An Alternating Direction Method of Multiplier with the BFGS update for Structured Convex Quadratic Optimization

Yan Gu, and Nobuo Yamashita

July 11, 2018

Abstract

The alternating direction method of multipliers (ADMM) is an effective method for solving wide fields of convex problems. At each iteration, the classical ADMM solves two subproblems exactly. However, in many applications, it is expensive or impossible to obtain the exact solutions of the subproblem. To overcome the difficulty, some proximal terms are added to the subproblems. This class of methods normally solves the original subproblem approximately, and thus takes more iterations. This fact urges us to consider that a special proximal term can lead to a better result as the classical ADMM. In this paper, we propose a proximal ADMM whose regularized matrix in the proximal term is generated by the BFGS update (or Limited memory BFGS) at every iteration. These types of matrices use the second-order information of the objective function. The convergence of the proposed method is proved under certain assumptions. Numerical results are given to show the effectiveness of the proposed proximal ADMM.

Keywords: alternating direction method, variable metric semi-proximal method, convergence, BFGS method, limited memory BFGS, convex minimization.

1 Introduction

We consider the following convex minimization problem:

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\|Ax-b\|^2 + g(x) \\ \text{subject to} & x \in \mathbb{R}^n, \end{array} \tag{1.1}$$

where $g : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ is a proper convex function, $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. For example, "g" here can be an indicator function on a convex set or the $l1$ penalty function: $g(x) = \|x\|_1 := \sum_{i=1}^m |x_i|$. The problem (1.1) includes as special cases many important statistical learning problems such as the LASSO problem [14]. The number n of variables in these learning problems is usually large.

Let $f(x) = \frac{1}{2}\|Ax-b\|^2$. Then the problem (1.1) can be written as

$$\begin{array}{ll} \text{minimize} & f(x) + g(y) \\ \text{subject to} & x - y = 0 \\ & x, y \in \mathbb{R}^n. \end{array} \tag{1.2}$$

Let $\mathcal{L}_\beta(x, y, \lambda)$ be the augmented Lagrangian function for (1.2) that defined by

$$\mathcal{L}_\beta(x, y, \lambda) := f(x) + g(y) - \langle \lambda, x - y\rangle + \frac{\beta}{2}\|x - y\|^2, \tag{1.3}$$

where $\lambda \in \mathbb{R}^n$ is multipliers associated to the linear constraints and $\beta > 0$ is a penalty parameter.

Based on the classic Douglas-Rachford operator splitting method [3], the alternating direction method of multipliers (ADMM) was proposed by Gabay and Mercier [6], Glowinski and Marrocco [7] in the mid-1970s, which generates the iterative sequence via the following recursion:

$$\begin{cases} x^{k+1} = \arg\min\limits_{x} \mathcal{L}_\beta(x, y^k, \lambda^k), & \text{(1.4a)} \\ y^{k+1} = \arg\min\limits_{y} \mathcal{L}_\beta(x^{k+1}, y, \lambda^k), & \text{(1.4b)} \\ \lambda^{k+1} = \lambda^k - \beta(x^{k+1} - y^{k+1}). & \text{(1.4c)} \end{cases}$$

The global convergence of the ADMM (1.4a)-(1.4c) can be established under very mild conditions [1].

By noting the fact that the subproblem in (1.4a)-(1.4c) may be difficult to solve exactly in many applications, Eckstein [4] and He et al. [10] have considered to add proximal terms to the subproblems for different purpose. Recently, Fazel et al. [5] proposed the following semi-proximal ADMM scheme:

$$\begin{cases} x^{k+1} = \arg\min\limits_{x} \ \mathcal{L}_\beta(x, y^k, \lambda^k) + \frac{1}{2}\|x - x^k\|_T^2, & \text{(1.5a)} \\ y^{k+1} = \arg\min\limits_{y} \ \mathcal{L}_\beta(x^{k+1}, y, \lambda^k) + \frac{1}{2}\|y - y^k\|_S^2, & \text{(1.5b)} \\ \lambda^{k+1} = \lambda^k - \gamma\beta(x^{k+1} - y^{k+1}), & \text{(1.5c)} \end{cases}$$

where $\gamma \in (0, (1+\sqrt{5})/2)$ and $\|z\|_G = \sqrt{z^\top G z}$ for $z \in \mathbb{R}^n$ and $G \in \mathbb{R}^{n\times n}$. Fazel et al. [5] show its global convergence when T and S are positive semidefinite, in contrast to the positive definite requirements in the classical algorithms [4,10], which makes the algorithm more flexible. See [2,5,11,15] for a brief history of the development of the semi-proximal ADMM and the corresponding convergence results.

In this paper, we suppose that y^{k+1} in (1.5b) is easily obtained. For example, if $g(y) = \tau\|y\|_1$ with $\tau > 0$ and $S = 0$, then we can get y^{k+1} within $O(n)$. Then our main focus is how to solve (1.5a) when n is large. We may choose a reasonable positive semidefinite matrix T so that we get x^{k+1} quickly.

One of such examples of T is $T = \gamma I - A^\top A$ with $\gamma > \lambda_{\max}(A^\top A)$. Then (1.5a) is written as

$$\begin{aligned} x^{k+1} &= \arg\min_x \left\{ f(x) - \langle \lambda^k, x - y^k\rangle + \frac{\beta}{2}\|x - y^k\|^2 + \frac{1}{2}\|x - x^k\|_T^2 \right\} \\ &= \arg\min_x \left\{ (Ax^k - b)^\top Ax - \lambda^k x + \frac{\beta}{2}\|x - y^k\|^2 + \frac{\gamma}{2}\|x - x^k\|^2 \right\} \\ &= (\lambda^k + \beta y^k + \gamma x^k - A^\top Ax^k + A^\top b)/(\beta + \gamma). \end{aligned}$$

The other example is $T = \gamma I - \beta I - A^\top A$ with $\gamma > \lambda_{\max}(\beta I + A^\top A)$. Then (1.5a) is written as

$$x^{k+1} = x^k - \gamma^{-1}(A^\top Ax^k - A^\top b - \lambda^k + \beta x^k - \beta y^k).$$

In both cases x^{k+1} is calculated within $O(mn)$. However, since the subproblems do not include second-order information on f, the convergence of ADMM with such T might be slow.

It is obvious that a desired matrix T is a matrix such that it is positive semidefinite, the subproblem (1.5a) is easily solved, and it has some information on f. Let M be defined as the Hessian matrix of the augmented Lagrangian function $\mathcal{L}_\beta$, that is $M := \nabla^2_{xx}\mathcal{L}_\beta(x, y, \lambda) = A^\top A + \beta I$. Note that $M \succ 0$ whenever $\beta > 0$.

Then, we consider a matrix B that has the following three properties:

(i) $T = B - M$;

(ii) $B \succeq M$;

(iii) B has some second order information on M.

Using B, x-subproblem (1.5a) is written as

$$\begin{aligned} x^{k+1} &= \arg\min_x \left\{ f(x) - \langle \lambda^k, x - y^k \rangle + \frac{\beta}{2}\|x - y^k\|^2 + \frac{1}{2}\|x - x^k\|_T^2 \right\} \\ &= \arg\min_x \left\{ \langle A^\top (Ax^k - b) + \beta(x^k - y^k) - \lambda^k, x \rangle + \frac{1}{2}\|x - x^k\|_B^2 \right\} \\ &= x^k - B^{-1}\left(A^\top A x^k - A^\top b - \lambda^k + \beta(x^k - y^k) \right). \end{aligned}$$

In this paper, we propose to construct B via the BFGS update at every iteration. Then the proximal term B and T at every step depend on k, that is, they become B_k and T_k, and the resulting ADMM is a variable metric semidefinite proximal ADMM. The detail discussion about the choice of B_k will be shown later in Section 3. We consider the iterative scheme of the variable metric semi-proximal ADMM (short by VMSP-ADMM) algorithm given as:

$$\begin{cases} x^{k+1} = \arg\min_x \ \mathcal{L}_\beta(x, y^k, \lambda^k) + \frac{1}{2}\|x - x^k\|_{T_k}^2, & \text{(1.6a)} \\ y^{k+1} = \arg\min_y \ \mathcal{L}_\beta(x^{k+1}, y, \lambda^k) + \frac{1}{2}\|y - y^k\|_S^2, & \text{(1.6b)} \\ \lambda^{k+1} = \lambda^k - \beta(x^{k+1} - y^{k+1}), & \text{(1.6c)} \end{cases}$$

which is also related to the methods studied in He et al. [10] with the T_k is assumed to be positive definite and Gonçalves et al. [8].

The main contributions of the paper are as follows:

1. We give sufficient conditions on positive semidefinite T_k under which a sequence $\{x^k\}$ generated by VMSP-ADMM (1.6) globally converges to a solution of (1.2).
2. We propose update formulae of T_k with BFGS which satisfy the above sufficient conditions for global convergence.
3. We report some numerical results for the proposed methods which shows that they outperform the existing proximal ADMM when n is large.

The rest of the paper is organized as follows. Some preliminary results for the variable metric semi-proximal ADMM are provided in Section 2. In Section 3, we propose our algorithm and show the convergence of the proposed method under certain flexible conditions on the proximal matrices sequence. In Section 4, we present some numerical experiment results for ADMM with BFGS and Limited memory BFGS. Finally, we make some concluding remarks in Section 5.

Notations : Here we give some notation that will be used in following sections.

We define $\langle \cdot, \cdot \rangle$ as the standard inner product in $\mathbb{R}^n$:

$$\langle x, y \rangle = \sum_{i=1}^n x_i y_i, \quad x, y \in \mathbb{R}^n.$$

We use $\|\cdot\|$ to denote the 2-norm of a vector:

$$\|x\| = \langle x, x \rangle^{1/2}.$$

For a real symmetric matrix S, we mark $S \succeq 0$ ($S \succ 0$) if S is positive semidefinite (positive definite).

2 Variable metric semi-proximal ADMM and its Convergence Property

In this section, we consider the variable metric semi-proximal ADMM (1.6) for problem (1.2) with a general convex function f. We first give the optimality condition of problem (1.2) and some properties which will be frequently used in our analysis. Then we show some general convergence properties for the variable metric semi-proximal ADMM (1.6a)-(1.6c) (VMSP-ADMM).

The KKT conditions of problem (1.2) are written as:

$$\begin{cases} f'(x^*) - \lambda^* = 0, & \text{(2.1a)} \\ g'(y^*) + \lambda^* = 0, & \text{(2.1b)} \\ x^* - y^* = 0, & \text{(2.1c)} \end{cases}$$

where $f'(x^*) \in \partial f(x^*)$ and $g'(y^*) \in \partial g(y^*)$.

Let Ω^* be a set of (x^*, y^*, λ^*) satisfying the KKT condition (2.1). Throughout this paper, we make the following assumption.

Assumption 2.1. *The set Ω^* of KKT points is non-empty.*

Let

$$F(w) = \begin{pmatrix} f'(x) - \lambda \\ g'(y) + \lambda \\ x - y \end{pmatrix},$$

where $f'(x) \in \partial f(x)$ and $g'(y) \in \partial g(y)$, and let $\Omega = \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^m$.

Definition 2.2. *F is said to be monotone if $(u - v)^\top (F(u) - F(v)) \geq 0, \ \ \forall u, v \in \Omega$.*

2.1 The algorithms

Let $(x^0, y^0, \lambda^0) \in \Omega$ be the initial arbitrary triplet. By deriving the first-order optimality conditions of the involved subproblems in (1.6), we can easily show that for the given triplet $(x^k, y^k, \lambda^k) \in \Omega$, and $\beta > 0$, the new triplet $(x^{k+1}, y^{k+1}, \lambda^{k+1})$ is generated by the following procedure.

- step 1: Find $x^{k+1} \in \mathbb{R}^n$ such that $f'(x^{k+1}) \in \partial f(x^{k+1})$ and

$$f'(x^{k+1}) - \lambda^k + \beta(x^{k+1} - y^k) + T_k(x^{k+1} - x^k) = 0, \tag{2.2}$$

- step 2: Find $y^{k+1} \in \mathbb{R}^n$ such that $g'(y^{k+1}) \in \partial g(y^{k+1})$ and

$$g'(y^{k+1}) + \lambda^k - \beta(x^{k+1} - y^{k+1}) + S(y^{k+1} - y^k) = 0, \tag{2.3}$$

- step 3: Update λ^{k+1} via

$$\lambda^{k+1} = \lambda^k - \beta(x^{k+1} - y^{k+1}). \tag{2.4}$$

In the following analysis, we will use (2.2) to (2.4).

2.2 Convergence property of variable metric semi-proximal ADMM

In this subsection, we give conditions on T_k under which VMSP-ADMM converges globally. To this end, we fist show that a sequence related to $\{(x^k, y^k, \lambda^k)\}$ generated by (1.6) is contractive.

For $k = 0, 1, 2, ...$, we use the following notation:

$$u^* = \begin{pmatrix} x^* \\ y^* \end{pmatrix}, \ u^k = \begin{pmatrix} x^k \\ y^k \end{pmatrix}, \ w^k = \begin{pmatrix} x^k \\ y^k \\ \lambda^k \end{pmatrix}, \text{ and } D_k = \begin{pmatrix} T_k & 0 \\ 0 & S \end{pmatrix}.$$

Moreover, for simplicity, we denote

$$F(w^k) = \begin{pmatrix} f'(x^k) - \lambda^k \\ g'(y^k) + \lambda^k \\ x^k - y^k \end{pmatrix}, \tag{2.5}$$

where $f'(x^k)$ and $g'(y^k)$ are obtained in steps 1 and 2 in VMSP-ADMM. Note that F is a set to point map in general.

Lemma 2.3. *Let $\{w^k\}$ be generated by* (1.6)*. Then it holds that for given $w^* = (x^*, y^*, \lambda^*) \in \Omega^*$,*

$$(u^{k+1} - u^*)^\top D_k(u^{k+1} - u^k) + \frac{1}{\beta}(\lambda^{k+1} - \lambda^*)^\top(\lambda^{k+1} - \lambda^k) \leq \beta(x^{k+1} - x^*)^\top(y^k - y^{k+1}).$$

Proof. Since $w^* \in \Omega^*$ and $x^{k+1}, y^{k+1} \in \mathbb{R}^n$, there exist $f'(x^*) \in \partial f(x^*)$ and $g'(y^*) \in \partial g(y^*)$ such that

$$(x^{k+1} - x^*)^\top\{f'(x^*) - \lambda^*\} = 0 \tag{2.6}$$

and

$$(y^{k+1} - y^*)^\top\{g'(y^*) + \lambda^*\} = 0. \tag{2.7}$$

On the other hand, from (2.2)-(2.3), we have

$$(x^* - x^{k+1})^\top\{f'(x^{k+1}) - \lambda^k + \beta(x^{k+1} - y^k) + T_k(x^{k+1} - x^k)\} = 0$$

and

$$(y^* - y^{k+1})^\top\{g'(y^{k+1}) + \lambda^{k+1} + S(y^{k+1} - y^k)\} = 0. \tag{2.8}$$

It then follows from $\lambda^{k+1} = \lambda^k - \beta(x^{k+1} - y^{k+1})$ that

$$(x^* - x^{k+1})^\top\{f'(x^{k+1}) - \lambda^{k+1} + \beta(y^{k+1} - y^k) + T_k(x^{k+1} - x^k)\} = 0 \tag{2.9}$$

Since the subdifferential mapping ∂f is maximal monotone, we have

$$(x^{k+1} - x^*)^\top(f'(x^{k+1}) - f'(x^*)) \geq 0. \tag{2.10}$$

Adding Eqs. (2.6) and (2.9) and using the monotonicity (2.10), we have

$$(x^{k+1} - x^*)^\top T_k(x^{k+1} - x^*) + (x^{k+1} - x^*)^\top(\lambda^* - \lambda^{k+1}) \leq \beta(x^{k+1} - x^*)^\top(y^k - y^{k+1}). \tag{2.11}$$

In a similar way, adding Eqs. (2.7) and (2.8) and using the monotonicity of ∂g, we have that

$$(y^{k+1} - y^*)^\top S(y^{k+1} - y^*) + (y^* - y^{k+1})^\top(\lambda^* - \lambda^{k+1}) \leq 0. \tag{2.12}$$

Adding (2.11) and (2.12) and using $x^* - y^* = 0$, it follows that

$$(u^{k+1} - u^*)^\top D_k(u^{k+1} - u^k) + (x^{k+1} - y^{k+1})^\top(\lambda^* - \lambda^{k+1}) \leq \beta(x^{k+1} - x^*)^\top(y^k - y^{k+1}).$$

Using $x^{k+1} - y^{k+1} = \frac{1}{\beta}(\lambda^k - \lambda^{k+1})$, the assertion follows directly. □

Lemma 2.4. *Let $w^* = (x^*, y^*, \lambda^*) \in \Omega^*$, and let $\{w^k\}$ be generated by the scheme* (1.6)*. Then it holds that*

$$\begin{aligned} &\|u^{k+1} - u^*\|^2_{D_k} + \frac{1}{\beta}\|\lambda^{k+1} - \lambda^*\|^2 + \beta\|y^{k+1} - y^*\|^2 \\ &\leq \|u^k - u^*\|^2_{D_k} + \frac{1}{\beta}\|\lambda^k - \lambda^*\|^2 + \beta\|y^k - y^*\|^2 - (\|u^{k+1} - u^k\|^2_{D_k} + \beta\|x^{k+1} - y^k\|^2). \end{aligned} \tag{2.13}$$

Proof. First, by the identity $\|a+b\|^2 = \|a\|^2 - \|b\|^2 + 2(a+b)^\top b$, we get

$$\begin{aligned}
&\|u^{k+1}-u^*\|_{D_k}^2 + \frac{1}{\beta}\|\lambda^{k+1}-\lambda^*\|^2 + \beta\|y^{k+1}-y^*\|^2 \\
&= \|u^k-u^*\|_{D_k}^2 + \frac{1}{\beta}\|\lambda^k-\lambda^*\|^2 + \beta\|y^k-y^*\|^2 - (\|u^{k+1}-u^k\|_{D_k}^2 + \frac{1}{\beta}\|\lambda^{k+1}-\lambda^k\|^2 + \beta\|y^{k+1}-y^k\|^2) \\
&\quad + 2(u^{k+1}-u^*)^\top D_k(u^{k+1}-u^k) + \frac{2}{\beta}(\lambda^{k+1}-\lambda^*)^\top(\lambda^{k+1}-\lambda^k) + 2\beta(y^{k+1}-y^*)^\top(y^{k+1}-y^k). \qquad (2.14)
\end{aligned}$$

Now we deal with the last three crossing terms of the right-hand side of (2.14). We want to relate the summation of these three terms to the term $\beta\|x^{k+1}-y^k\|^2$, and then the assertion (2.13) can be proved.

From Lemma 2.3, it follows that

$$2(u^{k+1}-u^*)^\top D_k(u^{k+1}-u^k) + \frac{2}{\beta}(\lambda^{k+1}-\lambda^*)^\top(\lambda^{k+1}-\lambda^k) \le 2\beta(x^{k+1}-x^*)^\top(y^k-y^{k+1}),$$

which implies that

$$\begin{aligned}
&2(u^{k+1}-u^*)^\top D_k(u^{k+1}-u^k) + \frac{2}{\beta}(\lambda^{k+1}-\lambda^*)^\top(\lambda^{k+1}-\lambda^k) + 2\beta(y^{k+1}-y^*)^\top(y^{k+1}-y^k) \\
&\le 2\beta(x^{k+1}-x^*)^\top(y^k-y^{k+1}) + 2\beta(y^{k+1}-y^*)^\top(y^{k+1}-y^k) \qquad (\text{use } x^*-y^*=0) \\
&= 2\beta(x^{k+1}-y^{k+1})^\top(y^k-y^{k+1}) \qquad (\text{use (2.4)}) \\
&= -2(\lambda^{k+1}-\lambda^k)^\top(y^k-y^{k+1}). \qquad (2.15)
\end{aligned}$$

We can also get

$$\frac{1}{\beta}\|\lambda^{k+1}-\lambda^k\|^2 + \beta\|y^{k+1}-y^k\|^2 + 2(\lambda^{k+1}-\lambda^k)^\top(y^k-y^{k+1}) = \beta\|x^{k+1}-y^k\|^2. \qquad (2.16)$$

Substituting (2.15) to (2.14) and using (2.16), we get the assertion of this lemma. □

Remark 2.5. *Lemma 2.4 provides the fundamental result in the convergence analysis of the method. Note that when $T_k \equiv T$, that is, $D_k \equiv D$, it yields form* (2.14) *that the sequence $\{\|u^{k+1}-u^*\|_D^2 + \frac{1}{\beta}\|\lambda^{k+1}-\lambda^*\|^2 + \beta\|y^{k+1}-y^*\|^2\}$ is strictly monotonically decreasing and $\lim_{k\to\infty}(\|u^{k+1}-u^k\|_D^2 + \beta\|x^{k+1}-y^k\|^2) = 0$, and thus the convergence can be directly established.*

Next we give an upper bound for the residual $\|F(w^{k+1})\|$.

Lemma 2.6. *Let $w^* = (x^*, y^*, \lambda^*) \in \Omega^*$, and let $\{w^k\}$ be generated by the scheme* (1.6)*. Suppose that sequence $\{T_k\}$ is bounded. Then, there exists a constant $\mu > 0$ such that for all $k \ge 0$, we have*

$$\|F(w^{k+1})\| \le \mu\left(\|u^{k+1}-u^k\|_{D_k}^2 + \|x^{k+1}-y^k\|^2\right). \qquad (2.17)$$

Proof. From (2.5) and (2.2)-(2.4), we get

$$\begin{aligned}
\|F(w^{k+1})\| &= \left\| \begin{array}{c} f'(x^{k+1}) - \lambda^{k+1}) \\ g'(y^{k+1}) + \lambda^{k+1} \\ x^{k+1}-y^{k+1} \end{array} \right\| = \left\| \begin{array}{c} \beta(y^k-y^{k+1}) - T_k(x^{k+1}-x^k) \\ -S(y^{k+1}-y^k) \\ (x^{k+1}-y^k) + (y^k-y^{k+1}) \end{array} \right\| \\
&\le \|T_k\|\|x^{k+1}-x^k\| + (1+\beta+\|S\|)\|y^{k+1}-y^k\| + \|x^{k+1}-y^k\| \\
&\le c_k(\|x^{k+1}-x^k\| + \|y^{k+1}-y^k\| + \|x^{k+1}-y^k\|),
\end{aligned}$$

where $c_k = \max\{\|T_k\|, 1+\beta+\|S\|, 1\}$. Since $\{T_k\}$ is bounded, then it follows from the above inequality that there exists a constant $\mu > 0$ such that

$$\|F(w^{k+1})\| \le \mu\left(\|u^{k+1}-u^k\|_{D_k}^2 + \|x^{k+1}-y^k\|^2\right).$$

□

Now, we begin to investigate the convergence of VMSP-ADMM. First, we assume some conditions for sequence $\{T_k\}$ (i.e., $\{D_k\}$) that should be obeyed to guarantee the convergence.

Condition 2.1. *For the sequence $\{T_k\}$ generated by the framework* (1.6)*, there exist $T \succeq 0$ and a non-negative sequence $\{\gamma_k\}$ such that*

1 $T \preceq T_{k+1} \preceq (1+\gamma_k)T_k, \ \ \forall k \geq 0,$

2 $\sum_{0}^{\infty} \gamma_k < \infty.$

From the definition of $\{D_k\}$, it follows that the sequence $\{D_k\}$ also satisfy $D \preceq D_{k+1} \preceq (1+\gamma_k)D_k$ for all k, where $D = \begin{pmatrix} T & 0 \\ 0 & S \end{pmatrix}$. We define two constants C_s and C_p as follows:

$$C_s := \sum_{k=0}^{\infty} \gamma_k \ \text{ and } \ C_p := \prod_{k=0}^{\infty} (1+\gamma_k). \tag{2.18}$$

From the assumption $\sum_0^\infty \gamma_k < \infty$ and $\gamma_k \geq 0$, it follows that $0 \leq C_s < \infty$ and $1 \leq C_p < \infty$. We can easily get

$$T \preceq T_k \preceq C_p T_0, \ \ \forall k \geq 0,$$

which means that the sequences $\{T_k\}$ and $\{D_k\}$ are bounded.

For convenience, we denote the following matrix:

$$G_k = \begin{pmatrix} T_k & 0 & 0 \\ 0 & S+\beta I & 0 \\ 0 & 0 & \frac{1}{\beta} I \end{pmatrix} \ \text{ and } \ \bar{G} = \begin{pmatrix} T & 0 & 0 \\ 0 & S+\beta I & 0 \\ 0 & 0 & \frac{1}{\beta} I \end{pmatrix}. \tag{2.19}$$

Obviously, $G_k \succeq \bar{G} \succeq 0$ since $T \succeq 0, S \succeq 0$ and $\beta > 0$.

Now we give the main theorem of this section.

Theorem 2.7. *Let $w^* = (x^*, y^*, \lambda^*) \in \Omega^*$, $\{w^k\}$ be generated by* (1.6) *and let $\{T_k\}$ be a sequence satisfying Condition 2.1. Then the sequence $\{w^k\}$ converges to a point $w^* \in \Omega^*$.*

Proof. First we show that the sequence $\{w^k\}$ is bounded.

It is trivial to know form (2.13) in Lemma 2.4 that $\lim_{k\to\infty} \left(\|u^{k+1} - u^k\|^2_{D_k} + \beta \|x^{k+1} - y^k\|^2 \right) = 0$, which indicates that

$$\lim_{k\to\infty} \|x^{k+1} - y^k\| = 0. \tag{2.20}$$

Besides, it is straightforward to see from (2.13) that $\|w^{k+1} - w^*\|^2_{G_k}$ is bounded where G_k is defined as (2.19). It shows that

$$\|x^{k+1} - x^*\|^2_{T_k}, \ \ \|y^{k+1} - y^*\|^2_{S+\beta I}, \ \ \|\lambda^{k+1} - \lambda^*\|^2$$

are all bounded. Obviously, we claim that $\{\lambda^k\}$ is bounded, and that $\{y^k\}$ is bounded as $S + \beta I \succ 0$. Note that $x^* = y^*$ and $\|x^{k+1} - x^*\| = \|x^{k+1} - y^k + y^k - y^*\| \leq \|x^{k+1} - y^k\| + \|y^k - y^*\|$. It then follows from (2.20) that $\|x^{k+1} - x^*\|$ is bounded, and hence $\{x^k\}$ is also bounded. Therefore, the sequence $\{w^k\}$ is bounded.

Next we show that any cluster point of the sequence $\{w^k\}$ is an optimal solution of (1.2).

Condition 2.1 implies that

$$0 \preceq G_{k+1} \preceq (1+\gamma_k) G_k,$$

and thus

$$\|w^{k+1} - w^*\|^2_{G_{k+1}} \leq (1+\gamma_k) \|w^{k+1} - w^*\|^2_{G_k}. \tag{2.21}$$

Using (2.19) and (2.13) in Lemma 2.4, we get

$$\|w^{k+1}-w^*\|_{G_k}^2 \le \|w^k-w^*\|_{G_k}^2 - (\|u^{k+1}-u^k\|_{D_k}^2 + \beta\|x^{k+1}-y^k\|^2). \tag{2.22}$$

Combine the inequality (2.21) with (2.22), we have that

$$\begin{aligned}\|w^{k+1}-w^*\|_{G_{k+1}}^2 &\le (1+\gamma_k)\|w^k-w^*\|_{G_k}^2 - (1+\gamma_k)c_1\left(\|u^{k+1}-u^k\|_{D_k}^2 + \|x^{k+1}-y^k\|^2\right)\\ &\le (1+\gamma_k)\|w^k-w^*\|_{G_k}^2 - c_1\left(\|u^{k+1}-u^k\|_{D_k}^2 + \|x^{k+1}-y^k\|^2\right).\end{aligned} \tag{2.23}$$

where $c_1 = \min\{1,\beta\}$.

It then follows from (2.23) that we have for all k,

$$\begin{aligned}\|w^{k+1}-w^*\|_G^2 &\le \|w^{k+1}-w^*\|_{G_{k+1}}^2\\ &\le (1+\gamma_k)\|w^k-w^*\|_{G_k}^2\\ &\vdots\\ &\le \left(\prod_{k=0}^{k}(1+\gamma_k)\right)\|w^0-w^*\|_{G_0}^2\\ &\le C_p\|w^0-w^*\|_{G_0}^2.\end{aligned} \tag{2.24}$$

From (2.23) and (2.24), we have

$$\begin{aligned}c_1\left(\|u^{k+1}-u^k\|_{D_k}^2 + \|x^{k+1}-y^k\|^2\right) &\le \|w^k-w^*\|_{G_k}^2 - \|w^{k+1}-w^*\|_{G_{k+1}}^2 + \gamma_k\|w^k-w^*\|_{G_k}^2\\ &\le \|w^k-w^*\|_{G_k}^2 - \|w^{k+1}-w^*\|_{G_{k+1}}^2 + \gamma_k C_p\|w^0-w^*\|_{G_0}^2.\end{aligned}$$

Then for all k, we obtain

$$\begin{aligned}\sum_{k=0}^{\infty} c_1\left(\|u^{k+1}-u^k\|_{D_k}^2 + \|x^{k+1}-y^k\|^2\right) &\le \|w^0-w^*\|_{G_0}^2 - \|w^{k+1}-w^*\|_{G_{k+1}}^2 + \left(\sum_{k=0}^{\infty}\gamma_k\right)C_p\|w^0-w^*\|_{G_0}^2\\ &\le (1+C_sC_p)\|w^0-w^*\|_{G_0}^2\\ &< \infty.\end{aligned}$$

Therefore, we have

$$\lim_{k\to\infty}\left(\|u^{k+1}-u^k\|_{D_k}^2 + \|x^{k+1}-y^k\|^2\right) = 0.$$

It follows from Lemma 2.6 that

$$\lim_{k\to\infty}\|F(w^{k+1})\| = 0.$$

We know that the sequence $\{w^k\}$ is bounded, so that it has at least one cluster point in Ω. Let $w^\infty \in \Omega$ be a cluster point of $\{w^k\}$, and let $\{w^{k_j}\}$ be a subsequence of $\{w^k\}$ that converges to point w^∞. Since ∂f and ∂g are upper semi-continuous, we can also get

$$\|F(w^\infty)\| = \lim_{j\to\infty}\|F(w^{k_j})\| = 0,$$

which shows $w^\infty \in \Omega^*$.

For any given $\epsilon > 0$ and $w^{k_j} \to w^\infty$, it follows that there exists positive integers q and $1 \le \delta < \infty$ such that

$$\|w^{k_q}-w^\infty\|_{G_{k_q}} < \frac{\epsilon}{\delta} \quad\text{and}\quad \left(\prod_{i=k_q}^{\infty}(1+\gamma_i)\right)^{1/2} < \delta. \tag{2.25}$$

Therefore, it follows from (2.24) and (2.25) that for any $k \geq k_q$,

$$\begin{aligned}\|w^k - w^\infty\|_{G_k} &\leq \left(\prod_{i=k_q}^{k-1}(1+\gamma_i)\right)^{1/2} \|w^{k_q} - w^\infty\|_{G_{k_q}} \\ &\leq \left(\prod_{i=k_q}^{\infty}(1+\gamma_i)\right)^{1/2} \|w^{k_q} - w^\infty\|_{G_{k_q}} < \epsilon.\end{aligned}$$

Therefore the whole sequence $\{w^k\}$ converges to w^∞. □

3 VMSP-ADMM with BFGS and its Convergence analysis

In this section, we first propose the updating of T_k via BFGS update and show a key property on T_k for the convergence. Then we establish the global convergence of the method as a corollary of the convergence results in Section 2.

3.1 Construction of the regularized matrix T_k via the BFGS update

As discussed in Introduction, we propose to construct T_k as $T_k = B_k - M$, where $M = \nabla^2_{xx}\mathcal{L}_\beta(x, y, \lambda)$. We want T_k to be positive semidefinite for global convergence as shown in the previous section. Moreover we want B_k to be as close to M as possible for rapid convergence. To this end, we propose to generate B_k by BFGS with respect to M.

Thus we may consider the BFGS update with a given $s \in R^n$ and $l = Ms$. Note that $s^\top l > 0$ when $s \neq 0$. Since BFGS usually constructs the inverse of B_k, let

$$H_k = B_k^{-1}.$$

Using H_k, we can easily solve subproblem (1.6a).

Now we briefly sketch BFGS and Limited memory BFGS. Let $s_k = x^{k+1} - x^k, l_k = Ms_k$. Then BFGS recursion for B_{k+1} and H_{k+1} are given as

$$B_{k+1}^{BFGS} = B_k + \frac{l_k l_k^\top}{l_k^\top s_k} - \frac{B_k s_k s_k^\top B_k^\top}{s_k^\top B_k s_k}$$

$$H_{k+1}^{BFGS} = \left(I - \frac{s_k l_k^\top}{s_k^\top l_k}\right) H_k \left(I - \frac{l_k s_k^\top}{s_k^\top l_k}\right) + \frac{s_k s_k^\top}{s_k^\top l_k}. \tag{3.1}$$

Since $s_k^\top l_k > 0$, B_{k+1}^{BFGS} and H_{k+1}^{BFGS} are positive definite whenever $B_k, H_k \succ 0$. Moreover

$$l_k = B_{k+1}^{BFGS} s_k \text{ and } s_k = H_{k+1}^{BFGS} l_k.$$

BFGS requires only matrix-vector multiplications which brings the computational cost at each iteration to $O(n^2)$ where n is the number of variables. If the number of variables is very large, even $O(n^2)$ per iteration is too expensive, both in terms of CPU time and sometimes also in terms of memory usage (a large matrix must be kept in memory at all times).

A less computationally intensive method when n is large is the limited memory BFGS method (L-BFGS), see [9, 13]. Instead of updating and storing the entire approximated inverse Hessian matrix, the L-BFGS method uses the last h iteration and uses only this input information. It means, we only need to store $s_k, s_{k-1}, ..., s_{k-h-1}$ and $l_k, l_{k-1}, ..., l_{k-h-1}$ to compute the update. The first h iterations, BFGS and L-BFGS generate the same result (assuming the initial setting for the two are identical). The updating in L-BFGS brings the computational cost down to $O(hn)$ per iteration. If $h \ll n$, this is effectively the same as $O(n)$.

3.2 Property of B_k via the BFGS update

Throughout this subsection, let $H_k = (B_k)^{-1}$. For the global convergence we need $T_k = B_k - M \succeq 0$, that is $B_k \succeq M$ where $M = \nabla^2_{xx}\mathcal{L}_\beta(x, y, \lambda)$. Note that $B_k \succeq M$ is equivalent to $H_k \preceq M^{-1}$. We will show that $H_k \preceq M^{-1}$ for all k when the initial matrix H_0 satisfies

$$H_0 \preceq M^{-1}.$$

We first show a technical lemma on s and l.

Lemma 3.1. *Let $s \in R^n$ such that $s \neq 0$. Moreover let $l = Ms$ and $\Phi = \{z \in R^n \mid \langle s, z\rangle = 0\}$. Then for any $v \in R^n$, there exist $c \in R$ and $z \in \Phi$ such that $v = cl + z$.*

Proof. Let $v \in R^n$. Then there exist $c_1, c_2 \in R$ and $z^1, z^2 \in \Phi$ such that $v = c_1 s + z^1$ and $l = c_2 s + z^2$. Since $s^\top l > 0$, $c_2 \neq 0$. Thus $s = \frac{1}{c_2} l - \frac{1}{c_2} z^2$. Substituting it into $v = c_1 s + z^1$ yields

$$v = c_1 \left(\frac{1}{c_2} l - \frac{1}{c_2} z^2\right) + z^1 = \frac{c_1}{c_2} l + z^1 - \frac{c_1}{c_2} z^2.$$

Let $c = \frac{c_1}{c_2}$ and $z = z^1 - \frac{c_1}{c_2} z^2$. Then $z \in \Phi$ and $v = cl + z$. □

Recall the BFGS recursion (3.1) is rewritten as

$$H_{next} = \left(I - \frac{sl^\top}{s^\top l}\right) H \left(l - \frac{ls^\top}{s^\top l}\right) + \frac{ss^\top}{s^\top l}, \tag{3.2}$$

where H is the proximal matrix for the current step and H_{next} is the new matrix generated via BFGS update with s and l for the next iteration. Moreover we have

$$H_{next} l = s = M^{-1} l. \tag{3.3}$$

The following theorem will play a key role in proving the convergence for our method.

Theorem 3.2. *Let $s \in R^n$ such that $s \neq 0$, and let $l = Ms$. If $H \preceq M^{-1}$, then $H_{next} \preceq M^{-1}$.*

Proof. Let v be an arbitrary nonzero vector in R^n. Let $\Omega = \{z \in R^n \mid \langle s, z\rangle = 0\}$. From Lemma 3.1 there exist $c \in R$ and $z \in \Omega$ such that $v = cl + z$. It then follows from (3.3) and the definition of z that

$$\begin{aligned} v^\top H_{\rm next} v &= (cl + z)^\top H_{\rm next}(cl + z) \\ &= c^2 l^\top s + 2cs^\top z + z^\top H_{\rm next} z \\ &= c^2 l^\top s + z^\top H_{\rm next} z \\ &= c^2 l^\top M^{-1} l + z^\top H_{\rm next} z. \end{aligned}$$

We now consider the last term of the right-hand side of the last equation. Since $z \in \Omega$, we have

$$z^\top \left(\frac{sl^\top}{s^\top l} H \frac{ls^\top}{s^\top l}\right) z = 0,$$

$$z^\top \left(\frac{sl^\top}{s^\top l} H\right) z = 0$$

and

$$z^\top \left(\frac{ss^\top}{s^\top l}\right) z = 0.$$

It then follows from (3.2) that

$$z^\top H_{\rm next} z = z^\top z - 2z^\top \left(\frac{sl^\top}{s^\top l} H\right) z + z^\top \left(\frac{sl^\top}{s^\top l} H \frac{ls^\top}{s^\top l}\right) z + \frac{z^\top s s^\top z}{s^\top l} = z^\top H z.$$

Moreover equation (3.3) implies

$$l^\top M^{-1} z = s^\top z = 0.$$

Consequently we have

$$\begin{aligned} v^\top H_{\rm next} v &= c^2 l^\top M^{-1} l + z^\top H z \\ &\leq c^2 l^\top M^{-1} l + z^\top M^{-1} z \\ &= (cl+z)^\top M^{-1}(cl+z) - 2cl^\top M^{-1} z \\ &= v^\top M^{-1} v, \end{aligned}$$

where the inequality follows from the assumption. Since v is arbitrary, we have $H_{\rm next} \preceq M^{-1}$. □

This theorem shows that if $H_0 \preceq M^{-1}$, then $H_k \preceq M^{-1}$, and hence $T_k \succeq 0$.

3.3 Convergence of the proposed method with BFGS

Based on the description of the BFGS (or L-BFGS) update, we first give our algorithm (ADM-BFGS) in detail.

Algorithm 1: Variable metric semi-proximal ADMM with the BFGS update (ADM-BFGS)

Input : size (m, n), data matrix A, initial point (x^0, y^0, λ^0), penalty parameter β, maxIter; initial matrix $H_0 \preceq M^{-1}$, constant $\bar{k} \in [1, \infty]$, stopping criterion ϵ.

Output: approximative solution (x^k, y^k, λ^k)

initialization;
while $k < \mathrm{maxIter}$ *or not convergence* **do**
  if $k \leq \bar{k}$ **then**
    update H_k via BFGS (or L-BFGS) with the initial matrix H_0;
  else
    $H_k = H_{k-1}$;
  end
  update the x^{k+1} by solving the $x-$subproblem: $x^{k+1} = x^k + H_k\left(\lambda^k + \beta y^k + A^\top b - M x^k\right)$;
  update the y^{k+1} by solving the $y-$subproblem:
  $y^{k+1} = \arg\min_y \left\{ g(y) - \langle \lambda^k, x^{k+1} - y\rangle + \frac{\beta}{2}\|x^{k+1} - y\|^2 + \frac{1}{2}\|y - y^k\|_S^2 \right\}$;
  update the augmented lagrangian parameter: $\lambda^{k+1} = \lambda^k - \beta(x^{k+1} - y^{k+1})$.
end

Now we will give the convergence of ADM-BFGS.

Theorem 3.3. *Suppose that there exists a constant $\bar{k}$, $0 < \bar{k} \leq \infty$, B_k updating by the BFGS procedure stopped at $\bar{k}$ and the sequence $\{T_k\}$ and $\{\gamma_k\}$ satisfy the Condition 2.1. Let $w^* = (x^*, y^*, \lambda^*) \in \Omega^*$, $\{w^k\}$ be generated by the Algorithm 1. Then the sequence $\{w^k\}$ converges.*

Proof. The convergence directly follows from Theorem 2.7. □

Note that the updating of B_k stopped at certain finite $\bar{k}$, that is

$$B_{k+1} = B_k,\ T_{k+1} = T_k \quad \text{for all } k \geq \bar{k},$$

i.e., $\gamma_k = 0$ when $k \geq \bar{k}$. Thus, it is reasonable to say that the sequence $\{T_k\}$ generated by the Algorithm 1 and some existing $\{\gamma_k\}$ satisfy the Condition 2.1.

4 Numerical results

In this section, we demonstrate the potential efficiency of our method by some numerical experiments. We consider the Lasso problem

$$\min_{x\in\mathbb{R}^n} \frac{1}{2}\|Ax-b\|_2^2 + \tau\|x\|_1, \quad \tau > 0, \tag{4.1}$$

where

- $A \in \mathbb{R}^{m\times n}$ is the given data matrix,
- $x \in \mathbb{R}^n$ is the vector of feature coefficients to be estimated,
- $b \in \mathbb{R}^m$ is the observation vector and $\tau \in \mathbb{R}$ is the regularization parameter,
- m is the number of the data points, and n is the number of features.

By introducing an auxiliary variable $y \in \mathbb{R}^n$, we reformulate the problem (4.1) as

$$\min_{x\in\mathbb{R}^n,\ y\in\mathbb{R}^n} \frac{1}{2}\|Ax-b\|_2^2 + \tau\|y\|_1, \quad \text{s.t.} \quad x - y = 0. \tag{4.2}$$

The stopping criterion used same as in [1] for all the numerical experiments is that the primal and dual residuals must be small, i.e.,

$$\|r^k\|_2 \leq \epsilon_k^{\text{pri}} \ \text{ and } \ \|\sigma^k\|_2 \leq \epsilon_k^{\text{dual}}, \tag{4.3}$$

where $r^k = x^k - y^k$ is the primal residual and $\sigma^k = -\beta(y^k - y^{k-1})$ is the dual residual at the iteration k; $\epsilon^{\text{pri}} > 0$ and $\epsilon^{\text{dual}} > 0$ are feasibility tolerances for the primal and dual feasibility conditions, respectively. These tolerances can be chosen using an absolute and relative criterion from the suggestion in [1], such as

$$\begin{aligned} \epsilon_k^{\text{pri}} &= \sqrt{n}\epsilon^{\text{abs}} + \epsilon^{\text{rel}}\max\{\|x^k\|_2, \|-y^k\|_2\}, \\ \epsilon_k^{\text{dual}} &= \sqrt{n}\epsilon^{\text{abs}} + \epsilon^{\text{rel}}\|\lambda^k\|_2, \end{aligned}$$

where $\epsilon^{\text{abs}} > 0$ is an absolute tolerance and $\epsilon^{\text{rel}} > 0$ is a relative tolerance. The choice of stopping criterion depends on the scale of the variable values.

In our implementation, we always choose $S = 0$ and $\beta = 1$. Starting from $x^0 = y^0 = 0$ and $\lambda^0 = 0$, with some suitably chosen proximal matrix T (T_k). We will test several data under different proximal terms and stopping criterions.

4.1 Test I: Comparison among three different methods

In the subsection, we test some random data for matrix A using four different methods: classical ADMM [6, 7](ADM-OPT), proximal ADMM including semi-proximal ADMM in [5] denoted as ADM-SPRO, and indefinite proximal ADMM based on [12] named as ADM-IPRO; and ADMM with BFGS (ADM-BFGS). ADM-OPT solves the original subproblem (1.4a) exactly using $(A^\top A + \beta I)^{-1}$. Note that $(A^\top A + \beta I)^{-1}$ is calculated only once, but it may not be available for large scale problems. First we test with the proximal terms chosen as following:

1. SPRO: An semi-definite proximal matrix T as

$$T = \xi I - \beta I - A^{\mathsf{T}}A, \text{ with } \xi = 1.01 * \lambda_{\max}\left(\beta I + A^{\mathsf{T}}A\right).$$

2. IPRO: An indefinite proximal matrix T as

$$T = \xi I - \beta I - A^{\mathsf{T}}A, \text{ with } \xi = \lambda_{\max}\left(\beta I + A^{\mathsf{T}}A\right).$$

3. BFGS: An semidefinite proximal matrix sequence T_k with P_k^{-1} generated by BFGS (3.1), the initial matrix

$$B_0 = \gamma I, \gamma = 1.01 * \lambda_{\max}(\beta I + A^{\top}A).$$

Now we are at the stage of conducting numerical simulations with different selections of data matrix A and different sparsity density. Given n, $m = n/2$ and a $p-$sparse vector $\bar{x} \in \mathbb{R}^n$ (p is the number of nonzero elements in x over n). The stopping criterions are chosen as $\epsilon^{\text{abs}} = 10^{-4}, \epsilon^{\text{rel}} = 10^{-3}$.

- Test 1. sparsity density $p = 0.1$, with A is drawn from random and then all columns of A are normalized. Matlab codes for generating data are given as

```
xbar = sprandn(n,1,p);   A = randn(m,n);
A = A*spdiags(1./sqrt(sum(A.^2))',0,n,n); % normalize columns
b = A*xbar + sqrt(0.001)*randn(m,1);
mu = 0.1* norm(A'*b, 'inf').
```

- Test 2. sparsity density $p = 0.1$ with the same sparsity density matrix A under an $N(0, 1)$ distribution;

```
A = sprandn(m,n,p); % N(0,1) with the density p
```

- Test 3. sparsity density $p = 0.5$ with the same sparsity density matrix A under an $N(0, 1)$ distribution. Other settings are same with above tests.

We have solved 10 problems in each test. Table 1 shows the average of iterative steps and the time of classical ADMM, proximal ADMM and ADMM with BFGS.

Table 1: Comparison on iteration steps and CPU time (seconds) among the three methods

	Dim		ADM-OPT		ADM-SPRO		ADM-IPRO		ADM-BFGS		
	m	n	Iter.	Time	Iter.	Time	Iter.	Time	Iter.	Time Total	Time-H
Test 1	1000	2000	19.0	0.21	63.2	0.84	63.3	0.74	39.5	5.06	4.31
Test 2	1000	2000	1486.4	5.83	2406.1	2.97	2461.8	2.87	1491.0	177.49	168.70
Test 3	1000	2000	5079.7	31.03	12189.3	56.77	12128.5	56.45	5094.7	603.99	562.56

"Time-H" for ADM-BFGS in the above table means the time (seconds) to compute the approximation matrix H_k of the inverse of Hessian matrix. From the above results, it is obviously to see that the classical ADMM admits the fastest method to catch the objective function value both at iterative steps and CPU times, while the ADMM with BFGS is better than the proximal ADMM at the iterations but spends much time to compute the H_k. When the data matrix A is ill condition or it is impossible to compute the inverse of Hessian matrix of augmented Lagrangian function, it is meaningful to use the H_k so that it can get the optimization result as the almost same iterative steps as the classical ADMM. Another problem is that BFGS needs more memory to save the H_k, thus we next will consider to use the L-BFGS to construct the H_k. The memory size can be chosen as some certain m.

4.2 Test II: ADMM with Limited memory BFGS

In the following implementation, we test how the ADMM with limited memory BFGS (ADM-LBFGS) works. From the above results in Table 1, we know that ADM-BFGS can get the solution as the same iteration steps as classical ADMM, thus it works better when A is very large and ill condition which means it is difficult to compute the inverse of Hessian matrix of augmented lagrangian function. Also, it should work better than normal proximal ADMM when we chose some general proximal terms for the proximal ADMM not like the choices in subsection 4.1.

We chose the semidefinite proximal matrix T as

$$T = \xi I - \beta I - A^{\mathsf{T}}A, \text{ with } \xi = \kappa * \left(\lambda_{\max}(A^{\top}A) + \beta\right). \tag{4.4}$$

The initial semidefinite proximal matrix generated by Limited memory BFGS chosen as

$$B_0 = \gamma I, \gamma = \kappa * \left(\lambda_{\max}(A^{\top}A) + \beta\right). \tag{4.5}$$

Note that $H_0 = \frac{1}{\gamma}I$. The L-BFGS can save a lot of storage memory during the updated proceeds. We test these problems with the time only for algorithms(without objective function calculation). Fix $H_0^k = H_0$ for every updating of L-BFGS matrix.

We use different "memory" for the L-BFGS update: $h = 10, 20, 30$, namely ADM-L10, ADM-L20, ADM-L30 respectively. Thus the comparison is among the classical ADMM, proximal ADMM and these proposed three types of ADM-LBFGS.

For the detail of this experiment, we set $\kappa = 1.01$ in above (4.4) and (4.5). The matrix A under the $N(0,1)$ distribution with different sizes: $A \in \mathbb{R}^{1000\times3000}$, $A \in \mathbb{R}^{2000\times3000}$, $A \in \mathbb{R}^{3000\times3000}$. The sparsity p for matrix A and vector $\bar{x}$ is chosen as $p = 0.1, 0.5$ and $p = 0.1, 0.2$ for $A \in \mathbb{R}^{5000\times5000}$. Also we set the stopping criterions as $\epsilon^{\text{abs}} = 10^{-4}$, and $\epsilon^{\text{rel}} = 10^{-2}$. The maximum iterations are set to be 20000 in all experiments.

The results of iteration steps and CPU time (seconds) are averaged over 10 random trials showed on Table 2.

Table 2: Comparison among classical ADM solution, proximal solution and ADM with L-BFGS

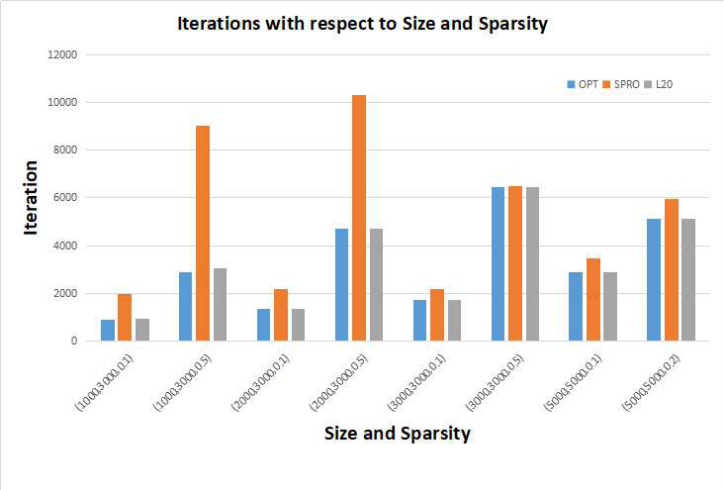
Dim			ADM-OPT		ADM-SPRO		ADM-L10		ADM-L20		ADM-L30	
m	n	p	Iter.	Time	Iter.	Time	Iter.	Time	Iter.	Time	Iter.	Time
1000	3000	0.1	915.3	2.16	1972.3	1.11	953.8	0.86	941.3	1.06	953.0	1.30
1000	3000	0.5	2881.1	13.20	9031.1	25.83	3257.5	10.84	3036.9	10.88	3111.2	11.83
2000	3000	0.1	1335.7	10.33	2175.1	2.79	1337.9	2.39	1338.4	2.84	1338.7	3.15
2000	3000	0.5	4703.7	57.59	10293.5	61.93	4723.0	30.42	4727.6	31.50	4727.1	32.48
3000	3000	0.1	1724.3	22.48	2185.0	4.82	1725.3	4.68	1724.5	5.07	1725.3	5.47
3000	3000	0.5	6439.0	83.10	6491.2	56.94	–	–	6441.3	59.41	6440.7	60.41
5000	5000	0.1	2886.5	110.20	3478.8	24.91	2886.9	22.46	2887.0	24.06	2886.9	27.14
5000	5000	0.2	5119.1	200.10	5957.5	80.87	5159.6	72.28	5119.8	72.82	5119.1	77.70

'$-$' means that the method cannot get the optimal solution within 20000 steps.

In order to see the iterations comparison among these methods clearly, we also plot the histogram of iterations with respect to size and sparsity of matrix A with algorithms ADM-OPT, ADM-SPRO and ADM-L20 in Figure 1.

From the above results, we know that when the "memory" = 10, 20 or 30, ADM-LBFGS performs best and stability enough. The normal proximal ADMM fast but takes much more iteration steps and the result precision is worse than the other methods. The BFGS and L-BFGS algorithms both can reach the optimization result by the same level iteration steps for the large scale problems. At the same time, the L-BFGS can save a lot of time comparing with BFGS algorithm. Especially, if the data matrix A is very large and ill condition in the real application, the classical ADMM cannot be used because the inverse of $A^{\top}A$ information is necessary for the classical method but not ADM-BFGS (or ADM-LBFGS).

Figure 1: Iterations with respect to size and sparsity of ADM-OPT, ADM-SPRO and ADM-L20



In the above experiments, we chosen $\kappa = 1.01$. This is unrealistic for some large scale applications where the calculation of maximum eigenvalue is expensive. Next we will test the behaviour of ADM-LBFGS with different parameters in H_0, that is, $\kappa = 1.01, 5.0, 10.0, 100$ in above (4.4) and (4.5). For every κ, we will test 10 random trials. Other settings are as $A \in \mathbb{R}^{3000\times 3000}$ with the sparsity $p = 0.1$, the stopping criterions as $\epsilon^{\rm abs} = 10^{-4}, \epsilon^{\rm rel} = 10^{-2}$.

Table 3: Different κ for proximal ADMM

	ADM-SPRO		ADM-L10		ADM-L20		ADM-L30	
κ	Iter.	Time	Iter.	Time	Iter.	Time	Iter.	Time
1.01	2188.3	4.99	1735.2	4.71	1734.9	5.19	1735.2	5.52
5.0	4726.1	9.92	1700.5	4.61	1699.3	5.08	1698.7	5.49
10.0	6569.9	13.22	8964.7*	20.46	1685.0	4.96	1683.5	5.34
100	–	–	1704.5	4.53	1684.8	4.91	1681.5	5.39

'$-$' means that the method cannot get the optimal solution within 20000 steps.

Remark 4.1. *"*": For this case, the ADM-L10 sometimes cannot stop within 20000 steps and we only average the successful trials. In the above table, these 10 random trials results are* 100% *successful. The ADM-L10 is not stability only when the* $\kappa = 10.00$ *while the ADM-L20 and ADM-L30 always work better. We can choose the "memory" freely due to the size and sparsity of the different problems. The normal proximal ADMM performs worse when the* κ *is big, especially, the* $\kappa = 100$*, the ADM-SPRO cannot stop within 20000 steps.*

From all the above results we conclude that

1. Compare with the classical ADMM (ADM-OPT), the proposed method (ADM-L20) is suitable for dense large scale problems;
2. Compare with the general proximal ADMM (ADM-SPRO), the proposed method is suitable when accurate estimation of maximum eigenvalues is difficult/expensive or evaluating Ax is expensive.

4.3 Test III: Behaviors of ADM-LBFGS that stops updating of H_k for some different $\bar{k}$

Theorem 3.3 says that the ADMM with BFGS converges when we stop the update of H_k for sufficiently large $\bar{k}$, that is, $H_k = H_{\bar{k}}$ for $k \geq \bar{k}$. Therefore, we test the behavior of ADM-LBFGS with various $\bar{k}$ when the H_k

stopped. We choose the $\bar{k}$ as $\{100, 500, 1000, 1500, 2000, \infty\}$. Note that "$\infty$" means that we update H_k for all k.

The matrix $A \in \mathbb{R}^{3000\times 3000}$ chosen under an $N(0,1)$ distribution with sparsity density $p = 0.1$, all the other initial information are same with the above Tests. The stopping criterion also uses the $\epsilon^{\rm abs} = 10^{-4}, \epsilon^{\rm rel} = 10^{-2}$. The parameter κ in proximal term is 1.01. The maximum iterations are set to be 20000 in the following experiments.

For every random matrix A, we test the different memory ADM-LBFGS with different $\bar{k}$ at the same time so that the results are more reliable. The detail results of CPU time and iterations of different stopping $\bar{k}$ averaged over 10 random trials are provided in Table 4.

Table 4: Results for stopping at different $\bar{k}$

	ADM-L10		ADM-L20		ADM-L30	
$\bar{k}$	Iter.	Time	Iter.	Time	Iter.	Time
100	2215.2	4.80	2188.1	4.71	2224.3	4.89
500	1825.9	4.27	1796.7	4.34	1795.4	4.49
800	1795.1	4.29	1786.6	4.47	1787.5	4.72
1000	1789.7	4.38	1785.6	4.63	1785.5	4.90
1500	1785.8	4.58	1784.9	4.96	1785.0	5.40
2000	1784.8	4.69	1784.5	5.13	1784.8	5.67
∞	1784.6	4.68	1784.4	5.19	1784.7	5.65

From the above results, we can get that for all $\bar{k}$, the ADM-LBFGS with different memory storage can catch a solution within the maximum iteration. If the memory is 10 (smaller), it is better to update H_k more times like choosing $\bar{k}$ from 800 to 1000. And for the memory is 20 or 30 (larger), it is good enough to chose the $\bar{k}$ from 500 to 800 in this case.

5 Conclusions

In this paper, we have proposed a special proximal ADMM where the proximal matrix derived from the BFGS or Limited memory BFGS method. The convergence of such methods have also been established under some certain conditions. Numerical results on several random problems with the large scale data are given to illustrate the effectiveness of the proposed method.

We have not considered the general convex optimization problem. This is because Theorem 3.2 holds only when the Hessian matrix of the augmented Lagrangian function, that is, $M = \nabla^2_{xx}\mathcal{L}_\beta(x, y, \lambda)$ is a constant matrix. As a future work, we will consider more general problems by ADMM with BFGS update whose x-subproblems become unconstrained quadratic programming problem as in this paper. Then we may apply Theorem 3.2 for global convergence.

References

[1] S. BOYD, N. PARIKH, E. CHU, B. PELEATO, AND J. ECKSTEIN, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Foundations and Trends® in Machine Learning, 3 (2011), pp. 1–122.

[2] W. DENG AND W. YIN, *On the global and linear convergence of the generalized alternating direction method of multipliers*, tech. report, DTIC Document, 2012.

[3] J. DOUGLAS AND H. RACHFORD, *On the numerical solution of heat conduction problems in two and three space variables*, Transactions of the American mathematical Society, (1956), pp. 421–439.

[4] J. ECKSTEIN AND M. FUKUSHIMA, *Some reformulations and applications of the alternating direction method of multipliers*, in Large scale optimization, Springer, 1994, pp. 115–134.

[5] M. FAZEL, T. K. PONG, D. SUN, AND P. TSENG, *Hankel matrix rank minimization with applications to system identification and realization*, SIAM Journal on Matrix Analysis and Applications, 34 (2013), pp. 946–977.

[6] D. GABAY AND B. MERCIER, *A dual algorithm for the solution of nonlinear variational problems via finite element approximation*, Computers & Mathematics with Applications, 2 (1976), pp. 17–40.

[7] R. GLOWINSKI AND A. MARROCO, *Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires*, ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique, 9 (1975), pp. 41–76.

[8] M. L. N. GONÇALVES, M. M. ALVES, AND J. G. MELO, *Pointwise and ergodic convergence rates of a variable metric proximal alternating direction method of multipliers*, Journal of Optimization Theory and Applications, 177 (2018), pp. 448–478.

[9] J. K. HALE AND S. M. V. LUNEL, *Introduction to functional differential equations*, vol. 99, Springer Science & Business Media, 2013.

[10] B. HE, L.-Z. LIAO, D. HAN, AND H. YANG, *A new inexact alternating directions method for monotone variational inequalities*, Mathematical Programming, 92 (2002), pp. 103–118.

[11] B. HE AND X. YUAN, *On the $O(1/n)$ convergence rate of the Douglas-Rachford alternating direction method*, SIAM Journal on Numerical Analysis, 50 (2012), pp. 700–709.

[12] M. LI, D. SUN, AND K.-C. TOH, *A majorized admm with indefinite proximal terms for linearly constrained convex composite optimization*, SIAM Journal on Optimization, 26 (2016), pp. 922–950.

[13] J. NOCEDAL, *Updating quasi-newton matrices with limited storage*, Mathematics of computation, 35 (1980), pp. 773–782.

[14] R. TIBSHIRANI, *Regression shrinkage and selection via the lasso*, Journal of the Royal Statistical Society. Series B (Methodological), (1996), pp. 267–288.

[15] M. XU AND T. WU, *A class of linearized proximal alternating direction methods*, Journal of Optimization Theory and Applications, 151 (2011), pp. 321–337.