac{\rho}{2}\mathbf{x}_i^\top A_i^\top A_i\mathbf{x}_i$ and adds $\frac{1}{2}\mathbf{x}_i^\top D_i\mathbf{x}_i$, where the matrix D_i can be chosen as some well-conditioned and simple matrix (e.g., a diagonal matrix), thereby leading to an easier subproblem.

Let us mention two commonly used choices of P_i:

- $P_i = \tau_i\mathbf{I}$ ($\tau_i > 0$): This corresponds to the standard *proximal method*.
- $P_i = \tau_i\mathbf{I} - \rho A_i^\top A_i$ ($\tau_i > 0$): This corresponds to the *prox-linear method* [7], which linearizes the quadratic penalty term of augmented Lagrangian at the current point $\mathbf{x}_i^k$ and adds a proximal term. The $\mathbf{x}_i$-subproblem is given by

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} f_i(\mathbf{x}_i) + \left\langle \rho A_i^\top (A\mathbf{x}^k - c - \lambda^k/\rho), \mathbf{x}_i \right\rangle + \frac{\tau_i}{2}\left\|\mathbf{x}_i - \mathbf{x}_i^k\right\|^2. \quad (1.12)$$

  It essentially uses an identity matrix $\tau_i\mathbf{I}$ to approximate the Hessian $\rho A_i^\top A_i$ of the quadratic term.

More choices of P_i have also been discussed in [12,38].

1.3 Summary of Contributions

This paper introduces novel results from the following perspectives. Firstly, we propose *Jacobi-Proximal ADMM* (Algorithm 4), which is suitable for parallel and distributed computing.

The flexible use of proximal terms makes it possible to solve subproblems in different ways, important for easy coding and fast computation. We establish its convergence at a rate of $o(1/k)$ for $\|\mathbf{x}^{k+1} - \mathbf{x}^k\|_M^2$. We also provide an adaptive parameter tuning scheme which updates the important parameters at little extra cost. Our numerical results on the exchange problem and ℓ_1-minimization problem show that the algorithm achieves competitive performance.

The second contribution is a condition that guarantees the convergence of *Jacobi ADMM* (Algorithm 3). The condition is applied to the coefficient matrices A_i, without assumptions on the objective functions f_i or penalty parameter ρ.

A minor contribution is the improvement of the established convergence rate of $O(1/k)$ for the standard ADMM to $o(1/k)$. The improvement requires showing a certain sequence is monotonically nonincreasing, which can be applied to improve the existing rate of $O(1/k)$ of several other algorithms to $o(1/k)$, e.g. [21].

Finally we have experimented Algorithm 4 numerically based on the Amazon Elastic Computing Cloud (EC2) to demonstrate its effectiveness of solving a compressive sensing problem of huge size.

1.4 Notation, Assumptions and Preliminary Results

To simplify the notation in this paper, we introduce

$$\mathbf{x} := \begin{pmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_N \end{pmatrix} \in \mathbb{R}^{\mathbf{n}},\ \mathbf{A} := \big(A_1, \ldots, A_N\big) \in \mathbb{R}^{\mathbf{m}\times\mathbf{n}},\ \mathbf{u} := \begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix} \in \mathbb{R}^{\mathbf{n}+\mathbf{m}},$$

where $n = \sum_{i=1}^N n_i$. We let $\langle\cdot,\cdot\rangle$ and $\|\cdot\|$ denote the standard inner product and ℓ_2-norm $\|\cdot\|_2$, respectively, in the Euclidean space. For a matrix $M \in \mathbb{R}^{l\times l}$, $\|M\|$ denotes the spectral norm, i.e., the largest singular value of M. For a positive definite matrix $G \in \mathbb{R}^{l\times l}$, we define the *G-norm* as follows:

$$\|\mathbf{z}\|_G := \sqrt{\mathbf{z}^\top G\mathbf{z}}, \quad \forall \mathbf{z} \in \mathbb{R}^l. \tag{1.13}$$

If the matrix G is positive semi-definite, then $\|\cdot\|_G$ is a semi-norm.

Throughout the paper, we make the following standard assumptions.

Assumption 1 Functions $f_i : \mathbb{R}^{n_i} \to (-\infty, +\infty]$ $(i = 1, 2, \ldots, N)$ are closed proper convex.

Assumption 2 There exists a saddle point $\mathbf{u}^* = (\mathbf{x}_1^*, \mathbf{x}_2^*, \ldots, \mathbf{x}_N^*, \lambda^*)$ to the problem (1.1), that is, $\mathbf{u}^*$ satisfies the KKT conditions:

$$A_i^\top \lambda^* \in \partial f_i(\mathbf{x}_i^*), \text{ for } i = 1, \ldots, N, \tag{1.14a}$$

$$A\mathbf{x}^* = \sum_{i=1}^{N} A_i\mathbf{x}_i^* = c. \tag{1.14b}$$

The conditions (1.14a) and (1.14b) can be written in a more compact form using variational inequality [21]:

$$f(\mathbf{x}) - f(\mathbf{x}^*) + (\mathbf{u} - \mathbf{u}^*)^\top F(\mathbf{u}^*) \geq 0, \ \forall \mathbf{u}, \tag{1.15}$$

where $f(\mathbf{x}) := \sum_i f_i(\mathbf{x}_i)$ and $F(\mathbf{u}) := [-A_1^\top\lambda\,,\ \ldots, -A_N^\top\lambda\,,\ A\mathbf{x} - c]^\top$.

Let $\partial f_i(\mathbf{x}_i)$ denote the subdifferential of f_i at $\mathbf{x}_i$:

$$\partial f_i(\mathbf{x}_i) := \left\{ s_i \in \mathbb{R}^{n_i} : s_i^\top(\mathbf{y}_i - \mathbf{x}_i) \leq f_i(\mathbf{y}_i) - f_i(\mathbf{x}_i), \ \forall \mathbf{y}_i \in \operatorname{dom} f_i \right\}, \tag{1.16}$$

which is nonempty under Assumption 1. We recall that a subdifferential map of a convex function is monotone, which will be used several times in later sections.

Proposition 1.1 *Under Assumption 1, for any* $\mathbf{x}_i, \mathbf{y}_i \in \operatorname{dom} f_i$*, we have*

$$(s_i - t_i)^\top(\mathbf{x}_i - \mathbf{y}_i) \geq 0, \quad \forall s_i \in \partial f_i(\mathbf{x}_i),\ t_i \in \partial f_i(\mathbf{y}_i),\ i = 1, 2, \ldots, N. \tag{1.17}$$

In addition, we use an elementary argument to improve the convergence rate from $O(1/k)$ to $o(1/k)$. Intuitively, the harmonic sequence $1/k$ is not summable, so a summable, nonnegative, monotonic sequence shall converge faster than $1/k$.

Lemma 1.1 *If a sequence* $\{a_k\} \subseteq \mathbb{R}$ *obeys: (1)* $a_k \geq 0$*; (2)* $\sum_{k=1}^{\infty} a_k < +\infty$*; (3)* a_k *is monotonically non-increasing, then we have* $a_k = o(1/k)$.

Proof Since $k \cdot a_{2k} \leq a_{k+1} + a_{k+2} + \cdots + a_{2k} \to 0$ as $k \to +\infty$, $a_k = o(1/k)$. □

More study on the $o(1/k)$ rate, its tightness, and other results for splitting schemes including the standard ADMM can be found in [10].

1.5 Organization

The rest of the paper is organized as follows. In Sect. 2, we establish the convergence as well as an $o(1/k)$ rate of convergence for the Jacobi-Proximal ADMM (Algorithm 4). We also propose a practical parameter tuning method, which updates important parameters on the fly at little extra cost and makes the algorithm run faster. In Sect. 3, we present several numerical results to demonstrate the efficiency of Algorithm 4 in comparison with some existing parallel algorithms. In Sect. 4, we show that the Jacobi ADMM (Algorithm 3), though lack of convergence in the general case, will still converge under certain conditions on the matrices A_i. It is interesting to point out that the Jacobi-Proximal ADMM can be derived based on the classical proximal point method(PPA). See Sect. 5. Finally, we conclude the paper in Sect. 6. For convenience, we include some detail on how to use Lemma 1.1 to improve some existing convergence rates from $O(1/k)$ to $o(1/k)$ in "Appendix".

2 Convergence Analysis of the Jacobi-Proximal ADMM

In this section, we study the convergence of Jacobi-Proximal ADMM (Algorithm 4). We first show its convergence and establish an $o(1/k)$ convergence rate for a quantity that is also used in [24]. Furthermore, we show how to automatically update the parameter in order to make Jacobi-Proximal ADMM more practical.

2.1 Convergence

To simplify the notation, we let

$$G_x := \begin{pmatrix} P_1 + \rho A_1^\top A_1 & & \\ & \ddots & \\ & & P_N + \rho A_N^\top A_N \end{pmatrix}, \quad G := \begin{pmatrix} G_x & \\ & \frac{1}{\gamma\rho}\mathbf{I} \end{pmatrix},$$

and

$$Q := \begin{pmatrix} P_1 + \rho A_1^\top A_1 & & & \frac{1}{\gamma} A_1^\top \\ & \ddots & & \vdots \\ & & P_N + \rho A_N^\top A_N & \frac{1}{\gamma} A_N^\top \\ \frac{1}{\gamma} A_1 & \dots & \frac{1}{\gamma} A_N & \frac{2-\gamma}{\rho\gamma^2}\mathbf{I} \end{pmatrix}, \tag{2.1}$$

where $\mathbf{I}$ is the identity matrix of size $m \times m$. In the rest of the section, we let $\{\mathbf{u}^k = (\mathbf{x}_1^k, \dots, \mathbf{x}_N^k, \lambda^k), k \geq 1\}$ denote the sequence generated by Jacobi-Proximal ADMM from any initial point. The analysis is based on bounding the error $\|\mathbf{u}^k - \mathbf{u}^*\|_G^2$ and estimating its decrease, motivated by the works [12,21,23].

Lemma 2.1 *For $k \geq 1$, we have*

$$\left\|\mathbf{u}^k - \mathbf{u}^*\|_G^2 - \|\mathbf{u}^{k+1} - \mathbf{u}^*\right\|_G^2 \geq \left\|\mathbf{u}^k - \mathbf{u}^{k+1}\right\|_Q^2, \tag{2.2}$$

where $\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_Q^2 := \|\mathbf{x}^k - \mathbf{x}^{k+1}\|_{G_x}^2 + \frac{2-\gamma}{\rho\gamma^2}\|\lambda^k - \lambda^{k+1}\|^2 + \frac{2}{\gamma}(\lambda^k - \lambda^{k+1})^\top A(\mathbf{x}^k - \mathbf{x}^{k+1})$.

Proof Recall that in Algorithm 4, we solve the following $\mathbf{x}_i$-subproblem:

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} f_i(\mathbf{x}_i) + \frac{\rho}{2}\left\|A_i\mathbf{x}_i + \sum_{j\neq i} A_j\mathbf{x}_j^k - c - \frac{\lambda^k}{\rho}\right\|^2 + \frac{1}{2}\left\|\mathbf{x}_i - \mathbf{x}_i^k\right\|_{P_i}^2.$$

Its optimality condition is given by

$$A_i^\top(\lambda^k - \rho\left(A_i\mathbf{x}_i^{k+1} + \sum_{j\neq i} A_j\mathbf{x}_j^k - c)\right) + P_i\left(\mathbf{x}_i^k - \mathbf{x}_i^{k+1}\right) \in \partial f_i\left(\mathbf{x}_i^{k+1}\right). \tag{2.3}$$

For convenience, we let $\hat{\lambda} := \lambda^k - \rho\left(A\mathbf{x}^{k+1} - c\right)$. Then (2.3) can be rewritten as

$$A_i^\top(\hat{\lambda} - \rho\sum_{j\neq i} A_j\left(\mathbf{x}_j^k - \mathbf{x}_j^{k+1})\right) + P_i\left(\mathbf{x}_i^k - \mathbf{x}_i^{k+1}\right) \in \partial f_i\left(\mathbf{x}_i^{k+1}\right). \tag{2.4}$$

By Lemma 1.1, it follows from (1.14a) and (2.4) that

$$\langle A_i\left(\mathbf{x}_i^{k+1} - \mathbf{x}_i^*\right),\ \hat{\lambda} - \lambda^* - \rho\sum_{j\neq i} A_j\left(\mathbf{x}_j^k - \mathbf{x}_j^{k+1}\right)\rangle + (\mathbf{x}_i^{k+1} - \mathbf{x}_i^*)^\top P_i(\mathbf{x}_i^k - \mathbf{x}_i^{k+1}) \geq 0.$$

Summing the above inequality over all i and using the following equality for each i:

$$\sum_{j\neq i} A_j\left(\mathbf{x}_j^k - \mathbf{x}_j^{k+1}\right) = A\left(\mathbf{x}^k - \mathbf{x}^{k+1}\right) - A_i\left(\mathbf{x}_i^k - \mathbf{x}_i^{k+1}\right),$$

we obtain

$$\begin{aligned}&\langle A\left(\mathbf{x}^{k+1} - \mathbf{x}^*\right), \hat{\lambda} - \lambda^*\rangle + \sum_{i=1}^N(\mathbf{x}_i^{k+1} - \mathbf{x}_i^*)^\top\left(P_i + \rho A_i^\top A_i\right)\left(\mathbf{x}_i^k - \mathbf{x}_i^{k+1}\right)\\ &\geq \rho\langle A\left(\mathbf{x}^{k+1} - \mathbf{x}^*\right), A\left(\mathbf{x}^k - \mathbf{x}^{k+1}\right)\rangle.\end{aligned} \tag{2.5}$$

Note that $A\left(\mathbf{x}^{k+1} - \mathbf{x}^*\right) = \frac{1}{\gamma\rho}\left(\lambda^k - \lambda^{k+1}\right)$ and

$$\hat{\lambda} - \lambda^* = \left(\hat{\lambda} - \lambda^{k+1}\right) + \left(\lambda^{k+1} - \lambda^*\right) = \frac{\gamma - 1}{\gamma}\left(\lambda^k - \lambda^{k+1}\right) + \left(\lambda^{k+1} - \lambda^*\right).$$

With the above two equations, the inequality (2.5) can be rewritten as

$$\begin{aligned}&\langle\frac{1}{\gamma\rho}\left(\lambda^k - \lambda^{k+1}\right), \lambda^{k+1} - \lambda^*\rangle + \sum_{i=1}^N\left(\mathbf{x}_i^{k+1} - \mathbf{x}_i^*\right)^\top\left(P_i + \rho A_i^\top A_i\right)\left(\mathbf{x}_i^k - \mathbf{x}_i^{k+1}\right)\\ &\geq \frac{1-\gamma}{\gamma^2\rho}\left\|\lambda^k - \lambda^{k+1}\right\|^2 + \frac{1}{\gamma}\left(\lambda^k - \lambda^{k+1})^\top A(\mathbf{x}^k - \mathbf{x}^{k+1}\right),\end{aligned} \tag{2.6}$$

or more compactly,

$$\begin{aligned}\left(\mathbf{u}^k - \mathbf{u}^{k+1}\right)^\top G\left(\mathbf{u}^{k+1} - \mathbf{u}^*\right) \geq \frac{1-\gamma}{\gamma^2\rho}&\left\|\lambda^k - \lambda^{k+1}\right\|^2\\ &+\frac{1}{\gamma}\left(\lambda^k - \lambda^{k+1}\right)^\top A\left(\mathbf{x}^k - \mathbf{x}^{k+1}\right).\end{aligned} \tag{2.7}$$

Since $\|\mathbf{u}^k - \mathbf{u}^*\|_G^2 - \|\mathbf{u}^{k+1} - \mathbf{u}^*\|_G^2 = 2(\mathbf{u}^k - \mathbf{u}^{k+1})^\top G(\mathbf{u}^{k+1} - \mathbf{u}^*) + \|\mathbf{u}^k - \mathbf{u}^{k+1}\|_G^2$, using the above inequality (2.7) yields (2.2) immediately. □

If the matrix Q is positive definite, there exists some $\eta > 0$ such that

$$\left\|\mathbf{u}^k - \mathbf{u}^{k+1}\right\|_Q^2 \geq \eta\cdot\|\mathbf{u}^k - \mathbf{u}^{k+1}\|^2 \geq 0. \tag{2.8}$$

Then Lemma 2.1 indicates that

$$\|\mathbf{u}^k - \mathbf{u}^*\|_G^2 - \|\mathbf{u}^{k+1} - \mathbf{u}^*\|_G^2 \geq \eta\cdot\|\mathbf{u}^k - \mathbf{u}^{k+1}\|^2, \tag{2.9}$$

i.e., the iterative sequence $\{\mathbf{u}^k\}$ is *strictly contractive*. In particular, the error $\|\mathbf{u}^k - \mathbf{u}^*\|_G^2$ is monotonically non-increasing and thus converging, as well as $\|\mathbf{u}^k - \mathbf{u}^{k+1}\|^2 \to 0$. Then the convergence of the algorithm ($\|\mathbf{u}^k - \mathbf{u}^*\|_G^2 \to 0$) follows immediately from the standard analysis for contraction methods (see, e.g., [19]). We omit the details of the proof for the sake of brevity and state our convergence theorem as follows.

Theorem 2.1 *Under Assumptions 1 and 2, one can find parameters in Algorithm 4 such that its sequence $\{\mathbf{u}^k\}$ converges to a solution $\mathbf{u}^*$ to the problem (1.1). Specifically, if one chooses some $\epsilon_i > 0$ such that the parameters ρ, γ and P_i $(i = 1, 2, \dots, N)$ satisfy the following condition:*

$$\begin{cases} P_i \succ \rho(\frac{1}{\epsilon_i} - 1)A_i^\top A_i, \ i = 1, 2, \dots, N \\ \sum_{i=1}^N \epsilon_i < 2 - \gamma, \end{cases} \tag{2.10}$$

then Q in (2.1) is positive definite and $\{\mathbf{u}^k\}$ converges to $\mathbf{u}^*$.

Furthermore, by letting each $\epsilon_i < \frac{2-\gamma}{N}$, the condition (2.10) can be simplified to

$$P_i \succ \rho\left(\frac{N}{2-\gamma} - 1\right) A_i^\top A_i, \ i = 1, 2, \dots, N. \tag{2.11}$$

For special choices of P_i:

- *$P_i = \tau_i \mathbf{I}$ (standard proximal), condition (2.11) reduces to $\tau_i > \rho\left(\frac{N}{2-\gamma} - 1\right)\|A_i\|^2$;*
- *$P_i = \tau_i \mathbf{I} - \rho A_i^\top A_i$ (prox-linear), condition (2.11) reduces to $\tau_i > \frac{\rho N}{2-\gamma}\|A_i\|^2$.*

Note that one can normalize A_i so that $\|A_i\| \le 1$ and further simplify the above sufficient conditions.

Proof For any $\mathbf{u} = (\mathbf{x}; \lambda) \in \mathbb{R}^{n+m}$, we have

$$\|\mathbf{u}\|_Q^2 := \|\mathbf{x}\|_{G_x}^2 + \frac{2-\gamma}{\rho\gamma^2}\|\lambda\|^2 + \frac{2}{\gamma}\lambda^\top A\mathbf{x}. \tag{2.12}$$

Using the following basic inequality:

$$\frac{2}{\gamma}\lambda^\top A\mathbf{x} = \sum_{i=1}^N \frac{2}{\gamma}\lambda^\top A_i\mathbf{x}_i \ge -\sum_{i=1}^N \left(\frac{\epsilon_i}{\rho\gamma^2}\|\lambda\|^2 + \frac{\rho}{\epsilon_i}\|A_i\mathbf{x}_i\|^2\right), \tag{2.13}$$

for any $\epsilon_i > 0$ $(i = 1, 2, \dots, N)$, we have

$$\|\mathbf{u}\|_Q^2 \ge \sum_{i=1}^N \|\mathbf{x}_i\|^2_{P_i + \rho A_i^\top A_i - \frac{\rho}{\epsilon_i}A_i^\top A_i} + \frac{2-\gamma-\sum_{i=1}^N \epsilon_i}{\rho\gamma^2}\|\lambda\|^2. \tag{2.14}$$

The condition (2.10) guarantees that $P_i + \rho A_i^\top A_i - \frac{\rho}{\epsilon_i}A_i^\top A_i \succ 0$ and $2 - \gamma - \sum_{i=1}^N \epsilon_i > 0$, and thus $\|\mathbf{u}\|_Q > 0$. Hence, Q is positive definite. The rest follows immediately. □

2.2 Rate of Convergence

Next, we shall establish the $o(1/k)$ convergence rate of Jacobi-Proximal ADMM. We use the quantity $\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_M^2$ as a measure of the convergence rate motivated by [21,24]. Here, we define the matrix M by

$$M := \begin{pmatrix} M_x & \\ & \frac{1}{\gamma\rho}\mathbf{I} \end{pmatrix} \quad \text{and} \quad M_x := G_x - \rho A^\top A.$$

Theorem 2.2 *If $Q \succ 0$ and $M_x \succeq 0$, then $\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_M^2 = o(1/k)$. Hence, $\|\mathbf{x}^k - \mathbf{x}^{k+1}\|_{M_x}^2 = o(1/k)$ and $\|\lambda^k - \lambda^{k+1}\|^2 = o(1/k)$.*

We need the following monotonic property of the iterations:

Lemma 2.2 *If $M_x \succeq 0$ and $0 < \gamma < 2$, then $\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_M^2 \le \|\mathbf{u}^{k-1} - \mathbf{u}^k\|_M^2$.*

Proof Let $\Delta\mathbf{x}_i^{k+1} = \mathbf{x}_i^k - \mathbf{x}_i^{k+1}$, $i = 1, \dots, N$, $\Delta\mathbf{x}^{k+1} = \mathbf{x}^k - \mathbf{x}^{k+1}$, and $\Delta\lambda^{k+1} = \lambda^k - \lambda^{k+1}$. By Lemma 1.1, the optimality conditions (2.4) at k-th and $(k+1)$-th iterations yield

$$\left\langle A_i \Delta\mathbf{x}_i^{k+1}, \Delta\lambda^k - \rho A \Delta\mathbf{x}^{k+1} - \rho \sum_{j \neq i} A_j (\Delta\mathbf{x}_j^k - \Delta\mathbf{x}_j^{k+1}) \right\rangle$$
$$+ (\Delta\mathbf{x}_i^{k+1})^\top P_i \left(\Delta\mathbf{x}_i^k - \Delta\mathbf{x}_i^{k+1}\right) \ge 0.$$

Summing up over all i and rearranging the terms, we have

$$\left\langle A \Delta\mathbf{x}^{k+1}, \Delta\lambda^k \right\rangle \ge \left\|\Delta\mathbf{x}^{k+1}\right\|_{G_x}^2 - (\Delta\mathbf{x}^k)^\top \left(G_x - \rho A^\top A\right) \Delta\mathbf{x}^{k+1}. \tag{2.15}$$

Since $M_x := G_x - \rho A^\top A \succeq 0$, we have $2(\Delta\mathbf{x}^k)^\top (G_x - \rho A^\top A) \Delta\mathbf{x}^{k+1} \le \|\Delta\mathbf{x}^k\|_{M_x}^2 + \|\Delta\mathbf{x}^{k+1}\|_{M_x}^2$, and thus

$$2\langle A\Delta\mathbf{x}^{k+1}, \Delta\lambda^k\rangle \ge \|\Delta\mathbf{x}^{k+1}\|_{2G_x - M_x}^2 - \|\Delta\mathbf{x}^k\|_{M_x}^2 = \|\Delta\mathbf{x}^{k+1}\|_{G_x + \rho A^\top A}^2 - \|\Delta\mathbf{x}^k\|_{M_x}^2.$$

Note that $\Delta\lambda^{k+1} = \Delta\lambda^k - \gamma\rho A \Delta\mathbf{x}^{k+1}$. It follows that

$$\begin{aligned} \frac{1}{\gamma\rho}\|\Delta\lambda^k\|^2 - \frac{1}{\gamma\rho}\|\Delta\lambda^{k+1}\|^2 &= 2\langle A\Delta\mathbf{x}^{k+1}, \Delta\lambda^k\rangle - \gamma\rho\|A\Delta\mathbf{x}^{k+1}\|^2 \\ &\ge \|\Delta\mathbf{x}^{k+1}\|_{G_x + (1-\gamma)\rho A^\top A}^2 - \|\Delta\mathbf{x}^k\|_{M_x}^2, \end{aligned}$$

i.e.,

$$\left(\|\Delta\mathbf{x}^k\|_{M_x}^2 + \frac{1}{\gamma\rho}\|\Delta\lambda^k\|^2\right) - \left(\|\Delta\mathbf{x}^{k+1}\|_{M_x} + \frac{1}{\gamma\rho}\|\Delta\lambda^{k+1}\|^2\right) \ge \|\Delta\mathbf{x}^{k+1}\|_{(2-\gamma)\rho A^\top A}^2 \ge 0,$$

which completes the proof. □

Proof of Theorem 2.2 By Theorem 2.1, there exists some $\eta > 0$ such that

$$\left\|\mathbf{u}^k - \mathbf{u}^*\right\|_G^2 - \left\|\mathbf{u}^{k+1} - \mathbf{u}^*\right\|_G^2 \ge \left\|\mathbf{u}^k - \mathbf{u}^{k+1}\right\|_Q^2 \ge \eta \left\|\mathbf{u}^k - \mathbf{u}^{k+1}\right\|_M^2. \tag{2.16}$$

Summing (2.16) over k gives $\sum_{k=1}^\infty \|\mathbf{u}^k - \mathbf{u}^{k+1}\|_M^2 < \infty$. On the other hand, Lemma 2.2 implies the monotone non-increasing of $\left\|\mathbf{u}^k - \mathbf{u}^{k+1}\right\|_M^2$. By Lemma 1.1, we have $\left\|\mathbf{u}^k - \mathbf{u}^{k+1}\right\|_M^2 = o(1/k)$. □

2.3 Adaptive Parameter Tuning

The parameters satisfying the condition (2.10) may be rather conservative, because the inequality (2.13) for bounding $\|\mathbf{u}\|_Q^2$ is usually very loose. In practice, we can compute $\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_Q^2$ exactly at very little extra cost. If $\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_Q^2 > 0$, then Lemma 2.1 assures the decreasing of the solution error (in the G-norm) so that the current parameters are acceptable. On the other hand, if $\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_Q^2 < 0$, then the matrix Q is not positive definite, meaning that the current parameters P_i, $i = 1, 2, \ldots, N$ may be too small. So we should make $\{P_i\}$ bigger until $\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_Q^2 > 0$ holds. Therefore, we propose a practical strategy for adaptively adjusting the matrices $\{P_i\}$ based on the value of $\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_Q^2$:

```
Initialize with small $P_i^0 \succeq 0$ $(i = 1, 2, \ldots, N)$ and a small $\eta > 0$;
for $k = 1, 2, \ldots$ do
    if $\|\mathbf{u}^{k-1} - \mathbf{u}^k\|_Q^2 > \eta \cdot \|\mathbf{u}^{k-1} - \mathbf{u}^k\|^2$ then
        $P_i^{k+1} \leftarrow P_i^k, \forall i$;
    else
        Increase $P_i$: $P_i^{k+1} \leftarrow \alpha_i P_i^k + \beta_i Q_i$ $(\alpha_i > 1,\ \beta_i \ge 0,\ Q_i \succ 0), \forall i$;
        Restart: $\mathbf{u}^k \leftarrow \mathbf{u}^{k-1}$;
```

The above strategy starts with relatively small proximal parameters $\{P_i\}$ and gradually increase them. By Theorem 2.1, we know that when the parameters $\{P_i\}$ are large enough for (2.10) to hold, the condition (2.8) will be satisfied (for sufficiently small η). Therefore, the adjustment of $\{P_i\}$ cannot occur infinite times. After a finite number of iterations, $\{P_i\}$ will remain constant and the contraction property (2.9) of the iterations will hold. Therefore, the convergence of such an adaptive parameter tuning scheme follows immediately from our previous analysis.

Theorem 2.3 *When the matrices P_i $(i = 1, 2, \ldots, N)$ in Algorithm 4 are adaptively adjusted in the above scheme, the algorithm converges to a solution to the problem* (1.1).

Empirical evidence shows that the parameters $\{P_i\}$ typically adjust themselves only during the first few iterations and then stay constant. Alternatively, one may also decrease the parameters after every few iterations or after they have not been updated for a certain number of iterations. But the total times of decrease should be bounded to guarantee convergence. By using this adaptive strategy, the resulting parameters $\{P_i\}$ are usually much smaller than those required by the condition (2.10), thereby leading to substantially faster convergence in practice.

3 Numerical Experiments

In this section, we present numerical results to compare the following parallel splitting algorithms:

- *Prox-JADMM* proposed Jacobi-Proximal ADMM (Algorithm 4);
- *VSADMM* Variable Splitting ADMM (Algorithm 1);
- *Corr-JADMM* Jacobi ADMM with correction steps [21]. At every iteration, it first generates a "predictor" $\tilde{\mathbf{u}}^{k+1}$ by an iteration of Jacobi ADMM (Algorithm 3) and then corrects $\tilde{\mathbf{u}}^{k+1}$ to generate the new iterate by:

$$\mathbf{u}^{k+1} = \mathbf{u}^k - \alpha_k(\mathbf{u}^k - \tilde{\mathbf{u}}^{k+1}), \tag{3.1}$$

  where $\alpha_k > 0$ is a step size. In our experiments, we adopt the dynamically updated step size α_k according to [21], which is shown to converge significantly faster than using a constant step size, though updating the step size requires extra computation.

– *YALL1* one of the state-of-the-art solvers for the ℓ_1-minimization problem.

In Sects. 3.1 and 3.2, all of the numerical experiments are run in MATLAB (R2011b) on a workstation with an Intel Core i5-3570 CPUs (3.40GHz) and 32 GB of RAM. Section 3.3 gives two very large instances that are solved by a C/MPI implementation on Amazon Elastic Compute Cloud (EC2).

3.1 Exchange Problem

Consider a network of N agents that exchange n commodities. Let $\mathbf{x}_i \in \mathbb{R}^n$ $(i = 1, 2, \ldots, N)$ denote the amount of commodities that are exchanged among the N agents. Each agent i has a certain cost function $f_i : \mathbb{R}^n \to \mathbb{R}$. The exchange problem (see, e.g., [3] for a review) is given by

$$\min_{\{\mathbf{x}_i\}} \sum_{i=1}^{N} f_i(\mathbf{x}_i) \quad \text{s.t.} \sum_{i=1}^{N} \mathbf{x}_i = 0, \tag{3.2}$$

which minimizes the total cost among N agents subject to an equilibrium constraint on the commodities. This is a special case of (1.1) where $A_i = \mathbf{I}$ and $c = 0$.

We consider quadratic cost functions $f_i(\mathbf{x}_i) := \frac{1}{2}\|C_i\mathbf{x}_i - d_i\|^2$, where $C_i \in \mathbb{R}^{p\times n}$ and $d_i \in \mathbb{R}^p$. All the compared algorithms solve the following type of subproblem:

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} \frac{1}{2}\|C_i\mathbf{x}_i - d_i\|^2 + \frac{\rho}{2}\|\mathbf{x}_i - b_i^k\|^2, \ \forall i = 1, 2, \ldots, N, \tag{3.3}$$

except that Prox-JADMM also adds a proximal term $\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_i^k\|_{P_i}^2$. Here $b_i^k \in \mathbb{R}^m$ is a vector independent of $\mathbf{x}_i$ and takes different forms in different algorithms. For Prox-JADMM, we simply set $P_i = \tau_i \mathbf{I}$ $(\tau_i > 0)$. Clearly, each $\mathbf{x}_i$-subproblem is a quadratic program that can be computed efficiently using various methods.

In our experiment, we randomly generate $\mathbf{x}_i^*$, $i = 1, 2, \ldots, N-1$, following the standard Gaussian distribution, and let $\mathbf{x}_N^* = -\sum_{i=1}^{N-1}\mathbf{x}_i^*$. Matrices C_i are random Gaussian matrices, and vectors d_i are computed by $d_i = C_i\mathbf{x}_i^*$. Apparently, $\mathbf{x}^*$ is a solution (not necessarily unique) to (3.2), and the optimal objective value is 0.

The penalty parameter ρ is set to be 0.01, 1 and 0.01 for Prox-JADMM, VSADMM and Corr-JADMM, respectively. They are nearly optimal for each algorithm, picked out of a number of different values. Note that the parameter for VSADMM is quite different from the other two algorithms because it has different constraints due to the variable splitting. For Prox-JADMM, the proximal parameters are initialized by $\tau_i = 0.1(N-1)\rho$ and adaptively updated by the strategy in Sect. 2.3; the parameter γ is set to be 1.

The size of the test problem is set to be $n = 100$, $N = 100$, $p = 80$. Letting all the algorithms run 200 iterations, we plot their objective value $\sum_{i=1}^{N} f_i(\mathbf{x}_i)$ and residual $\|\sum_{i=1}^{N}\mathbf{x}_i\|_2$. Note that the per-iteration cost (both computation and communication) is roughly the same for all the compared algorithms. Figure 1 shows the comparison result, which is averaged over 100 random trials. We can see that Prox-JADMM is clearly the fastest one among the compared algorithms.

3.2 ℓ_1-Minimization

We consider the ℓ_1-minimization problem for finding sparse solutions of an underdetermined linear system:

$$\min_{\mathbf{x}} \|\mathbf{x}\|_1 \quad \text{s.t. } A\mathbf{x} = c, \tag{3.4}$$

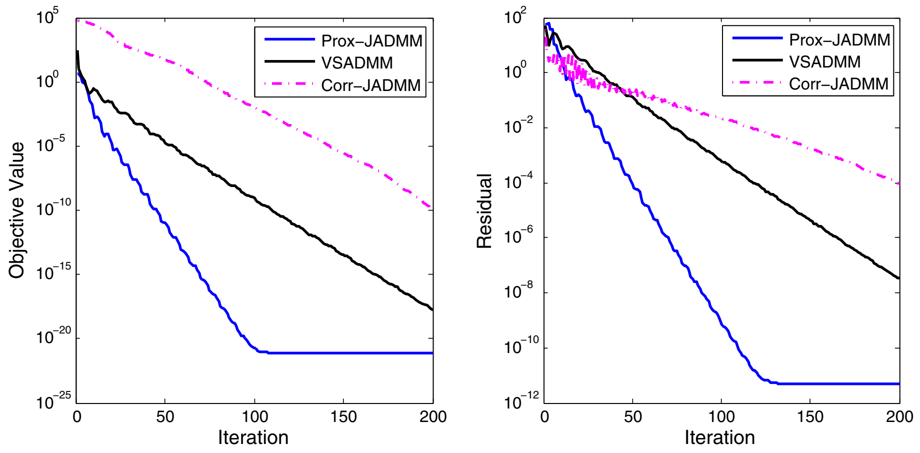

Fig. 1 Exchange problem ($n = 100,\ N = 100,\ p = 80$)

where $\mathbf{x} \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$ and $c \in \mathbb{R}^m$ $(m < n)$. It is also known as the *basis pursuit* problem, which has been widely used in compressive sensing, signal and image processing, statistics, and machine learning. Suppose that the data is partitioned into N blocks: $\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$ and $A = [A_1, A_2, \ldots, A_N]$. Then the problem (3.4) can be written in the form of (1.1) with $f_i(\mathbf{x}_i) = \|\mathbf{x}_i\|_1$.

In our experiment, a sparse solution $\mathbf{x}^*$ is randomly generated with k $(k \ll n)$ nonzeros drawn from the standard Gaussian distribution. Matrix A is also randomly generated from the standard Gaussian distribution, and it is partitioned evenly into N blocks. The vector c is then computed by $c = A\mathbf{x}^* + \eta$, where $\eta \sim \mathcal{N}(0,\ \sigma^2\mathbf{I})$ is Gaussian noise with standard deviation σ.

Prox-JADMM solves the $\mathbf{x}_i$-subproblems with $P_i = \tau_i\mathbf{I} - \rho A_i^\top A_i$ $(i = 1, 2, \ldots, N)$ as follows:

$$\begin{aligned}\mathbf{x}_i^{k+1} &= \arg\min_{\mathbf{x}_i}\ \|\mathbf{x}_i\|_1 + \frac{\rho}{2}\left\|A_i\mathbf{x}_i + \sum_{j\neq i} A_j\mathbf{x}_j^k - c - \frac{\lambda^k}{\rho}\right\|^2 + \frac{1}{2}\left\|\mathbf{x}_i - \mathbf{x}_i^k\right\|_{P_i}^2 \\ &= \arg\min_{\mathbf{x}_i}\ \|\mathbf{x}_i\|_1 + \left\langle \rho A_i^\top\left(A\mathbf{x}^k - c - \frac{\lambda^k}{\rho}\right), \mathbf{x}_i\right\rangle + \frac{\tau_i}{2}\left\|\mathbf{x}_i - \mathbf{x}_i^k\right\|^2. \end{aligned} \tag{3.5}$$

Here, we choose the prox-linear P_i's to linearize the original subproblems, and thus (3.5) admits a simple closed-form solution by the *shrinkage* (or *soft-thresholding*) formula. The proximal parameters are initialized as $\tau_i = 0.1N\rho$ and are adaptively updated by the strategy discussed in Sect. 2.3.

Recall that VSADMM needs to solve the following $\mathbf{x}_i$-subproblems:

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i}\ \|\mathbf{x}_i\|_1 + \frac{\rho}{2}\left\|A_i\mathbf{x}_i - \mathbf{z}_i^{k+1} - \frac{c}{N} - \frac{\lambda_i^k}{\rho}\right\|^2. \tag{3.6}$$

Such subproblems are not easily computable, unless $\mathbf{x}_i$ is a scalar (i.e., $n_i = 1$) or $A_i^\top A_i$ is a diagonal matrix. Instead, we solve the subproblems approximately using the prox-linear approach:

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} \|\mathbf{x}_i\|_1 + \left\langle \rho A_i^\top \left(A_i\mathbf{x}_i^k - \mathbf{z}_i^{k+1} - \frac{c}{N} - \frac{\lambda_i^k}{\rho} \right), \mathbf{x}_i - \mathbf{x}_i^k \right\rangle + \frac{\tau_i}{2} \left\| \mathbf{x}_i - \mathbf{x}_i^k \right\|^2,$$

which can be easily computed by the shrinkage operator. We set $\tau_i = 1.01\rho\|A_i\|^2$ in order to guarantee the convergence, as suggested in [36].

Corr-JADMM solves the following $\mathbf{x}_i$-subproblems in the "prediction" step:

$$\tilde{\mathbf{x}}_i^{k+1} = \arg\min_{\mathbf{x}_i} \|\mathbf{x}_i\|_1 + \frac{\rho}{2} \left\| A_i\mathbf{x}_i + \sum_{j\neq i} A_j\mathbf{x}_j^k - c - \frac{\lambda^k}{\rho} \right\|^2. \tag{3.7}$$

Because the correction step in [21] is based on exact minimization of the subproblems, we do not apply the prox-linear approach to solve the subproblems approximately. Instead, we always partition $\mathbf{x}$ into scalar components (i.e., $N = n$) so that the subproblems (3.7) can still be computed exactly. The same penalty parameter $\rho = 10/\|c\|_1$ is used for the three algorithms. It is nearly optimal for each algorithm, selected out of a number of different values.

We also include the YALL1 package [37] in the experiment, which is one of the state-of-the-art solvers for ℓ_1 minimization. Though YALL1 is not implemented in parallel, the matrix-vector multiplication by A and $A^\top$ in the code can be parallelized (see [32]). Since all the compared algorithms have roughly the same amount of per-iteration cost (in terms of both computation and communication), we simply let all the algorithms run for a fixed number of iterations and plot their relative error $\frac{\|\mathbf{x}^k - \mathbf{x}^*\|_2}{\|\mathbf{x}^*\|_2}$.

Figure 2 shows the comparison result where $n = 1000$, $m = 300$, $k = 60$ and the standard deviation of noise σ is set to be 0 and 10^{-3}, respectively. For Prox-JADMM and VSADMM, we set $N = 100$; for Corr-JADMM, we set $N = 1000$. The results are average of 100 random trials. We can see that Prox-JADMM and Corr-JADMM achieve very close performance and are the fastest ones among the compared algorithms. YALL1 also shows competitive performance. However, VSADMM is far slower than the others, probably due to inexact minimization of the subproblems and the conservative proximal parameters.

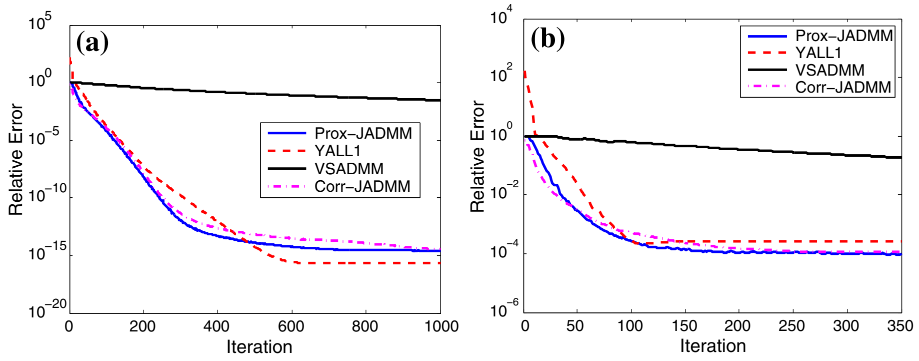

Fig. 2 ℓ_1-problem ($n = 1000,\ m = 300,\ k = 60$). **a** Noise-free ($\sigma = 0$), **b** noise added ($\sigma = 10^{-3}$)

Table 1 Two large datasets

	m	n	RAM
dataset 1	1.0×10^5	2.0×10^5	150 GB
dataset 2	1.5×10^5	3.0×10^5	337 GB

3.3 Distributed Large-Scale ℓ_1-Minimization

We now use Algorithm 4 to solve two very large instances of the ℓ_1-minimization problem (3.4) using a C code with MPI (for inter-process communication) and the GNU Scientific Library (GSL) (for BLAS operations). The experiments were carried out on Amazon EC2.

We generate two test instances as shown in Table 1. Specifically, a sparse solution $\mathbf{x}^*$ is randomly generated with 5% and 15% nonzeros drawn from the standard Gaussian distribution. Matrix A is also randomly generated from the standard Gaussian distribution with m rows and n columns, and it is partitioned evenly into $N = 80$ blocks. Vector c is then computed by $c = A\mathbf{x}^*$. Note that A is dense and has double precision. For Test 1 it requires over 150 GB of RAM and has 20 billion nonzero entries, and for Test 2 it requires over 337 GB of RAM. Those two tests are far too large to process on a single PC or workstation. We want to point out that we cannot find a dataset of similar or larger size in the public domain. We are willing to try a larger problem per reader's request. Note that this is the first time that an alternating direction type of algorithm has ever been tested on problems of such a large scale.

We solve the problem on a cluster of 10 machines, where each machine is a "memory-optimized instance" with 68 GB RAM and 1 eight-core Intel Xeon E5-2665 CPU. They run Ubuntu 12.04 and are connected with 10 Gigabit ethernet network. Since each has 8 cores, we run the code with 80 processes so that each process runs on its own core. Such a setup is charged for under $17 per hour.

We solve the large-scale ℓ_1 minimization problems with a C code that matches our Matlab code. The C code has about 300 lines and it is available for download on authors' website.[1]

The breakdown of the wall-clock time is summarized in Tables 2 and 3. We can observe that Jacobi ADMM is very efficient in obtaining a relative low accuracy, which is usually sufficient for large-scale problems. We want to point out that the basic BLAS operations in our implantation can be further improved by using other libraries such as hardware-optimized BLAS libraries produced by ATLAS, Armadillo, etc. Those libraries might lead to several times of speedup.[2] We use GSL due to its ease of use and easy adaptation to other problems.

4 A Sufficient Condition for Convergence of Jacobi ADMM

In this section, we provide a sufficient condition to guarantee the convergence of Jacobi ADMM (Algorithm 3), which does not use proximal terms or any correction step. The condition only depends on the matrices A_i, without imposing further assumptions on the objective functions f_i or the penalty parameter ρ. For the Gauss–Seidel ADMM (Algorithm 2), a sufficient condition for convergence is provided in [5] for the special case $N = 3$, assuming two of the three coefficient matrices are orthogonal. Our condition does not require

[1] https://github.com/ZhiminPeng/Jacobi-ADMM.

[2] http://nghiaho.com/?p=1726.

Table 2 Time results for large scale ℓ_1 minimization examples (sparsity 5%)

	150 GB test			337 GB test		
	Itr	Time (m)	Cost ($)	Itr	Time (m)	Cost ($)
Data generation	–	0.7	0.2	–	1.6	0.5
CPU per iteration	–	1.3	–	–	2.9	–
Comm. per iteration	–	0.07	–	–	0.15	–
Reach 10^{-1}	46	1.0	0.2	56	2.8	0.4
Reach 10^{-2}	176	3.9	0.6	203	10.0	1.6
Reach 10^{-3}	791	17.5	2.9	880	43.5	7.1

Table 3 Time results for large scale ℓ_1 minimization examples (sparsity 15%)

	150 GB test			337 GB test		
	Itr	Time (m)	Cost ($)	Itr	Time (m)	Cost ($)
Reach 10^{-1}	115	2.5	0.4	181	8.8	1.4
Reach 10^{-2}	614	13.4	2.1	745	36.2	5.9
Reach 10^{-3}	1000 (reach 5.0×10^{-3})	21.8	3.6	1000 (reach 6.0×10^{-3})	48.5	7.9

exact orthogonality. Instead, we mainly assume that the matrices $A_i,\ i = 1, 2, \dots, N$ are mutually "near-orthogonal" and have full column-rank.

Theorem 4.1 *If there exists $\delta \geq 0$ such that*

$$\|A_i^\top A_j\| \leq \delta,\ \forall\, i \neq j,\ \text{and}\ \lambda_{min}(A_i^\top A_i) > 3(N-1)\delta,\ \forall\, i, \tag{4.1}$$

where $\lambda_{min}(A_i^\top A_i)$ denotes the smallest eigenvalue of $A_i^\top A_i$, then the sequence $\{\mathbf{u}^k\}$ generated by Algorithm 3 converges to a solution $\mathbf{u}^*$ to the problem (1.1).

The proof technique is motivated by the contraction analysis of the sequence $\{\mathbf{u}^k\}$ under some G-norm (e.g., [12,21,23]). We first need the following lemma:

Lemma 4.1 *Let*

$$G_0 := \begin{pmatrix} \rho A_1^\top A_1 & & & \\ & \ddots & & \\ & & \rho A_N^\top A_N & \\ & & & \frac{1}{\rho}\mathbf{I} \end{pmatrix},\quad Q_0 := \begin{pmatrix} \rho A_1^\top A_1 & & & A_1^\top \\ & \ddots & & \vdots \\ & & \rho A_N^\top A_N & A_N^\top \\ A_1 & \dots & A_N & \frac{1}{\rho}\mathbf{I} \end{pmatrix},$$

where $\mathbf{I}$ is the identity matrix of size $m \times m$. For $k \geq 1$, we have

$$\|\mathbf{u}^k - \mathbf{u}^*\|_{G_0}^2 - \|\mathbf{u}^{k+1} - \mathbf{u}^*\|_{G_0}^2 \geq \|\mathbf{u}^k - \mathbf{u}^{k+1}\|_{Q_0}^2, \tag{4.2}$$

where

$$\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_{Q_0}^2 := \|\mathbf{u}^k - \mathbf{u}^{k+1}\|_{G_0}^2 + 2(\lambda^k - \lambda^{k+1})^\top A(\mathbf{x}^k - \mathbf{x}^{k+1}). \tag{4.3}$$

This lemma follows directly from Lemma 2.1 since it is a special case with $\gamma = 1$ and $P_i = 0, \ \forall i$. Now we are ready to prove the theorem.

Proof of Theorem 4.1 By the assumption $\|A_i^\top A_j\| \le \delta, \ i \ne j$, we have

$$|\sum_{i\ne j}\langle A_i\mathbf{a}_i, A_j\mathbf{b}_j\rangle| \le \sum_{i\ne j}\delta\|\mathbf{a}_i\|\|\mathbf{b}_j\| \le \frac{\delta}{2}(N-1)(\|\mathbf{a}\|^2+\|\mathbf{b}\|^2), \ \forall\, \mathbf{a},\mathbf{b} \tag{4.4}$$

To simplify the notation, we let $\mathbf{a}_i^k := \mathbf{x}_i^k - \mathbf{x}_i^*, \ i = 1, 2, \ldots, N$. Note that $\lambda^k - \lambda^{k+1} = \rho A\mathbf{a}^{k+1}, \ \mathbf{x}^k - \mathbf{x}^{k+1} = \mathbf{a}^k - \mathbf{a}^{k+1}$. Then, we can rewrite (4.3) as

$$\begin{aligned}\frac{1}{\rho}\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_{Q_0}^2 &= \sum_i \|A_i(\mathbf{a}_i^k - \mathbf{a}_i^{k+1})\|^2 + \|A\mathbf{a}^{k+1}\|^2 + 2\langle A\mathbf{a}^{k+1}, \ A(\mathbf{a}^k - \mathbf{a}^{k+1})\rangle\\ &= \sum_i \|A_i\mathbf{a}_i^k\|^2 + 2\sum_{i\ne j}\langle A_i\mathbf{a}_i^{k+1}, A_j\mathbf{a}_j^k\rangle - \sum_{i\ne j}\langle A_i\mathbf{a}_i^{k+1}, A_j\mathbf{a}_j^{k+1}\rangle\\ &\ge \sum_i \|A_i\mathbf{a}_i^k\|^2 - (N-1)\delta(\|\mathbf{a}^{k+1}\|^2 + \|\mathbf{a}^k\|^2) - (N-1)\delta\|\mathbf{a}^{k+1}\|^2\\ &= \sum_i \|A_i\mathbf{a}_i^k\|^2 - (N-1)\delta\|\mathbf{a}^k\|^2 - 2(N-1)\delta\|\mathbf{a}^{k+1}\|^2,\end{aligned}$$

where the inequality comes from (4.4). By Lemma 4.1, we have

$$\begin{aligned}\|\mathbf{u}^k - \mathbf{u}^*\|_{G_0}^2 - 2(N-1)\delta\rho\|\mathbf{a}^k\|^2 \ge \|\mathbf{u}^{k+1} - \mathbf{u}^*\|_{G_0}^2 - 2(N-1)\delta\rho\|\mathbf{a}^{k+1}\|^2\\ + \rho\sum_i \|A_i\mathbf{a}_i^k\|^2 - 3(N-1)\delta\rho\|\mathbf{a}^k\|^2.\end{aligned} \tag{4.5}$$

We further simplify (4.5) as

$$b^k - b^{k+1} \ge d^k, \tag{4.6}$$

where the sequences $\{b^k\}$ and $\{d^k\}$ are defined by

$$b^k := \|\mathbf{u}^k - \mathbf{u}^*\|_{G_0}^2 - 2(N-1)\delta\rho\|\mathbf{a}^k\|^2, \tag{4.7}$$

$$d^k := \rho\sum_i \|A_i\mathbf{a}_i^k\|^2 - 3(N-1)\delta\rho\|\mathbf{a}^k\|^2. \tag{4.8}$$

By the definition of G_0, we have

$$b^k = \rho\sum_i \|A_i\mathbf{a}_i^k\|^2 - 2(N-1)\delta\rho\|\mathbf{a}_i^k\|^2 + \frac{1}{\rho}\|\lambda^k - \lambda^*\|^2. \tag{4.9}$$

Since we assume $\lambda_{\min}(A_i^\top A_i) > 3(N-1)\delta$, it follows that

$$\|A_i\mathbf{a}_i^k\|^2 \ge 3(N-1)\delta\|\mathbf{a}_i^k\|^2, \ \forall i. \tag{4.10}$$

Then it is easy to see that $b^k \ge 0$ and $d^k \ge 0$. By (4.6), the nonnegative sequence $\{b^k\}$ is monotonically non-increasing. Hence, $\{b^k\}$ converges to some $b^* \ge 0$. By (4.6), it also follows that $d^k \to 0$. Therefore, $\mathbf{a}^k \to 0$, i.e., $\mathbf{x}^k \to \mathbf{x}^*$.

Next we show $\lambda^k \to \lambda^*$. By taking limit of (4.9) and using $\mathbf{a}^k \to 0$, we have

$$b^* = \lim_{k\to\infty} b^k = \lim_{k\to\infty} \frac{1}{\rho}\|\lambda^k - \lambda^*\|^2. \tag{4.11}$$

To show $\lambda^k \to \lambda^*$, it thus suffices to show $b^* = 0$.

By (4.11), $\{\lambda^k\}$ is bounded and must have a convergent subsequence $\lambda^{k_j} \to \bar{\lambda}$. Recall the optimality conditions for the $\mathbf{x}_i$-subproblems (1.11):

$$A_i^\top \left(\lambda^k - \rho(A_i\mathbf{x}_i^{k+1} + \sum_{j\neq i} A_j\mathbf{x}_j^k - c)\right) \in \partial f_i(\mathbf{x}_i^{k+1}). \tag{4.12}$$

By Theorem 24.4 of [33], taking limit over the subsequence $\{k_j\}$ on both sides of (4.12) yields: $A_i^\top\bar{\lambda} \in \partial f_i(\mathbf{x}_i^*)$, $\forall\, i$. Therefore, $(\mathbf{x}^*, \bar{\lambda})$ satisfies the KKT conditions of the problem (1.1). Since $(\mathbf{x}^*, \lambda^*)$ is any KKT point, now we let $\lambda^* = \bar{\lambda}$. By (4.11) and $\|\lambda^{k_j} - \lambda^*\|^2 \to 0$, we must have $b^* = 0$, thereby completing the proof. □

Similar to Theorem 2.2, we can find the convergence rate. That is,

Theorem 4.2 *Let W be blockly diagonal matrix diag$(\rho\delta(N-1)I_1, \cdots, \rho\delta(N-1)I_N, \frac{1}{\rho}I)$. Under the assumptions in Theorem 4.1, $\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_W^2 = o(1/k)$. That is, $\|\mathbf{x}^k - \mathbf{x}^{k+1}\|^2 = o(1/k)$ and $\|\lambda^k - \lambda^{k+1}\|^2 = o(1/k)$.*

Under the similar near-orthogonality assumption on the matrices A_i, $i = 1, 2, \ldots, N$, we have the following convergence result for Jacobi-Proximal ADMM:

Theorem 4.3 *Suppose there exists $\delta \geq 0$ such that $\|A_i^\top A_j\| \leq \delta$ for all $i \neq j$, and the parameters in Algorithm 4 satisfy the following condition: for some $\alpha, \beta > 0$ and $0 < \gamma < 2$,*

$$\begin{cases} P_i \succ \rho(\frac{1}{\alpha} - 1)A_i^\top A_i + \frac{\rho}{\beta}\delta(N-1)\mathbf{I} \\ \lambda_{\min}(A_i^\top A_i) > \frac{2-\gamma+\beta}{2-\gamma-\alpha}\delta(N-1) \end{cases} \text{for } i = 1, \ldots, N. \tag{4.13}$$

Then Algorithm 4 converges to a solution to the problem (1.1).

Proof Let

$$H := \begin{pmatrix} A_1^\top A_1 & & \\ & \ddots & \\ & & A_N^\top A_N \end{pmatrix}.$$

If $\|A_i^\top A_j\| \leq \delta$ for all $i \neq j$, then it is easy to show the following: for any $\mathbf{x}$ and $\mathbf{y}$,

$$\begin{aligned} \|A\mathbf{x}\|^2 &= \sum_{i=1}^N \|A_i\mathbf{x}_i\|^2 + \sum_{i\neq j} \mathbf{x}_i^\top A_i^\top A_j\mathbf{x}_j \geq \sum_{i=1}^N \|A_i\mathbf{x}_i\|^2 - \delta\sum_{i\neq j}\|\mathbf{x}_i\|\|\mathbf{x}_j\| \\ &\geq \sum_{i=1}^N \|A_i\mathbf{x}_i\|^2 - \delta(N-1)\|\mathbf{x}\|^2 = \|\mathbf{x}\|^2_{[H-\delta(N-1)\mathbf{I}]}, \end{aligned}$$

and

$$2\mathbf{x}^\top A^\top A\mathbf{y} = 2\sum_{i=1}^N \mathbf{x}_i^\top A_i^\top A_j\mathbf{y}_j + 2\sum_{i\neq j}\mathbf{x}_i^\top A_i^\top A_j\mathbf{y}_j \geq 2\sum_{i=1}^N \mathbf{x}_i^\top A_i^\top A_j\mathbf{y}_j - 2\delta\sum_{i\neq j}\|\mathbf{x}_i\|\|\mathbf{y}_j\|$$
$$\geq -\sum_{i=1}^N \alpha\|A_i\mathbf{x}_i\|^2 - \beta\delta(N-1)\|\mathbf{x}\|^2 - \sum_{i=1}^N \frac{1}{\alpha}\|A_i\mathbf{y}_i\|^2 - \frac{1}{\beta}\delta(N-1)\|\mathbf{y}\|^2$$
$$= -\|\mathbf{x}\|^2_{[\alpha H+\beta\delta(N-1)\mathbf{I}]} - \|\mathbf{y}\|^2_{[\frac{1}{\alpha}H+\frac{1}{\beta}\delta(N-1)\mathbf{I}]}, \ \forall\alpha,\beta>0,$$

Using the above inequalities, we have

$$\frac{2}{\gamma}\left(\lambda^k-\lambda^{k+1}\right)^\top A\left(\mathbf{x}^k-\mathbf{x}^{k+1}\right) = 2\rho\left(\mathbf{x}^{k+1}-\mathbf{x}^*\right)A^\top A\left(\mathbf{x}^k-\mathbf{x}^{k+1}\right)$$
$$\geq -\rho\|\mathbf{x}^{k+1}-\mathbf{x}^*\|^2_{[\alpha H+\beta\delta(N-1)\mathbf{I}]} - \rho\|\mathbf{x}^k-\mathbf{x}^{k+1}\|^2_{[\frac{1}{\alpha}H+\frac{1}{\beta}\delta(N-1)\mathbf{I}]},$$

and $\|\lambda^k-\lambda^{k+1}\|^2 = \gamma^2\rho^2\|A(\mathbf{x}^{k+1}-\mathbf{x}^*)\|^2 \geq \gamma^2\rho^2\|\mathbf{x}^{k+1}-\mathbf{x}^*\|^2_{[H-\delta(N-1)\mathbf{I}]}$. Therefore,

$$\|\mathbf{u}^k-\mathbf{u}^{k+1}\|^2_Q \geq \|\mathbf{x}^k-\mathbf{x}^{k+1}\|^2_{G_x} + (2-\gamma)\rho\|\mathbf{x}^{k+1}-\mathbf{x}^*\|^2_{[H-\delta(N-1)\mathbf{I}]}$$
$$-\rho\|\mathbf{x}^{k+1}-\mathbf{x}^*\|^2_{[\alpha H+\beta\delta(N-1)\mathbf{I}]} - \rho\|\mathbf{x}^k-\mathbf{x}^{k+1}\|^2_{[\frac{1}{\alpha}H+\frac{1}{\beta}\delta(N-1)\mathbf{I}]}.$$

As long as the following holds:

$$\begin{cases} G_x \succ \frac{\rho}{\alpha}H + \frac{\rho}{\beta}\delta(N-1)\mathbf{I}, \\ (2-\gamma)\rho[H-\delta(N-1)\mathbf{I}] \succ \rho[\alpha H+\beta\delta(N-1)\mathbf{I}], \end{cases}$$

which is equivalent to the condition (4.13), there exists some $\eta > 0$ such that (2.8) and (2.9) hold. Then the convergence of Algorithm 4 follows immediately from the standard analysis of contraction methods [19]. □

Remark 4.1 The conditions in Theorem 4.1 and Theorem 4.3 guarantee that (2.8) holds, i.e., there exists some $\eta > 0$ such that $\|\mathbf{u}^k-\mathbf{u}^{k+1}\|^2_Q \geq \eta\cdot\|\mathbf{u}^k-\mathbf{u}^{k+1}\|^2$. But the matrix Q (or Q_0) is not necessarily positive semi-definite. Also, the term $\dfrac{2-\gamma+\beta}{2-\gamma-\alpha}\delta(N-1)$ in (4.13) may be negative for some α. Then, $\lambda_{min}(A_i^\top A_i)$ is allowed to be 0, in other words, A_i may not be of full column rank. As long as the conditions in (4.13) are satisfied, Algorithm 4 will converge.

5 Connections between the Prox-JADMM Algorithm and the Proximal Point Algorithm

The Jacobi-Proximal ADMM is equivalent to the proximal point algorithm (PPA) applied to the following optimal condition of (1.1):

$$\begin{bmatrix}0\\0\\\vdots\\0\\0\end{bmatrix} \in \begin{bmatrix}\partial f_1(\mathbf{x}_1)\\\partial f_2(\mathbf{x}_2)\\\vdots\\\partial f_N(\mathbf{x}_N)\\0\end{bmatrix} + \underbrace{\begin{bmatrix}0&0&\cdots&0&-A_1^T\\0&0&\cdots&0&-A_2^T\\\vdots&\vdots&\vdots&\vdots&\vdots\\0&0&\cdots&0&-A_N^T\\A_1&A_2&\cdots&A_N&0\end{bmatrix}}_{B}\begin{bmatrix}\mathbf{x}_1\\\mathbf{x}_2\\\vdots\\\mathbf{x}_N\\\lambda\end{bmatrix} - \begin{bmatrix}0\\0\\\vdots\\0\\c\end{bmatrix}. \quad (5.1)$$

Note that $\mathbf{x}$ is an optimal solution if and only if $\mathbf{x}_1, \ldots, \mathbf{x}_m$, and λ satisfy (5.1). Define $\mathbf{z} := (\mathbf{x}_1, \ldots, \mathbf{x}_N, \lambda)$, $F(\mathbf{z}) := \sum_{i=1}^N f_i(\mathbf{x}_i)$, and let $S\mathbf{z} = B\mathbf{z} - c$, then finding $\mathbf{z}$ satisfying (5.1) is equivalent to

$$\text{finding} \mathbf{z} \text{such that} \quad 0 \in \partial F(\mathbf{z}) + S\mathbf{z}. \tag{5.2}$$

Introducing a symmetric positive definite matrix W, and applying PPA to (5.2) gives

$$\mathbf{z}^{k+1} = J_{W^{-1}(\partial F + S)}(\mathbf{z}^k), \tag{5.3}$$

where $J_A := (I + A)^{-1}$ is the resolvent of the operator A. Next, we show that (5.3) is equivalent to Prox-JADMM with a proper choice of W. Let

$$W = \begin{bmatrix} Q - \rho A^T A & 0 \\ 0 & \frac{1}{\rho} I \end{bmatrix}, \tag{5.4}$$

where $Q = \mathrm{diag}(Q_1, \ldots, Q_N)$, Q_i are symmetric matrices and $A = (A_1, \ldots, A_N)$. We require $Q \succ \rho A^T A$ so that W is symmetric positive definite. The reason for choosing such a W is to decouple $\mathbf{x}_1, \ldots, \mathbf{x}_N$ and λ. Based on the definition of the resolvent operator, we have

$$W\mathbf{z}^{k+1} = W\mathbf{z}^k - \tilde{\nabla} F(\mathbf{z}^{k+1}) - S\mathbf{z}^{k+1}, \tag{5.5}$$

where $\tilde{\nabla} F(\mathbf{z}^{k+1}) \in \partial F(\mathbf{z}^{k+1})$. Substituting (5.4) to (5.5) gives the following update

$$\tilde{\nabla} f_i\left(\mathbf{x}_i^{k+1}\right) + Q_i\left(\mathbf{x}_i^{k+1} - \mathbf{x}^k\right) + \rho A_i^T\left(A\mathbf{x}^k - \frac{\lambda^k}{\rho} - c\right) = 0, \tag{5.6}$$

$$\lambda^{k+1} = \lambda^k - \rho\left(A\mathbf{x}^{k+1} - c\right). \tag{5.7}$$

where (5.6) corresponds to the optimal condition of the following problem

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} \; f_i(\mathbf{x}_i) + \langle \rho A_i^T\left(A\mathbf{x}^k - \frac{\lambda^k}{\rho} - c\right), \mathbf{x}_i\rangle + \frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_i^k\|_{Q_i}^2. \tag{5.8}$$

To show the equivalence, we distinguish two cases:

1. Let $Q_i = \tau_i I$, then (5.8) and (5.7) correspond to Prox-JADMM with $P_i = \tau I - \rho A_i^T A_i$.
2. Let $Q_i = \tau_i I + \rho A_i^T A_i$, then (5.8) is equivalent to

$$\begin{aligned}\mathbf{x}_i^{k+1} &= \arg\min_{\mathbf{x}_i} \; f_i(\mathbf{x}_i) + \left\langle \rho A_i^T\left(A\mathbf{x}^k - \frac{\lambda^k}{\rho} - c\right), \mathbf{x}_i\right\rangle + \frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_i^k\|_{\tau_i I + \rho A_i^T A_i}^2 \\ &= \arg\min_{\mathbf{x}_i} \; f_i(\mathbf{x}_i) + \frac{\rho}{2}\left\| A_i\mathbf{x}_i + \sum_{j \neq i} A_i \mathbf{x}_i^k - \frac{\lambda^k}{\rho} - c\right\|_2^2 + \frac{\tau_i}{2}\left\|\mathbf{x}_i - \mathbf{x}_i^k\right\|^2,\end{aligned}$$

which is the primal update of Prox-JADMM with $P_i = \tau_i I$.

6 Conclusion

Due to the dramatically increasing demand for dealing with big data, parallel and distributed computational methods are highly desirable. ADMM, as a versatile algorithmic tool, has proven to be very effective at solving many large-scale problems and well suited for distributed computing. Yet, its parallelization still needs further investigation and improvement. This paper proposes a simple parallel and distributed ADMM for solving problems with separable structures. The algorithm framework introduces more flexibility for solving the subproblems due to the use of proximal terms $\|\mathbf{x}_i - \mathbf{x}_i^k\|_{P_i}^2$ with wisely chosen P_i. Its theoretical properties such as global convergence and an $o(1/k)$ rate are established. Our numerical results demonstrate the efficiency of the proposed method in comparison with several existing algorithms. The code is available online for further studies.

Acknowledgements Wei Deng is partially supported by NSF grant ECCS-1028790. Ming-Jun Lai is partially supported by a Simon collaboration grant for 2013–2018 and by NSF grant DMS-1521537. Zhimin Peng and Wotao Yin are partially supported by NSF grants DMS-0748839 and DMS-1317602, and ARO/ARL MURI grant FA9550-10-1-0567. The authors would like to thank anonymous reviewers for their careful reading and suggestions.

Appendix: On $o(1/k)$ Convergence Rate of ADMM

The convergence of the standard two-block ADMM has been long established in the literature [14,16]. Its convergence rate has been actively studied; see [9,10,12,17,23–25,28] and the references therein. In the following, we briefly review the convergence analysis for ADMM ($N = 2$) and then improve the $O(1/k)$ convergence rate established in [24] slightly to $o(1/k)$ by using the same technique as in Sect. 2.2.

As suggested in [24], the quantity $\|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2$ can be used to measure the optimality of the iterations of ADMM , where

$$\mathbf{w} := \begin{pmatrix} \mathbf{x}_2 \\ \lambda \end{pmatrix}, \quad H := \begin{pmatrix} \rho A_2^\top A_2 & \\ & \frac{1}{\rho}\mathbf{I} \end{pmatrix},$$

and $\mathbf{I}$ is the identity matrix of size $m \times m$. Note that $\mathbf{x}_1$ is not part of $\mathbf{w}$ because $\mathbf{x}_1$ can be regarded as an intermediate variable in the iterations of ADMM, whereas $(\mathbf{x}_2, \lambda)$ are the essential variables [3]. In fact, if $\|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2 = 0$ then $\mathbf{w}^{k+1}$ is optimal. The reasons are as follows. Recall the subproblems of ADMM:

$$\mathbf{x}_1^{k+1} = \arg\min_{\mathbf{x}_1} f_1(\mathbf{x}_1) + \frac{\rho}{2}\left\|A_1\mathbf{x}_1 + A_2\mathbf{x}_2^k - \lambda^k/\rho\right\|^2, \tag{6.1}$$

$$\mathbf{x}_2^{k+1} = \arg\min_{\mathbf{x}_2} f_2(\mathbf{x}_2) + \frac{\rho}{2}\left\|A_1\mathbf{x}_1^{k+1} + A_2\mathbf{x}_2 - \lambda^k/\rho\right\|^2. \tag{6.2}$$

By the formula for λ^{k+1}, their optimality conditions can be written as:

$$A_1^\top\lambda^{k+1} - \rho A_1^\top A_2\left(\mathbf{x}_2^k - \mathbf{x}_2^{k+1}\right) \in \partial f\left(\mathbf{x}_1^{k+1}\right), \tag{6.3}$$

$$A_2^\top\lambda^{k+1} \in \partial f_2\left(\mathbf{x}_2^{k+1}\right). \tag{6.4}$$

In comparison with the KKT conditions (1.14a) and (1.14b), we can see that $\mathbf{u}^{k+1} = \left(\mathbf{x}_1^{k+1}, \mathbf{x}_2^{k+1}, \lambda^{k+1}\right)$ is a solution of (1.1) if and only if the following holds:

$$\mathbf{r}_p^{k+1} := A_1\mathbf{x}_1^{k+1} + A_2\mathbf{x}_2^{k+1} - c = 0 \quad \text{(primal feasibility)}, \tag{6.5}$$

$$\mathbf{r}_d^{k+1} := \rho A_1^\top A_2\left(\mathbf{x}_2^k - \mathbf{x}_2^{k+1}\right) = 0 \quad \text{(dual feasibility)}. \tag{6.6}$$

By the update formula for λ^{k+1}, we can write $\mathbf{r}_p$ equivalently as

$$\mathbf{r}_p^{k+1} = \frac{1}{\rho}\left(\lambda^k - \lambda^{k+1}\right). \tag{6.7}$$

Clearly, if $\|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2 = 0$ then the optimality conditions (6.5) and (6.6) are satisfied, so $\mathbf{w}^{k+1}$ is a solution. On the other hand, if $\|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2$ is large, then $\mathbf{w}^{k+1}$ is likely to be far away from being a solution. Therefore, the quantity $\|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2$ can be viewed as a measure of the distance between the iteration $\mathbf{w}^{k+1}$ and the solution set. Furthermore, based on the variational inequality (1.15) and the variational characterization of the iterations of ADMM, it is reasonable to use the quadratic term $\|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2$ rather than $\|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H$ to measure the convergence rate of ADMM (see [24] for more details).

The work [24] proves that $\|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2$ converges to zero at a rate of $O(1/k)$. The key steps of the proof are to establish the following properties:

– the sequence $\{\mathbf{w}^k\}$ is contractive:

$$\|\mathbf{w}^k - \mathbf{w}^*\|_H^2 - \|\mathbf{w}^{k+1} - \mathbf{w}^*\|_H^2 \geq \|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2, \tag{6.8}$$

– the sequence $\|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2$ is monotonically non-increasing:

$$\|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2 \leq \|\mathbf{w}^{k-1} - \mathbf{w}^k\|_H^2. \tag{6.9}$$

The contraction property (6.8) has been long established and its proof dates back to [14,16]. Inspired by [24], we provide a shorter proof for (6.9) than the one in [24].

Proof (Proof of (6.9)) Let $\Delta\mathbf{x}_i^{k+1} = \mathbf{x}_i^k - \mathbf{x}_i^{k+1}$ and $\Delta\lambda^{k+1} = \lambda^k - \lambda^{k+1}$. By Lemma 1.1, i.e., (1.17), the optimality condition 6.3 at the k-th and $(k+1)$-th iterations yields: $\langle \Delta\mathbf{x}_1^{k+1},\ A_1^\top \Delta\lambda^{k+1} - \rho A_1^\top A_2(\Delta\mathbf{x}_2^k - \Delta\mathbf{x}_2^{k+1})\rangle \geq 0$. Similarly for (6.4), we obtain $\langle \Delta\mathbf{x}_2^{k+1},\ A_2^\top \Delta\lambda^{k+1}\rangle \geq 0$. Adding the above two inequalities together, we have

$$\left(A_1\Delta\mathbf{x}_1^{k+1} + A_2\Delta\mathbf{x}_2^{k+1}\right)^\top \Delta\lambda^{k+1} - \rho\left(A_1\Delta\mathbf{x}_1^{k+1}\right)^\top A_2\left(\Delta\mathbf{x}_2^k - \Delta\mathbf{x}_2^{k+1}\right) \geq 0. \tag{6.10}$$

Using the equality according to (6.7):

$$A_1\Delta\mathbf{x}_1^{k+1} + A_2\Delta\mathbf{x}_2^{k+1} = \frac{1}{\rho}\left(\Delta\lambda^k - \Delta\lambda^{k+1}\right), \tag{6.11}$$

(6.10) becomes $\frac{1}{\rho}\left(\Delta\lambda^k - \Delta\lambda^{k+1}\right)^\top \Delta\lambda^{k+1} - \left(\Delta\lambda^k - \Delta\lambda^{k+1} - \rho A_2\Delta\mathbf{x}_2^{k+1}\right)^\top A_2\left(\Delta\mathbf{x}_2^k - \Delta\mathbf{x}_2^{k+1}\right) \geq 0$. After rearranging the terms, we get

$$\begin{aligned}
&\left(\sqrt{\rho}A_2\Delta\mathbf{x}_2^k + \frac{1}{\sqrt{\rho}}\Delta\lambda^k\right)^\top \left(\sqrt{\rho}A_2\Delta\mathbf{x}_2^{k+1} + \frac{1}{\sqrt{\rho}}\Delta\lambda^{k+1}\right) - \left(A_2\Delta\mathbf{x}_2^k\right)^\top \Delta\lambda^k \\
&- \left(A_2\Delta\mathbf{x}_2^{k+1}\right)^\top \Delta\lambda^{k+1} \\
&\geq \frac{1}{\rho}\left\|\Delta\lambda^{k+1}\right\|^2 + \rho\left\|A_2\Delta\mathbf{x}_2^{k+1}\right\|^2 = \|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2.
\end{aligned} \tag{6.12}$$

By the Cauchy-Schwarz inequality, we have $(a_1+b_1)^\top(a_2+b_2) \leq (\|a_1+b_1\|^2 + \|a_2+b_2\|^2)/2$, or equivalently, $(a_1+b_1)^\top(a_2+b_2) - a_1^\top b_1 - a_2^\top b_2 \leq (\|a_1\|^2 + \|b_1\|^2 + \|a_2\|^2 +$

$\|b_2\|^2)/2$. Applying this inequality to the left-hand side of (6.12), we have

$$\|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2 \leq \left(\rho\|A_2\Delta\mathbf{x}_2^k\|^2 + \frac{1}{\rho}\|\Delta\lambda^k\|^2 + \rho\|A_2\Delta\mathbf{x}_2^{k+1}\|^2 + \frac{1}{\rho}\|\Delta\lambda^{k+1}\|^2\right)/2$$
$$= \left(\|\mathbf{w}^{k-1} - \mathbf{w}^k\|_H^2 + \|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2\right)/2,$$

and thus (6.9) follows immediately.

We are now ready to improve the convergence rate from $O(1/k)$ to $o(1/k)$.

Theorem 6.1 *The sequence $\{\mathbf{w}^k\}$ generated by Algorithm 2 (for $N = 2$) converges to a solution $\mathbf{w}^*$ of problem (1.1) in the H-norm, i.e., $\|\mathbf{w}^k - \mathbf{w}^*\|_H^2 \to 0$, and $\|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2 = o(1/k)$. Therefore,*

$$\left\|A_1\mathbf{x}_1^k - A_1\mathbf{x}_1^{k+1}\right\|^2 + \left\|A_2\mathbf{x}_2^k - A_2\mathbf{x}_2^{k+1}\right\|^2 + \left\|\lambda^k - \lambda^{k+1}\right\|^2 = o(1/k). \qquad (6.13)$$

Proof Using the contractive property of the sequence $\{\mathbf{w}^k\}$ (6.8) along with the optimality conditions, the convergence of $\|\mathbf{w}^k - \mathbf{w}^*\|_H^2 \to 0$ follows from the standard analysis for contraction methods [19].

By (6.8), we have

$$\sum_{k=1}^{n}\|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2 \leq \|\mathbf{w}^1 - \mathbf{w}^*\|_H^2 - \|\mathbf{w}^{n+1} - \mathbf{w}^*\|_H^2, \ \forall n. \qquad (6.14)$$

Therefore, $\sum_{k=1}^{\infty}\|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2 < \infty$. By (6.9), $\|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2$ is monotonically nonincreasing and nonnegative. So Lemma 1.1 indicates that $\|\mathbf{w}^k - \mathbf{w}^{k+1}\|_H^2 = o(1/k)$, which further implies that $\|A_2\mathbf{x}_2^k - A_2\mathbf{x}_2^{k+1}\|^2 = o(1/k)$ and $\|\lambda^k - \lambda^{k+1}\|^2 = o(1/k)$. By (6.11), we also have $\|A_1\mathbf{x}_1^k - A_1\mathbf{x}_1^{k+1}\|^2 = o(1/k)$. Thus (6.13) follows immediately. □

Remark 6.1 The proof technique based on Lemma 1.1 can be applied to improve some other existing convergence rates of $O(1/k)$ (e.g., [8,21]) to $o(1/k)$ as well.

References

1. Awanou, G., Lai, M.J., Wenston, P.: The multivariate spline method for numerical solution of partial differential equations and scattered data interpolation. In: Chen, G., Lai, M.J. (eds.) Wavelets and Splines, pp. 24–74. Nashboro Press, Nashville (2006)
2. Bertsekas, D., Tsitsiklis, J.: Parallel and Distributed Computation: Numerical Methods, 2nd edn. Athena Scientific, Belmont (1997)
3. 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 (2011)
4. Chandrasekaran, V., Parrilo, P.A., Willsky, A.S.: Latent variable graphical model selection via convex optimization. Ann. Stat. **40**(4), 1935–1967 (2012)
5. Chen, C., He, B.S., Ye, Y.Y., Yuan, X.M.: The direct extension of admm for multi-block convex minimization problems is not necessarily convergent. Math. Program. **155**(1), 57–79 (2016)
6. Chen, C., Shen, Y., You, Y.: On the convergence analysis of the alternating direction method of multipliers with three blocks. Abstr. Appl. Anal. **2013**, 183961 (2013)
7. Chen, G., Teboulle, M.: A proximal-based decomposition method for convex minimization problems. Math. Program. **64**(1), 81–101 (1994)
8. Corman, E., Yuan, X.M.: A generalized proximal point algorithm and its convergence rate. SIAM J. Optim. **24**(4), 1614–1638 (2014)
9. Davis, D., Yin, W.: Convergence rate analysis of several splitting schemes. UCLA CAM Report, pp. 14–51 (2014)

10. Davis, D., Yin, W.: Convergence rates of relaxed peaceman–rachford and admm under regularity assumptions. UCLA CAM Report, pp. 14–58 (2014)
11. Davis, D., Yin, W.: A three-operator splitting scheme and its optimization applications. UCLA CAM Report, pp. 15–13 (2015)
12. Deng, W., Yin, W.: On the global and linear convergence of the generalized alternating direction method of multipliers. J. Sci. Comput. **66**(3), 889–916 (2016)
13. Everett, H.: Generalized lagrange multiplier method for solving problems of optimum allocation of resources. Oper. Res. **11**(3), 399–417 (1963)
14. Gabay, D., Mercier, B.: A dual algorithm for the solution of nonlinear variational problems via finite element approximation. Comput. Math. Appl. **2**(1), 17–40 (1976)
15. Glowinski, R.: Numerical methods for nonlinear variational problems. Springer Series in Computational Physics. Springer, Berlin (1984)
16. Glowinski, R., Marrocco, A.: Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité, d'une classe de problèmes de Dirichlet non linéaires. Laboria (1975)
17. Goldstein, T., O'Donoghue, B., Setzer, S., Baraniuk, R.: Fast alternating direction optimization methods. SIAM J. Imaging Sci. **7**(3), 1588–1623 (2014)
18. Han, D., Yuan, X.: A note on the alternating direction method of multipliers. J. Optim. Theory Appl. **155**(1), 227–238 (2012)
19. He, B.S.: A class of projection and contraction methods for monotone variational inequalities. Appl. Math. Optim. **35**(1), 69–76 (1997)
20. He, B.S.: Parallel splitting augmented lagrangian methods for monotone structured variational inequalities. Comput. Optim. Appl. **42**(2), 195–212 (2009)
21. He, B.S., Hou, L.S., Yuan, X.M.: On full Jacobian decomposition of the augmented lagrangian method for separable convex programming. SIAM J. Optim. **25**, 2274–2312 (2015)
22. He, B.S., Tao, M., Yuan, X.M.: Alternating direction method with gaussian back substitution for separable convex programming. SIAM J. Optim. **22**(2), 313–340 (2012)
23. He, B.S., Yuan, X.M.: On the $O(1/n)$ convergence rate of the Douglas-Rachford alternating direction method. SIAM J. Numer. Anal. **50**(2), 700–709 (2012)
24. He, B.S., Yuan, X.M.: On non-ergodic convergence rate of Douglas-Rachford alternating direction method of multipliers. Numer. Math. **130**(3), 567–577 (2015)
25. Hong, M., Luo, Z.Q.: On the Linear Convergence of the Alternating Direction Method of Multipliers. arXiv:1208.3922 (2012)
26. Li, M., Sun, D., Toh, K.C.: A Convergent 3-Block Semi-Proximal ADMM for Convex Minimization Problems with One Strongly Convex Block. arXiv:1410.7933 [math] (2014)
27. Lin, T., Ma, S., Zhang, S.: On the Convergence Rate of Multi-Block ADMM. arXiv:1408.4265 [math] (2014)
28. Lions, P.L., Mercier, B.: Splitting algorithms for the sum of two nonlinear operators. SIAM J. Numer. Anal. **16**(6), 964–979 (1979)
29. Mota, J.F., Xavier, J.M., Aguiar, P.M., Puschel, M.: D-admm: a communication-efficient distributed algorithm for separable optimization. IEEE Trans. Signal Process. **61**, 2718–2723 (2013)
30. Nesterov, Y.: Smooth minimization of non-smooth functions. Math. Program. **103**(1), 127–152 (2005)
31. Peng, Y., Ganesh, A., Wright, J., Xu, W., Ma, Y.: RASL: robust alignment by sparse and low-rank decomposition for linearly correlated images. IEEE Trans. Pattern Anal. Mach. Intell. **34**, 2233–2246 (2012)
32. Peng, Z., Yan, M., Yin, W.: Parallel and distributed sparse optimization. In: IEEE Asilomar Conference on Signals Systems and Computers (2013)
33. Rockafellar, R.T.: Convex Analysis. Princeton University Press, Princeton (1997)
34. Shor, N.Z., Kiwiel, K.C., Ruszcayski, A.: Minimization Methods for Non-differentiable Functions. Springer, New York (1985)
35. Tao, M., Yuan, X.M.: Recovering low-rank and sparse components of matrices from incomplete and noisy observations. SIAM J. Optim. **21**(1), 57–81 (2011)
36. Wang, X.F., Hong, M.Y., Ma, S.Q., Luo, Z.Q.: Solving multiple-block separable convex minimization problems using two-block alternating direction method of multipliers. Pac. J. Optim. **11**(4), 57–81 (2015)
37. Yang, J.F., Zhang, Y.: Alternating direction algorithms for ℓ_1-problems in compressive sensing. SIAM J. Sci. Comput. **33**(1), 250–278 (2011)
38. Zhang, X., Burger, M., Osher, S.: A unified primal-dual algorithm framework based on Bregman iteration. J. Sci. Comput. **46**(1), 20–46 (2011)