# Faster Alternating Direction Method of Multipliers with a Worst-case $O(1/n^2)$ Convergence Rate

WENYI TIAN* XIAOMING YUAN†

July 17, 2016

**Abstract.** The alternating direction method of multipliers (ADMM) is being widely used for various convex programming models with separable structures arising in specifically many scientific computing areas. The ADMM's worst-case $O(1/n)$ convergence rate measured by the iteration complexity has been established in the literature when its penalty parameter is a constant, where $n$ is the iteration counter. Research on ADMM's worst-case $O(1/n^2)$ convergence rate, however, is still in its infancy. In this paper, we suggest applying a rule proposed recently by Chambolle and Pock to iteratively update the penalty parameter and show that ADMM with this adaptive penalty parameter has a worst-case $O(1/n^2)$ convergence rate. Without strong convexity requirement on the objective function, our assumptions on the model are mild and can be satisfied by some representative applications. We test the LASSO model and numerically verify the significant acceleration effectiveness of the faster ADMM with a worst-case $O(1/n^2)$ convergence rate. Moreover, the faster ADMM is more user-favorable than the ADMM with a constant penalty parameter in senses of that it can pursue solutions with very high accuracy and that it is not sensitive to the initial value of the penalty parameter.

**Keywords.** Convex programming, Alternating direction method of multipliers, Convergence rate, Acceleration, First order methods

## 1 Introduction

Many applications can be modeled as convex minimization problems with certain separable structures. For example, their objective functions may be separable and representable by a sum of two or more functions. A representative case is where one function represents a data-fidelity term and the other is a regularization term; this case arises frequently in areas such as inverse problems, image processing and machine learning. For such a separable convex minimization model, the alternating direction method of multipliers (ADMM) proposed originally in [18] (see also [6, 16]) turns out to be a benchmark solver and it is being widely used for many applications in a broad spectrum of areas. We refer the reader to [3, 14, 17] for some review papers on ADMM.

The convergence of ADMM has been well studied in earlier literature, e.g., [15, 16], and recently its worst-case $O(1/n)$ convergence rate measured by the iteration complexity has also been established in [24, 25]. Here, $n$ is the iteration counter and we refer to

*Center for Applied Mathematics, Tianjin University, Tianjin 300072, China. Email: twymath@gmail.com

†Department of Mathematics, Hong Kong Baptist University, Hong Kong, China. This author was partially supported by the General Research Fund from Hong Kong Research Grants Council: HKBU 12300515. Email: xmyuan@hkbu.edu.hk

[30, 31, 32] for some seminal work of convergence rate analysis in terms of the iteration complexity. The main goal of this paper is to investigate under which scenario the ADMM has a faster worst-case $O(1/n^2)$ convergence rate. Note that it still remains open whether or not the ADMM can achieve a worst-case $O(1/n^2)$ convergence rate under the general setting where no special assumptions on the model are posed. This can be partially understood by the result in [8] (see Theorem 11 therein).

To discuss the possibility of a worst-case $O(1/n^2)$ convergence rate for ADMM, we concentrate on the convex minimization model

$$\min_{x\in\mathcal{X}} \ f(x)+g(Ax) \tag{1.1}$$

where $\mathcal{X}\subset\mathbb{R}^d$ is closed and convex, $f:\mathcal{X}\to(-\infty,\infty]$ and $g:\mathbb{R}^m\to(-\infty,\infty]$ are closed, proper, and convex functions, and $A\in\mathbb{R}^{m\times d}$ is full column rank. Throughout the solution set of (1.1) is assumed to be nonempty. The model (1.1) can be written as

$$\begin{aligned} \min \ & f(x)+g(y)\\ \text{s.t.} \ & Ax-y=0\\ & x\in\mathcal{X}, y\in\mathbb{R}^m, \end{aligned} \tag{1.2}$$

where $y\in\mathbb{R}^m$ is an auxiliary variable. Then, the iterative scheme of ADMM for (1.2) reads as

$$\begin{cases} x_{n+1}=\underset{x\in\mathcal{X}}{\operatorname{argmin}}\Big\{f(x)+\dfrac{\sigma}{2}\big\|Ax-y_n+\dfrac{\lambda_n}{\sigma}\big\|^2\Big\}\\ y_{n+1}=\underset{y\in\mathbb{R}^m}{\operatorname{argmin}}\Big\{g(y)+\dfrac{\sigma}{2}\big\|Ax_{n+1}-y+\dfrac{\lambda_n}{\sigma}\big\|^2\Big\}\\ \lambda_{n+1}=\lambda_n+\sigma(Ax_{n+1}-y_{n+1}), \end{cases} \tag{1.3}$$

with $\sigma>0$ the penalty parameter and $\lambda\in\mathbb{R}^m$ the Lagrange multiplier. There are different ways to understand the ADMM. For example, it can be regarded as a splitting version of the classical augmented Lagrangian method in [26, 33]; it was also explained in [15] as an application of the Douglas-Rachford splitting method (DRSM), which was first proposed in [11] for linear heat equations and then generalized in [27] to the nonlinear case, to the dual problem of (1.1); and it was further analyzed in [13] that the ADMM is an application of the proximal point algorithm (PPA) in [28, 29] from the maximal monotone operator perspective.

An important variant of ADMM is the proximal version

$$\begin{cases} x_{n+1}=\underset{x\in\mathcal{X}}{\operatorname{argmin}}\Big\{f(x)+\dfrac{\sigma}{2}\big\|Ax-y_n+\dfrac{\lambda_n}{\sigma}\big\|^2+\dfrac{1}{2}\|x-x_n\|_Q^2\Big\}\\ y_{n+1}=\underset{y\in\mathbb{R}^m}{\operatorname{argmin}}\Big\{g(y)+\dfrac{\sigma}{2}\big\|Ax_{n+1}-y+\dfrac{\lambda_n}{\sigma}\big\|^2\Big\}\\ \lambda_{n+1}=\lambda_n+\sigma(Ax_{n+1}-y_{n+1}), \end{cases} \tag{1.4}$$

where $Q\in\mathbb{R}^{d\times d}$ is a positive definite matrix; see e.g. [12, 21]. In particular, when $Q=\mu I-\sigma A^TA$ with $\mu>\sigma\|A^TA\|$, it is easy to see that the $x$-subproblem in (1.4) reduces to

$$x_{n+1}=\underset{x\in\mathcal{X}}{\operatorname{argmin}}\Big\{f(x)+\frac{\mu}{2}\big\|x-x_n+\frac{1}{\mu}A^T\big(\lambda_n+\sigma(Ax_n-y_n)\big)\big\|^2\Big\}.$$

When $\mathcal{X} = \mathbb{R}^d$, the minimization problem in the equation above amounts to computing the proximal operator of $f$:

$$\text{prox}_{\gamma f}(x) := \underset{y}{\text{argmin}}\Big\{ f(y) + \frac{1}{2\gamma}\|y - x\|^2 \Big\}, \tag{1.5}$$

with $\gamma > 0$. Note that the proximal operator (1.5) has a closed-form solution for some interesting cases such as $f(x) = \|x\|_1$. In this case, the proximal version (1.4) reduces to the linearized version of ADMM for (1.2):

$$\begin{cases} x_{n+1} = \underset{x\in\mathcal{X}}{\text{argmin}}\Big\{ f(x) + \frac{\mu}{2}\big\|x - x_n + \frac{1}{\mu}A^T\big(\lambda_n + \sigma(Ax_n - y_n)\big)\big\|^2 \Big\} \\ y_{n+1} = \underset{y\in\mathbb{R}^m}{\text{argmin}}\Big\{ g(y) + \frac{\sigma}{2}\big\|Ax_{n+1} - y + \frac{\lambda_n}{\sigma}\big\|^2 \Big\} \\ \lambda_{n+1} = \lambda_n + \sigma(Ax_{n+1} - y_{n+1}), \end{cases} \tag{1.6}$$

We refer to, e.g., [39, 40, 42], for some efficient applications of the linearized version of ADMM. Note that we only consider the case where the $x$-subproblem is proximally regularized because it is useful enough for most of the ADMM's applications. Technically, there is no difficulty if both the subproblems are proximally regularized, see e.g., [21].

In [4], the problem (1.1) with $\mathcal{X} = \mathbb{R}^d$ was written as the saddle-point problem

$$\min_{x\in\mathbb{R}^d}\max_{\lambda\in\mathbb{R}^m}\big\{ f(x) + (Ax, \lambda) - g^*(\lambda)\big\}, \tag{1.7}$$

where $g^*(\lambda) := \sup_y\{(y, \lambda) - g(y)\}$ is the Fenchel conjugate of $g(y)$ (see, e.g., [34]). Then, the generalized primal-dual algorithm was proposed to solve (1.7):

$$\begin{cases} \lambda_{n+1} = \text{prox}_{\sigma g^*}\big(\lambda_n + \sigma A\tilde{x}_n\big) \\ x_{n+1} = \text{prox}_{\tau f}\big(x_n - \tau A^T\lambda_{n+1}\big) \\ \tilde{x}_{n+1} = x_{n+1} + \theta(x_{n+1} - x_n), \end{cases} \tag{1.8}$$

where "prox" is defined in (1.5), $\theta \in [0, 1]$ is a combination parameter, $\tau > 0$ and $\sigma > 0$ are two constants satisfying $\tau\sigma\|A\|^2 < 1$ when $\theta = 1$. For the scheme (1.8) with $\theta = 1$, its worst-case $O(1/n)$ convergence rate in the ergodic sense was established in [4]. Then, the scheme (1.8) was extended in [23] with the combination parameter $\theta \in [-1, 1]$; some correction steps were combined with the primal-dual step and the convergence was established under the condition

$$\tau\sigma\frac{(1+\theta)^2}{4}\|A^TA\| < 1.$$

The convergence and convergence rate for scheme (1.8) with $\theta \in [-1, 1]$ were further established in [22, 37] if $f$ is strongly convex. As analyzed in [4], the primal-dual algorithm (1.8) with $\theta = 1$ is equivalent to the application of the linearized (also called preconditioned) version of ADMM with $Q = \sigma^{-1}I - \tau AA^T$ and $\tau\sigma\|A\|^2 < 1$.

Moreover, in [4] (see also [5]), the authors suggested choosing the involved parameters $\theta$, $\tau$ and $\sigma$ dynamically, instead of constants, in (1.8). That is, consider the generalized primal-dual algorithm with dynamical parameters:

$$\begin{cases} \lambda_{n+1} = \text{prox}_{\sigma_n g^*}\big(\lambda_n + \sigma_n A\tilde{x}_n\big) \\ x_{n+1} = \text{prox}_{\tau_n f}\big(x_n - \tau_n A^T\lambda_{n+1}\big) \\ \tilde{x}_{n+1} = x_{n+1} + \theta_{n+1}(x_{n+1} - x_n). \end{cases} \tag{1.9}$$

A worst-case $O(1/n^2)$ convergence rate of (1.9) in the ergodic sense was established in [4] provided that $f$ is further assumed to be strongly convex with the modulus $\gamma$ and these parameters satisfy the conditions

$$\theta_{n+1} = \frac{1}{\sqrt{1+2\gamma\tau_n}},\ \tau_{n+1} = \theta_{n+1}\tau_n,\ \sigma_{n+1} = \sigma_n/\theta_{n+1}. \tag{1.10}$$

It turns out that these conditions are crucial for establishing the desired $O(1/n^2)$ convergence rates in [4, 5]. Thus, compared with the primal-dual scheme (1.8) with constant parameters, the scheme (1.10) possesses a higher convergence rate in terms of the iteration complexity despite that it additionally requires to determine three parameter sequences. These existing results strongly inspire us to consider under which conditions the ADMM (1.3) with dynamically adjusted penalty parameters has a worst-case $O(1/n^2)$ convergence rate.

In the literature, there are some works related to how to derive a worst-case $O(1/n^2)$ convergence rate for the ADMM; which either require stronger assumptions or are eligible only for some variants of the original ADMM scheme (1.3). In [9, 19], the following more general problem was considered:

$$\begin{aligned} &\min\ f(x)+g(y)\\ &\text{s.t.}\ Ax+By=b, \end{aligned} \tag{1.11}$$

where $f:\mathbb{R}^{d_1}\to(-\infty,\infty]$ and $g:\mathbb{R}^{d_2}\to(-\infty,\infty]$ are closed, proper and convex functions, $A\in\mathbb{R}^{m\times d_1}$, $B\in\mathbb{R}^{m\times d_2}$ and $b\in\mathbb{R}^m$. It was shown in [9] that if $g$ is strongly convex and the ADMM's penalty parameter $\sigma$ satisfies the condition $\sigma<\kappa\upsilon_g/\|B\|^2$ where $\upsilon_g$ is the strong convexity modulus of $g$ and $\kappa(\approx 1.24698)$ is the positive root of the equation $z^3+z^2-2z-1$, then the residual of the constraints in (1.11) is decreasing in order of $o(1/n^2)$ in a nonergodic sense while the measurement of the error of the objective function in the primal model (1.11) is still in order of $o(1/n)$[1]. In [19], it was shown that if both $f$ and $g$ are strongly convex with $g$ being further assumed to be quadratic; and if the penalty parameter $\sigma$ is chosen as

$$\sigma^3 < \frac{\upsilon_f\upsilon_g^2}{\rho(A^TA)\rho(B^TB)^2},$$

where $\upsilon_f$ and $\upsilon_g$ are the strong convexity modulus of $f$ and $g$, respectively; and $\rho(\cdot)$ is the spectral radius of a matrix, then the acceleration step proposed in [32] can be combined

[1]In our discussion, for simplicity, we do not differentiate the $O(1/n)$ and $o(1/n)$ (also $O(1/n^2)$ and $o(1/n^2)$) rates because of two reasons. First, they are of the same order in the worst-case nature; thus usually their difference is not that significant. Second, technically, for some basic operator splitting methods it is not hard to improve an $O(1/n)$ rate to $o(1/n)$ or from $O(1/n^2)$ to $o(1/n^2)$, see, e.g., [7].

with ADMM to yield an $O(1/n^2)$ convergence rate. That is, for the scheme

$$
\begin{cases}
x_{n+1} = \underset{x}{\operatorname{argmin}}\Big\{ f(x) + \dfrac{\sigma}{2}\big\|Ax + B\hat{y}_n - b - \dfrac{\hat{\lambda}_n}{\sigma}\big\|^2 \Big\} \\
y_{n+1} = \underset{y}{\operatorname{argmin}}\Big\{ g(y) + \dfrac{\sigma}{2}\big\|Ax_{n+1} + By - b - \dfrac{\hat{\lambda}_n}{\sigma}\big\|^2 \Big\} \\
\lambda_{n+1} = \hat{\lambda}_n - \sigma(Ax_{n+1} + By_{n+1} - b) \\
\alpha_{n+1} = (1 + \sqrt{1 + 4\alpha_n^2})/2 \\
\hat{y}_{n+1} = y_{n+1} + \dfrac{\alpha_n - 1}{\alpha_{n+1}}(y_{n+1} - y_n) \\
\hat{\lambda}_{n+1} = \lambda_{n+1} + \dfrac{\alpha_n - 1}{\alpha_{n+1}}(\lambda_{n+1} - \lambda_n)
\end{cases}
\tag{1.12}
$$

with $\{\alpha_n\}$ iteratively updated from $\alpha_0 = 1$, it has a worst-case $O(1/n^2)$ convergence rate in a nonergodic sense, where the accuracy of an iterate is measured by the error of the objective function of the dual problem of (1.11).

In this paper, we will establish a worst-case $O(1/n^2)$ convergence rate in the ergodic sense for both the original ADMM scheme (1.3) and the proximal version (1.4), under the condition that the penalty parameter $\sigma$ is iteratively adjusted by a specific rule similar as the one in [4, 5]. The restriction of the penalty parameter is mild and can be automatically determined with a given initial value (see (2.5)). We also show that the proximity of the Lagrange multiplier $\{\lambda_n\}$ to the optimal value is reduced on an $O(1/n^2)$ rate. Our assumptions on the model (1.1) are given at the beginning of Section 2. Note that we do not assume any strong convexity on the objective function of the model (1.1) as existing work such as [7, 9, 19, 36].

Finally, we would mention that some convergence rate results in the asymptotical sense can be established for ADMM if further assumptions are assumed. For example, the asymptotical linear convergence rate of ADMM was established in [2, 20] for the special case of (1.2) where both functions are quadratic. But this type of analysis is not the focus of this paper.

The rest of the paper is organized as follows. In Section 2, a faster ADMM with a worst-case $O(1/n^2)$ convergence rate is proposed and some remarks are given. We first prove the convergence for the faster ADMM in Section 3 and then establish its worst-case $O(1/n^2)$ convergence rate in Section 4. In Section 5, we elaborate on the connection between the faster ADMM and the primal-dual algorithm in [4]. In Section 6, we test the LASSO model and report some preliminary numerical results; some conclusions are also drawn based on these numerical results.

## 2 Faster ADMM with a Worst-case $O(1/n^2)$ Convergence Rate

As mentioned, we consider the original ADMM (1.3) and its proximal version (1.4) simultaneously for (1.2). So we relax the restriction of $Q$ in (1.4) and only require it to be positive semi-definite in our discussions. To derive a worst-case $O(1/n^2)$ convergence rate for the ADMM, our assumptions on the model (1.1) are summarized as follows.

**Assumption**: Both $f(x)$ and $g(x)$ are proper, lower semicontinuous (l.s.c.) and convex functions; $g(x)$ is smooth and $\nabla g$ is Lipschitz continuous with constant $\frac{1}{\gamma}$; and $A$ is full column rank.

We propose the faster ADMM with a worst-case $O(1/n^2)$ convergence rate in Algorithm 1. First, we notice that with the above assumption, it follows from [1, Theorem 18.15] that the following inequality holds:

$$g(\mu) \geq g(\nu) + \langle \nabla g(\nu), \mu - \nu \rangle + \frac{\gamma}{2}\|\nabla g(\mu) - \nabla g(\nu)\|^2, \quad \forall\, \mu, \nu \in \mathbb{R}^m. \tag{2.1}$$

---

**Algorithm 1:** Faster ADMM with a worst-case $O(1/n^2)$ convergence rate

---

Specify an integer $\kappa > 0$ as the frequency of adjusting the penalty parameter $\sigma_n$; an initial value of $\sigma_0 > 0$; and $(x_0, y_0, \lambda_0) \in \mathcal{X} \times \mathbb{R}^m \times \mathbb{R}^m$. Let $\frac{1}{\gamma}$ be the Lipschitz continuity constant of $\nabla g$. Choose a positive semi-definite matrix $Q \in \mathbb{R}^{d\times d}$. For the $(n+1)$-th iteration, perform the following steps:

$$\begin{cases} x_{n+1} = \underset{x\in\mathcal{X}}{\operatorname{argmin}}\Big\{ f(x) + \dfrac{\sigma_n}{2}\Big\|Ax - y_n + \dfrac{\lambda_n}{\sigma_n}\Big\|^2 + \dfrac{\sigma_n}{2}\|x - x_n\|_Q^2 \Big\} \\ y_{n+1} = \underset{y\in\mathbb{R}^m}{\operatorname{argmin}}\Big\{ g(y) + \dfrac{\sigma_n}{2}\Big\|Ax_{n+1} - y + \dfrac{\lambda_n}{\sigma_n}\Big\|^2 \Big\} \\ \lambda_{n+1} = \lambda_n + \sigma_n(Ax_{n+1} - y_{n+1}) \end{cases} \tag{2.2}$$

where the penalty parameter $\sigma_n$ is updated every $\kappa$ iterations by the rule

$$\sigma_n = \tilde{\sigma}_{\lfloor \frac{n}{\kappa} \rfloor}, \tag{2.3}$$

where $\lfloor \frac{n}{\kappa} \rfloor$ is the largest integer no greater than $\frac{n}{\kappa}$ and the sequence $\{\tilde{\sigma}_i\}$ is given by

$$\tilde{\sigma}_{i+1} = \frac{\tilde{\sigma}_i}{\sqrt{1 + \gamma\tilde{\sigma}_i}}, \text{ with } \tilde{\sigma}_0 = \sigma_0 > 0. \tag{2.4}$$

---

**Remark 2.1.** *Note that the sequence $\{\tilde{\sigma}_i\}$ is specified with a given $\sigma_0$ and the rule (2.3) adjusts the penalty parameter $\{\sigma_n\}$ after every $\kappa$ iterations by assigning each $\tilde{\sigma}_i$ to $\kappa$ iterations of the faster ADMM (2.2) consecutively. Thus, the sequence $\{\sigma_n\}$ is also automatically determined with a given initial value $\sigma_0$ and a frequency $\kappa$. For the extreme case where $\kappa = 1$, then we have*

$$\kappa = 1, \quad \begin{Bmatrix} \sigma_0 & \sigma_1 & \sigma_2 & \cdots & \sigma_n & \cdots \\ \downarrow & \downarrow & \downarrow & & \downarrow & \\ \tilde{\sigma}_0 & \tilde{\sigma}_1 & \tilde{\sigma}_2 & \cdots & \tilde{\sigma}_n & \cdots \end{Bmatrix},$$

*which means the sequence $\{\sigma_n\}$ is iteratively updated by*

$$\sigma_{n+1} = \frac{\sigma_n}{\sqrt{1 + \gamma\sigma_n}}. \tag{2.5}$$

*This is precisely the formula for updating the parameters of the accelerated primal-dual scheme in [4, 5]. If we choose $\kappa > 1$, e.g., $\kappa = 10$, then we have*

$$\kappa = 10, \quad \underbrace{\sigma_0 \;\; \sigma_1 \;\; \cdots \;\; \sigma_9}_{\tilde{\sigma}_0} \;\; \cdots \;\; \underbrace{\sigma_{10} \;\; \sigma_{11} \;\; \cdots \;\; \sigma_{19}}_{\tilde{\sigma}_1} \;\; \cdots;$$

*and for the general $\kappa$, we have*

$$\underbrace{\sigma_0 \ \cdots \ \sigma_{\kappa-1}}_{\tilde{\sigma}_0} \ \cdots \ \underbrace{\sigma_{\kappa} \ \cdots \ \sigma_{2\kappa-1}}_{\tilde{\sigma}_1} \ \cdots \ \underbrace{\sigma_{s\kappa} \ \cdots \ \sigma_{(s+1)\kappa-1}}_{\tilde{\sigma}_s} \ \cdots,$$

*For an integer $n$, it can be decomposed as $n = s\kappa + j$ with $0 \le j \le \kappa - 1$. Thus, it follows*

$$\sigma_{n+1} = \begin{cases} \sigma_n, & 0 \le j \le \kappa - 2, \\ \dfrac{\sigma_n}{\sqrt{1+\gamma\sigma_n}}, & j = \kappa - 1. \end{cases} \tag{2.6}$$

*Clearly, the sequence $\{\sigma_n\}$ is monotonically non-increasing. Thus, it is easy to understand that if the sequence $\{\sigma_n\}$ is updated on a too high frequency, i.e., $\kappa$ is small, then the sequence $\{\sigma_n\}$ decreases too fast and the step size for updating the dual variable becomes too small. In this case, the efficiency of the scheme (2.2) may be deteriorated. On the other hand, if the sequence $\{\sigma_n\}$ is updated on a too low frequency, i.e., $\kappa$ is huge, as we shall show in Theorems 4.1 and 4.2 (see (4.18) and (4.21)), the coefficient of the $O(1/n^2)$ convergence rate to be established is too large and it deteriorates the convergence also. So, in general we do not recommend too extreme values of $\kappa$ for the proposed faster ADMM (2.2). As we shall numerically verify later, medium values such as $\kappa = 5$ or 10, usually can result in very good numerical results even though the "optimal" choice, we believe, still depends on the specific application of the abstract model (1.2) and the data set under consideration.*

## 3 Convergence

Recall our main goal is to establish a worst-case $O(1/n^2)$ convergence rate for Algorithm 1. First of all, in this section we prove the convergence of Algorithm 1.

Let the Lagrangian function of (1.2) be defined as

$$L(x, y; \lambda) := f(x) + g(y) + (\lambda, Ax - y) \tag{3.1}$$

with $\lambda \in \mathbb{R}^m$ the Lagrange multiplier. Further, we define $\Omega := \mathcal{X} \times \mathbb{R}^m \times \mathbb{R}^m$. Then, solving (1.2) is equivalent to finding a saddle point of $L(x, y; \lambda)$. This is equivalent to solving the variational inequality: finding $w^* = (x^*, y^*, \lambda^*) \in \Omega$ such that

$$\Theta(v) - \Theta(v^*) + \big(F(w^*), w - w^*\big) \ge 0, \ \ \forall\, w \in \Omega, \tag{3.2}$$

where

$$v = \begin{pmatrix} x \\ y \end{pmatrix}, \ w = \begin{pmatrix} x \\ y \\ \lambda \end{pmatrix}, \ F(w) = \begin{pmatrix} A^T\lambda \\ -\lambda \\ -Ax + y \end{pmatrix}, \ \Theta(v) = f(x) + g(y). \tag{3.3}$$

To prove the convergence of the sequence $\{w_n\}$ generated by Algorithm 1, we first give a lemma.

**Lemma 3.1.** *Let the sequence $\{w_n = (x_n, y_n, \lambda_n)\}$ be generated by Algorithm 1. Then we have*

$$\begin{aligned} &\Theta(v) - \Theta(v_{n+1}) + \Big(F(w_{n+1}) + \Phi(y_n, y_{n+1}) + \sigma_n M_n(w_{n+1} - w_n), w - w_{n+1}\Big) \\ &\ge \frac{\gamma}{2}\|\nabla g(y) - \nabla g(y_{n+1})\|^2, \ \forall\, w \in \Omega, \end{aligned} \tag{3.4}$$

*where*

$$\Phi(y_n, y_{n+1}) = \begin{pmatrix} \sigma_n A^T(y_{n+1} - y_n) \\ -\sigma_n(y_{n+1} - y_n) \\ 0 \end{pmatrix} \text{ and } M_n = \begin{pmatrix} Q & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & \frac{1}{\sigma_n^2} I \end{pmatrix}. \tag{3.5}$$

*Proof.* First, the optimality condition of the $x$-subproblem in (2.2) is

$$f(x) - f(x_{n+1}) + \left( A^T\left(\sigma_n(Ax_{n+1} - y_n) + \lambda_n\right) + \sigma_n Q(x_{n+1} - x_n), x - x_{n+1} \right) \geq 0, \ \forall\, x \in \mathcal{X}. \tag{3.6}$$

Using the updating scheme for $\lambda_{n+1}$ in (2.2), we obtain

$$f(x) - f(x_{n+1}) + \left( A^T\lambda_{n+1} + \sigma_n A^T(y_{n+1} - y_n) + \sigma_n Q(x_{n+1} - x_n), x - x_{n+1} \right) \geq 0, \ \forall\, x \in \mathcal{X}. \tag{3.7}$$

In addition, it follows from (2.1) and the updating scheme for $\lambda_{n+1}$ in (2.2) that

$$g(y) - g(y_{n+1}) - \left(\lambda_{n+1}, y - y_{n+1}\right) \geq \frac{\gamma}{2}\|\nabla g(y) - \nabla g(y_{n+1})\|^2, \ \forall\, y \in \mathbb{R}^m. \tag{3.8}$$

Together with (3.7), (3.8) and the following identity

$$-Ax_{n+1} + y_{n+1} + \frac{1}{\sigma_n}(\lambda_{n+1} - \lambda_n) = 0, \tag{3.9}$$

we obtain the result (3.4). □

**Theorem 3.1.** *Let $w^* = (x^*, y^*, \lambda^*)$ be a saddle point of (3.1) and the sequence $\{w_n = (x_n, y_n, \lambda_n)\}$ be generated by Algorithm 1. Then we have*

$$\|w_{n+1} - w^*\|_{M_{n+1}}^2 \leq \|w_n - w^*\|_{M_n}^2 - \|w_n - w_{n+1}\|_{M_n}^2, \tag{3.10}$$

*where $M_n$ is given in (3.5).*

*Proof.* From the $y$-subproblem in (2.2), it holds

$$\lambda_{n+1} = \nabla g(y_{n+1}), \ \ \lambda^* = \nabla g(y^*).$$

Then, it follows from (3.4) with $w = w^*$ that

$$\begin{aligned} \sigma_n\left(M_n(w_{n+1} - w_n), w^* - w_{n+1}\right) \geq\ & \Theta(v_{n+1}) - \Theta(v^*) + \left(F(w_{n+1}), w_{n+1} - w^*\right) \\ & + \left(\Phi(y_n, y_{n+1}), w_{n+1} - w^*\right) + \frac{\gamma}{2}\|\lambda^* - \lambda_{n+1}\|^2. \end{aligned} \tag{3.11}$$

Taking $w = w_{n+1}$ in (3.2) and adding $(F(w_{n+1}), w_{n+1} - w^*)$ to both sides, we have

$$\Theta(v_{n+1}) - \Theta(v^*) + \left(F(w_{n+1}), w_{n+1} - w^*\right) \geq 0. \tag{3.12}$$

The optimality condition of $y$-subproblem in (2.2) is

$$g(y) - g(y_{n+1}) - (\lambda_{n+1}, y - y_{n+1}) \geq 0, \ \forall\, y \in \mathbb{R}^m, \tag{3.13}$$

and it also satisfies

$$g(y) - g(y_n) - (\lambda_n, y - y_n) \geq 0, \ \forall\, y \in \mathbb{R}^m. \tag{3.14}$$

Taking $y = y_n$ in (3.13) and $y = y_{n+1}$ in (3.14), and summarizing them, we obtain

$$\big(\Phi(y_n, y_{n+1}), w_{n+1} - w^*\big) = (\lambda_{n+1} - \lambda_n, y_{n+1} - y_n) \geq 0. \tag{3.15}$$

Therefore, it follows from (3.11), (3.12) and (3.15) that

$$\big(M_n(w_{n+1} - w_n), w^* - w_{n+1}\big) \geq \frac{\gamma}{2\sigma_n}\|\lambda^* - \lambda_{n+1}\|^2. \tag{3.16}$$

Using the identity

$$\|w_{n+1} - w^*\|_{M_n}^2 + 2\big(M_n(w_{n+1} - w_n), w^* - w_{n+1}\big) = \|w_n - w^*\|_{M_n}^2 - \|w_n - w_{n+1}\|_{M_n}^2,$$

and (3.16), we have

$$\|w_{n+1} - w^*\|_{M_n}^2 + \frac{\gamma}{\sigma_n}\|\lambda^* - \lambda_{n+1}\|^2 \leq \|w_n - w^*\|_{M_n}^2 - \|w_n - w_{n+1}\|_{M_n}^2. \tag{3.17}$$

Moreover, according to (2.6), it holds

$$\frac{1 + \gamma\sigma_n}{\sigma_n^2} \geq \frac{1}{\sigma_{n+1}^2}. \tag{3.18}$$

Thus, it follows from (3.17), (3.18) and the definition of $M_n$ in (3.5) that the assertion (3.10) holds. □

The assertion (3.10) implies that the sequence $\{w_n\}$ generated by Algorithm 1 is strictly contractive with respect to the solution set of (3.1), which essentially implies the convergence of the sequence $\{w_n\}$. We prove a lemma and then present the convergence result.

Recall the special case of the rule (2.3)-(2.4) with $\kappa = 1$, i.e., the sequence $\{\sigma_n\}$ is updated by (2.5). Then, as proved in [4], we have $\lim_{n\to\infty} \frac{\gamma}{2} n\sigma_n = 1$, which means $\sigma_n \sim O(1/n)$. Now, we generalize this result to the general rule (2.3)-(2.4) with a general frequency $\kappa$.

**Lemma 3.2.** *For $\{\sigma_n\}$ updated by the rule (2.3)-(2.4) in Algorithm 1, we have $\sigma_n \sim O(\kappa/n)$.*

*Proof.* For $\{\sigma_n\}$ updated by the rule (2.3)-(2.4), we have $\lim_{s\to\infty} \frac{\gamma}{2} s\tilde{\sigma}_s = 1$. For an integer $n$, it can be written as $n = s\kappa + j$, $0 \leq j \leq \kappa - 1$, where $s = \lfloor \frac{n}{\kappa} \rfloor$. Thus, we have $\lim_{n\to\infty} \frac{\gamma}{2} n\sigma_n = \kappa$, because $\kappa$ is a fixed integer. This yields $\sigma_n = \tilde{\sigma}_{\lfloor \frac{n}{\kappa} \rfloor} \sim O(\kappa/n)$ and completes the proof. □

**Theorem 3.2.** *Let $\{w_n = (x_n, y_n, \lambda_n)\}$ be the sequence generated by Algorithm 1. Then, the sequence $\{w_n\}$ converges to a saddle point $w^* = (x^*, y^*, \lambda^*)$ of (3.1). The convergence of Algorithm 1 for model (1.2) is thus established in senses of*

$$\lim_{n\to\infty}(Ax_n - y_n) = 0, \quad \textit{and} \quad \lim_{n\to\infty}\big\{f(x_n) + g(y_n)\big\} = f(x^*) + g(y^*). \tag{3.19}$$

*Proof.* Taking the summation of (3.10) for $n$ from 0 to $N$, we have

$$\sum_{n=0}^{N} \|w_n - w_{n+1}\|_{M_n}^2 \leq \|w_0 - w^*\|_{M_0}^2,$$

which indicates

$$\lim_{n\to\infty} \|w_n - w_{n+1}\|_{M_n}^2 = 0. \tag{3.20}$$

By the definition of $M_n$ given in (3.5), we have

$$\lim_{n\to\infty} Q(x_n - x_{n+1}) = 0, \ \lim_{n\to\infty} (y_n - y_{n+1}) = 0 \ \text{ and } \ \lim_{n\to\infty} \frac{1}{\sigma_n}(\lambda_n - \lambda_{n+1}) = 0. \tag{3.21}$$

It follows from (3.10) that $\|w_n - w^*\|_{M_n}^2$ is bounded. Recall the identity (3.9), we know that $\|Ax_n - Ax^*\|^2$ is bounded. Since $A$ is full column rank and $\sigma_n \sim O(\kappa/n)$ shown in Lemma 3.2, we conclude that the sequence $\{w_n\}$ has a cluster point. Let us denote it by $w^*$. Then, substituting it into (3.4) and using (3.21), we obtain

$$\Theta(v) - \Theta(v^*) + \big(F(w^*), w - w^*\big) \geq 0, \ \forall\, w \in \Omega,$$

which implies that $w^*$ is a saddle point of (3.1). Furthermore, by (3.9) and (3.21), it immediately yields

$$\lim_{n\to\infty} (Ax_n - y_n) = \lim_{n\to\infty} \frac{1}{\sigma_{n-1}}(\lambda_n - \lambda_{n-1}) = 0. \tag{3.22}$$

Taking $w = (x_n, y_n, \lambda_n)$ in (3.2), we get

$$f(x_n) + g(y_n) \geq f(x^*) + g(y^*) - (\lambda^*, Ax_n - y_n),$$

and thus

$$\liminf_{n\to\infty} \big\{f(x_n) + g(y_n)\big\} \geq f(x^*) + g(y^*). \tag{3.23}$$

In addition, we set $w = w^*$ in (3.4) and simplify it as

$$\begin{aligned} f(x^*) + g(y^*) \geq\ & f(x_{n+1}) + g(y_{n+1}) + (\lambda^*, Ax_{n+1} - y_{n+1}) \\ & + \sigma_n(y_{n+1} - y_n, Ax_{n+1} - y_{n+1}) \\ & + \big(\sigma_n M_n(w_{n+1} - w_n), w_{n+1} - w^*\big) \end{aligned}$$

Thus, using the boundedness of $\|w_n - w^*\|_{M_n}^2$, the results in (3.20), (3.21) and (3.22), and $\sigma_n \sim O(\kappa/n)$ shown in Lemma 3.2, we obtain

$$f(x^*) + g(y^*) \geq \limsup_{n\to\infty} \big\{f(x_n) + g(y_n)\big\}. \tag{3.24}$$

Therefore, (3.23) and (3.24) imply the assertion (3.19) and the proof is complete. □

## 4 Worst-case $O(1/n^2)$ Convergence Rate

In this section, we establish a worst-case $O(1/n^2)$ convergence rate for Algorithm 1. Our analysis is based on the saddle-point reformulation of the model (1.1)

$$\min_{x\in\mathcal{X}} \max_{\lambda\in\mathbb{R}^m} \big\{\mathcal{L}(x,\lambda) := f(x) + (Ax, \lambda) - g^*(\lambda)\big\}. \tag{4.1}$$

As analyzed in [4], the following partial primal-dual gap can be used to measure the accuracy of an iterate generated by Algorithm 1:

$$\mathcal{G}_{\mathcal{B}_1\times\mathcal{B}_2}(x,\lambda) = \max_{\lambda'\in\mathcal{B}_2} \mathcal{L}(x,\lambda') - \min_{x'\in\mathcal{B}_1} \mathcal{L}(x',\lambda), \tag{4.2}$$

where $\mathcal{B}_1 \times \mathcal{B}_2$ is a open subset of $U := \mathcal{X} \times \mathbb{R}^m$ containing a solution point $(x^*, \lambda^*)$ of the saddle-point reformulation (4.1). According to [4], we know that for $(\hat{x}, \hat{\lambda})$ in $\mathcal{B}_1 \times \mathcal{B}_2$, if $\mathcal{G}_{\mathcal{B}_1 \times \mathcal{B}_2}(\hat{x}, \hat{\lambda}) \leq 0$, then $(\hat{x}, \hat{\lambda})$ is also a solution point of (4.1). Hence, we can define $(\tilde{x}, \tilde{\lambda}) \in \mathcal{B}_1 \times \mathcal{B}_2$ as an approximate solution to (4.1) with an accuracy of $\epsilon$ if

$$\mathcal{G}_{\mathcal{B}_1 \times \mathcal{B}_2}(\tilde{x}, \tilde{\lambda}) \leq \epsilon \tag{4.3}$$

with $\epsilon > 0$. We additionally need the following notation for further analysis:

$$u = \begin{pmatrix} x \\ \lambda \end{pmatrix}, \quad H_n = \begin{pmatrix} \sigma_n(A^T A + Q) & 0 \\ 0 & \frac{1}{\sigma_n} I \end{pmatrix}. \tag{4.4}$$

Now we start to establish a worst-case $O(1/n^2)$ convergence rate for Algorithm 1. First, based on the first-order optimality conditions of the subproblems in (2.2), we prove a lemma.

**Lemma 4.1.** *Let $\{(x_n, y_n, \lambda_n)\}$ be generated by Algorithm 1. Then, we have*

$$\begin{aligned} f(x) - f(x_{n+1}) + \Big( A^T \big( \sigma_n A(x_{n+1} - x_n) + \frac{\sigma_n}{\sigma_{n-1}} (\lambda_n - \lambda_{n-1}) + \lambda_n \big) \\ + \sigma_n Q(x_{n+1} - x_n), x - x_{n+1} \Big) \geq 0, \ \ \forall\, x \in \mathcal{X}; \end{aligned} \tag{4.5}$$

$$\lambda_{n+1} = \nabla g(y_{n+1}); \tag{4.6}$$

$$\begin{aligned} g^*(\lambda) - g^*(\lambda_{n+1}) - \big( A x_{n+1} - \frac{1}{\sigma_n} (\lambda_{n+1} - \lambda_n), \lambda - \lambda_{n+1} \big) \\ - \frac{\gamma}{2} \|\lambda - \lambda_{n+1}\|^2 \geq 0, \ \ \forall\, \lambda \in \mathbb{R}^m. \end{aligned} \tag{4.7}$$

*Proof.* From the optimality conditions of the subproblems in (2.2), we have the first two assertions trivially. Furthermore, because of (4.6), we have

$$y_{n+1} \in \partial g^*(\lambda_{n+1}). \tag{4.8}$$

Since $\nabla g$ is Lipschitz continuous with constant $\frac{1}{\gamma}$, it follows from [1, Theorem 18.15] that $g^*$ is $\gamma$-strongly convex. Then, together with (4.8), we obtain

$$g^*(\lambda) - g^*(\lambda_{n+1}) - \big( y_{n+1}, \lambda - \lambda_{n+1} \big) - \frac{\gamma}{2} \|\lambda - \lambda_{n+1}\|^2 \geq 0, \ \ \forall\, \lambda \in \mathbb{R}^m. \tag{4.9}$$

Thus, using the updating scheme for $\lambda$ in (2.2), we prove the assertion (4.7). □

Then, we prove one more lemma, based on which the worst-case $O(1/n^2)$ convergence rate of Algorithm 1 can be easily obtained.

**Lemma 4.2.** *Let $\{u_n = (x_n, \lambda_n)\}$ be generated by Algorithm 1 and $U = \mathcal{X} \times \mathbb{R}^m$. Then, we have*

$$\big( \mathcal{L}(x_{n+1}, \lambda) - \mathcal{L}(x, \lambda_{n+1}) \big) + \frac{1}{\theta_{n+1}} S_{n+1}(u) \leq S_n(u), \ \ \forall\, u \in U, \tag{4.10}$$

*where*

$$\theta_n := \frac{\sigma_n}{\sigma_{n-1}}$$

*and*

$$S_n(u) := \frac{1}{2} \|u - u_n\|_{H_n}^2 + \frac{\theta_n^2}{2\sigma_n} \|\lambda_n - \lambda_{n-1}\|^2 + \theta_n \big( A(x - x_n), (\lambda_n - \lambda_{n-1}) \big). \tag{4.11}$$

*Proof.* First, it follows from (4.5) and (4.7) that

$$\begin{aligned}
&\Big(f(x)+(Ax,\lambda_{n+1})-g^*(\lambda_{n+1})\Big)-\Big(f(x_{n+1})+(Ax_{n+1},\lambda)-g^*(\lambda)\Big)\\
&+\sigma_n\big((A^TA+Q)(x_{n+1}-x_n),x-x_{n+1}\big)+\frac{1}{\sigma_n}(\lambda_{n+1}-\lambda_n,\lambda-\lambda_{n+1})\\
&\geq\big(A(x-x_{n+1}),\lambda_{n+1}-\lambda_n\big)-\frac{\sigma_n}{\sigma_{n-1}}\big(A(x-x_{n+1}),\lambda_n-\lambda_{n-1}\big)\\
&+\frac{\gamma}{2}\|\lambda-\lambda_{n+1}\|^2,\ \ \forall\, u\in U.
\end{aligned}\tag{4.12}$$

With (4.1), the definition of $H_n$ in (4.4) and the equality

$$\begin{aligned}
&\big(A(x-x_{n+1}),\lambda_n-\lambda_{n-1}\big)\\
&=\big(A(x-x_n),\lambda_n-\lambda_{n-1}\big)-\big(A(x_{n+1}-x_n),\lambda_n-\lambda_{n-1}\big),
\end{aligned}\tag{4.13}$$

we can reformulate (4.12) as

$$\begin{aligned}
&-\big(\mathcal{L}(x_{n+1},\lambda)-\mathcal{L}(x,\lambda_{n+1})\big)+\big(H_n(u_{n+1}-u_n),u-u_{n+1}\big)\\
\geq\ &\big(A(x-x_{n+1}),(\lambda_{n+1}-\lambda_n)\big)-\frac{\sigma_n}{\sigma_{n-1}}\big(A(x-x_n),(\lambda_n-\lambda_{n-1})\big)\\
&+\frac{\sigma_n}{\sigma_{n-1}}\big(A(x_{n+1}-x_n),(\lambda_n-\lambda_{n-1})\big)+\frac{\gamma}{2}\|\lambda-\lambda_{n+1}\|^2\\
\geq\ &\big(A(x-x_{n+1}),(\lambda_{n+1}-\lambda_n)\big)-\frac{\sigma_n}{\sigma_{n-1}}\big(A(x-x_n),(\lambda_n-\lambda_{n-1})\big)\\
&-\frac{\sigma_n}{2}\|A(x_{n+1}-x_n)\|^2-\frac{\sigma_n^2}{2\sigma_{n-1}^2\sigma_n}\|\lambda_n-\lambda_{n-1}\|^2+\frac{\gamma}{2}\|\lambda-\lambda_{n+1}\|^2,\ \ \forall\, u\in U.
\end{aligned}\tag{4.14}$$

Further, using the identity

$$\big(H_n(u_{n+1}-u_n),u-u_{n+1}\big)=\frac{1}{2}\big(\|u-u_n\|_{H_n}^2-\|u-u_{n+1}\|_{H_n}^2-\|u_n-u_{n+1}\|_{H_n}^2\big),$$

we have

$$\begin{aligned}
&\frac{1}{2}\|u-u_n\|_{H_n}^2+\frac{\sigma_n^2}{2\sigma_{n-1}^2\sigma_n}\|\lambda_n-\lambda_{n-1}\|^2+\frac{\sigma_n}{\sigma_{n-1}}\big(A(x-x_n),(\lambda_n-\lambda_{n-1})\big)\\
\geq\ &\frac{\sigma_n}{2}\|A(x-x_{n+1})\|^2+\frac{\sigma_n}{2}\|x-x_{n+1}\|_Q^2+\frac{1+\gamma\sigma_n}{2\sigma_n}\|\lambda-\lambda_{n+1}\|^2\\
&+\frac{\sigma_n}{2}\|x_{n+1}-x_n\|_Q^2+\frac{1}{2\sigma_n}\|\lambda_{n+1}-\lambda_n\|^2+\big(A(x-x_{n+1}),(\lambda_{n+1}-\lambda_n)\big)\\
&+\big(\mathcal{L}(x_{n+1},\lambda)-\mathcal{L}(x,\lambda_{n+1})\big),\ \ \forall\, u\in U.
\end{aligned}\tag{4.15}$$

Recall (2.3), (2.4) and (2.6). We have

$$\sigma_n\theta_{n+1}=\sigma_{n+1},\quad \frac{1+\gamma\sigma_n}{\sigma_n}\geq\frac{1}{\theta_{n+1}\sigma_{n+1}}.\tag{4.16}$$

Then, it easily yields from (4.15) that

$$S_n(u)\geq\frac{1}{\theta_{n+1}}S_{n+1}(u)+\big(\mathcal{L}(x_{n+1},\lambda)-\mathcal{L}(x,\lambda_{n+1})\big),\ \ \forall\, u\in U.\tag{4.17}$$

The proof is complete. □

Now, we establish a worst-case $O(1/n^2)$ convergence rate in the ergodic sense for Algorithm 1.

**Theorem 4.1** ($O(1/n^2)$ convergence rate in the ergodic sense)**.** *Let $\{u_n = (x_n, \lambda_n)\}$ be the sequence generated by Algorithm 1. Let*

$$T_n = \sum_{i=0}^{n} \frac{\sigma_0}{\sigma_i} \quad \text{and} \quad \tilde{u}_n = \frac{1}{T_n} \sum_{i=0}^{n} \frac{\sigma_0}{\sigma_i} u_{i+1}.$$

*Then, we have*

$$\mathcal{G}_{\mathcal{B}_1 \times \mathcal{B}_2}(\tilde{x}_n, \tilde{\lambda}_n) \leq c\frac{\kappa}{n^2} = O(\frac{1}{n^2}), \tag{4.18}$$

*where $c > 0$ is a constant.*

*Proof.* Multiplying the inequality (4.10) in Lemma 4.2 by $\frac{\sigma_0}{\sigma_i}$, summarizing it with $i = 0, 1, \cdots, n$, and using the property of convex functions $f$ and $g$, we obtain

$$T_n\big(\mathcal{L}(\tilde{x}_n, \lambda) - \mathcal{L}(x, \tilde{\lambda}_n)\big) + \frac{\sigma_0}{\sigma_{n+1}} S_{n+1}(u) \leq \frac{1}{2}\|u - u_0\|_{H_0}^2, \tag{4.19}$$

where $\lambda_{-1} := \lambda_0$ is set. It follows from Lemma 3.2 that $\sigma_n \sim O(\kappa/n)$. With the definition of $T_n$, this immediately yields that $T_n \sim O(n^2/\kappa)$. Since $S_{n+1}(u)$ is nonnegative, we have

$$\mathcal{L}(\tilde{x}_n, \lambda) - \mathcal{L}(x, \tilde{\lambda}_n) \leq \frac{1}{2T_n}\|u - u_0\|_{H_0}^2 \leq c\frac{\kappa}{n^2}\|u - u_0\|_{H_0}^2, \ \ \forall\, u \in U. \tag{4.20}$$

From the result of Theorem 3.1, we know that the sequence $\{u_n\}$ is bounded, then its linear average $\tilde{u}_n$ is also bounded. Therefore, for some open bounded subset $\mathcal{B}_1 \times \mathcal{B}_2$ of $U$ containing the sequence $\{\tilde{u}_n\}$, the estimate (4.20) implies the assertion (4.18). The proof is complete. □

The assertion (4.18) means that $\tilde{u}_n$ calculated by $n$ iterations of Algorithm 1 is an approximate solution of the saddle-point reformulation (4.1) with an accuracy of $O(1/n^2)$. Therefore, a worst-case $O(1/n^2)$ convergence rate in the ergodic sense is established for Algorithm 1. Moreover, we can obtain a stronger convergence rate for the sequence of $\{\lambda_n\}$ in terms of the proximity to the optimal value $\lambda^*$ in the following theorem. This is a by-product of this paper.

**Theorem 4.2** (Convergence rate of dual variable)**.** *Let $u^* = (x^*, \lambda^*)$ be a solution point of the saddle-point reformulation (4.1), and $\{u_n = (x_n, \lambda_n)\}$ be generated by Algorithm 1. Then, we have*

$$\|\lambda_{n+1} - \lambda^*\|^2 \leq \frac{\sigma_{n+1}^2}{\sigma_0}\|u^* - u_0\|_{H_0}^2 = O(\frac{\kappa^2}{n^2}). \tag{4.21}$$

*Proof.* Taking $u = u^*$ in (4.19), and recalling the fact

$$\mathcal{L}(\tilde{x}_n, \lambda^*) - \mathcal{L}(x^*, \tilde{\lambda}_n) \geq 0,$$

we have

$$\frac{\sigma_0}{\sigma_{n+1}} S_{n+1}(u^*) \leq \frac{1}{2}\|u^* - u_0\|_{H_0}^2. \tag{4.22}$$

From (4.4) and the definition of $S_n$ in (4.11), we can easily have

$$\frac{1}{2\sigma_{n+1}}\|\lambda^* - \lambda_{n+1}\|^2 \leq S_{n+1}(u^*). \tag{4.23}$$

Therefore, the result (4.21) follows from the above two inequalities and Lemma 3.2. □

## 5 Connection with Primal-Dual Algorithms

In this section, we elaborate on the connection between Algorithm 1 with the primal-dual algorithm proposed in [4] for (1.1) with $\mathcal{X} = \mathbb{R}^d$. Recall the relationship between the primal-dual algorithm (1.9) and the linearized version of ADMM (1.6) for (1.2) with $\mathcal{X} = \mathbb{R}^d$. Here, we consider a symmetric version of (1.9) with an exchange of the primal and dual roles of the variables:

$$\begin{cases} x_{n+1} = \mathrm{prox}_{\tau_n f}\big(x_n - \tau_n A^T \tilde{\lambda}_n\big) \\ \lambda_{n+1} = \mathrm{prox}_{\sigma_n g^*}\big(\lambda_n + \sigma_n A x_{n+1}\big) \\ \tilde{\lambda}_{n+1} = \lambda_{n+1} + \theta_{n+1}(\lambda_{n+1} - \lambda_n). \end{cases} \tag{5.1}$$

First, using the Moreau's identity in [34]:

$$\mathrm{prox}_{\sigma g^*}(\lambda) + \sigma \mathrm{prox}_{g/\sigma}(\lambda/\sigma) = \lambda,$$

we can reformulate the $\lambda$-subproblem in (5.1) as

$$\lambda_{n+1} = \lambda_n + \sigma_n(Ax_{n+1} - y_{n+1}), \tag{5.2}$$

where

$$y_{n+1} = \operatorname*{argmin}_y \Big\{ g(y) + \frac{\sigma_n}{2}\big\| y - Ax_{n+1} - \frac{\lambda_n}{\sigma_n}\big\|^2 \Big\}. \tag{5.3}$$

Second, the $x$-subproblem in (5.1) can be rewritten as

$$\begin{aligned} x_{n+1} &= \operatorname*{argmin}_x \left\{ \begin{array}{c} f(x) + \frac{\sigma_n}{2}\big\|(\frac{\theta_n\sigma_{n-1}}{\sigma_n} - 1)Ax_n + Ax - \frac{\theta_n\sigma_{n-1}}{\sigma_n} y_n + \frac{\lambda_n}{\sigma_n}\big\|^2 \\ + \frac{1}{2}\|x - x_n\|^2_{\tau_n^{-1} I - \sigma_n A^T A} + C_n \end{array} \right\} \\ &= \operatorname*{argmin}_x \left\{ \begin{array}{c} f(x) + \frac{\sigma_n}{2}\big\|Ax - y_n + (1-\vartheta_n)\frac{\lambda_n}{\sigma_n} + \vartheta_n \frac{\lambda_{n-1}}{\sigma_n}\big\|^2 \\ + \frac{1}{2}\|x - x_n\|^2_{\tau_n^{-1} I - \sigma_n A^T A} \end{array} \right\}, \end{aligned} \tag{5.4}$$

where

$$\vartheta_n := \frac{\sigma_n - \theta_n \sigma_{n-1}}{\sigma_{n-1}}$$

and

$$C_n := \frac{\tau_n}{2}\big\|A^T\big(\lambda_n + \theta_n(\lambda_n - \lambda_{n-1})\big)\big\|^2 - \frac{1}{2\sigma_n}\big\|\lambda_n + \theta_n(\lambda_n - \lambda_{n-1})\big\|^2.$$

Thus, the primal-dual algorithm (5.1) is equivalent to the following scheme

$$\begin{cases} x_{n+1} = \operatorname*{argmin}_x \left\{ \begin{array}{c} f(x) + \frac{\sigma_n}{2}\big\|Ax - y_n + (1-\vartheta_n)\frac{\lambda_n}{\sigma_n} + \vartheta_n \frac{\lambda_{n-1}}{\sigma_n}\big\|^2 \\ + \frac{1}{2}\|x - x_n\|^2_{\tau_n^{-1} I - \sigma_n A^T A} \end{array} \right\} \\ y_{n+1} = \operatorname*{argmin}_y \Big\{ g(y) + \frac{\sigma_n}{2}\big\| Ax_{n+1} - y + \frac{\lambda_n}{\sigma_n}\big\|^2 \Big\} \\ \lambda_{n+1} = \lambda_n + \sigma_n(Ax_{n+1} - y_{n+1}), \end{cases} \tag{5.5}$$

which can be viewed as a linearized version of the ADMM with varying penalty parameters

$$\begin{cases} x_{n+1} = \operatorname*{argmin}_x \Big\{ f(x) + \frac{\sigma_n}{2}\big\|Ax - y_n + (1-\vartheta_n)\frac{\lambda_n}{\sigma_n} + \vartheta_n \frac{\lambda_{n-1}}{\sigma_n}\big\|^2 \Big\} \\ y_{n+1} = \operatorname*{argmin}_y \Big\{ g(y) + \frac{\sigma_n}{2}\big\| Ax_{n+1} - y + \frac{\lambda_n}{\sigma_n}\big\|^2 \Big\} \\ \lambda_{n+1} = \lambda_n + \sigma_n(Ax_{n+1} - y_{n+1}) \end{cases} \tag{5.6}$$

whose $x$-subproblem is proximally regularized by the term

$$\frac{1}{2}\|x-x_n\|^2_{\tau_n^{-1}I-\sigma_n A^TA}.$$

Furthermore, as shown in Lemma 4.2, $\vartheta_n=0$ if $\theta_n=\sigma_n/\sigma_{n-1}$. Therefore, the scheme (5.6) can be simplified as

$$\text{(5.7)}\qquad \begin{cases} x_{n+1}=\operatorname{argmin}_x\Big\{f(x)+\dfrac{\sigma_n}{2}\Big\|Ax-y_n+\dfrac{\lambda_n}{\sigma_n}\Big\|^2\Big\}\\ y_{n+1}=\operatorname{argmin}_y\Big\{g(y)+\dfrac{\sigma_n}{2}\Big\|Ax_{n+1}-y+\dfrac{\lambda_n}{\sigma_n}\Big\|^2\Big\}\\ \lambda_{n+1}=\lambda_n+\sigma_n(Ax_{n+1}-y_{n+1}). \end{cases}$$

This is precisely the application of the standard ADMM scheme (1.3) with varying penalty parameters $\sigma_n$ to (1.2). Therefore, the conclusion is that if $\sigma_n=\theta_n\sigma_{n-1}$ is satisfied, then the primal-dual algorithm (5.1) is the case of Algorithm 1 with

$$Q=(\tau_n\sigma_n)^{-1}I-A^TA.$$

This connection can be regarded as a generalization of the elaboration in [35] on these two methods with constant parameters.

**Remark 5.1.** *Different from the ($x$-$\lambda$-$\lambda$) scheme in (5.1), the accelerated scheme (1.9) discussed in [4] performs iterations in order of ($\lambda$-$x$-$x$). With an analysis similar as above, we can derive that the scheme (1.9) is equivalent to*

$$\text{(5.8)}\qquad \begin{cases} \lambda_{n+1}=\operatorname{argmin}_\lambda\left\{\begin{array}{c} g^*(\lambda)+\frac{\tau_n}{2}\Big\|A^T\lambda+z_n-(1-\delta_n)\frac{x_n}{\tau_n}-\delta_n\frac{x_{n-1}}{\tau_n}\Big\|^2\\ +\frac{1}{2}\|\lambda-\lambda_n\|^2_{\sigma_n^{-1}I-\tau_nAA^T}\end{array}\right\}\\ z_{n+1}=\operatorname{argmin}_z\Big\{f^*(z)+\dfrac{\tau_n}{2}\Big\|A^T\lambda_{n+1}+z-\dfrac{x_n}{\tau_n}\Big\|^2\Big\}\\ x_{n+1}=x_n-\tau_n(Ax_{n+1}+z_{n+1}), \end{cases}$$

*where*

$$\delta_n=\frac{\tau_n-\theta_n\tau_{n-1}}{\tau_{n-1}}.$$

*If $\theta_n=\tau_n/\tau_{n-1}$, then (5.8) can be simplified as*

$$\text{(5.9)}\qquad \begin{cases} \lambda_{n+1}=\operatorname{argmin}_\lambda\Big\{g^*(\lambda)+\dfrac{\tau_n}{2}\Big\|A^T\lambda+z_n-\dfrac{x_n}{\tau_n}\Big\|^2+\dfrac{1}{2}\|\lambda-\lambda_n\|^2_{\sigma_n^{-1}I-\tau_nAA^T}\Big\}\\ z_{n+1}=\operatorname{argmin}_z\Big\{f^*(z)+\dfrac{\tau_n}{2}\Big\|A^T\lambda_{n+1}+z-\dfrac{x_n}{\tau_n}\Big\|^2\Big\}\\ x_{n+1}=x_n-\tau_n(Ax_{n+1}+z_{n+1}), \end{cases}$$

*which is an application of the proximal version of the ADMM with varying penalty parameters $\tau_n$ to the problem*

$$\text{(5.10)}\qquad \begin{aligned} \min_{\lambda\in\mathbb{R}^m,z\in\mathbb{R}^d}\ & f^*(z)+g^*(\lambda)\\ \text{s.t.}\ \ & A^T\lambda+z=0. \end{aligned}$$

*Note that the dual problem of* (1.2) *with* $\mathcal{X} = \mathbb{R}^d$ *can be written as*

$$\max_{\lambda \in \mathbb{R}^m} \ \big\{ -\big(f^*(-A^T\lambda) + g^*(\lambda)\big)\big\}. \tag{5.11}$$

*Thus, the problem* (5.10) *is equivalent to the dual problem* (5.11) *of* (1.2) *by introducing variable* $z = -A^T\lambda$ *and regarding* $x$ *in* (5.9) *as the Lagrange multiplier of the constraint in* (5.10).

## 6 Numerical Results

As mentioned, the ADMM has found many applications in a broad spectrum of areas. In this section, we take the LASSO model in [38] as an illustrative example to numerically verify the efficiency of the proposed faster ADMM. The LASSO model is probably the simplest yet most representative application to which ADMM can be applied. All the codes were written by MATLAB R2012a and all experiments were performed on a desktop with Windows 8 system and an Intel(R) Core(TM) i5-4570s CPU processor (2.9GHz) with a 8GB memory.

### 6.1 Experiment Setup

The LASSO model in [38] is

$$\min_x \ \Big\{\alpha\|x\|_1 + \frac{1}{2}\|Dx - c\|^2\Big\}, \tag{6.1}$$

where $\|x\|_1 := \sum_{i=1}^d |x_i|$, $D \in \mathbb{R}^{l\times d}$ is a design matrix usually with $l \ll d$, $l$ is the number of data points, $d$ is the number of features, $c \in \mathbb{R}^l$ is the response vector and $\alpha > 0$ is a regularization parameter. The LASSO model provides a sparse estimation of $x$ when there are more features than data points. It can also be explained as a model for finding a sparse solution of the under-determined system of linear equations $Dx = c$.

Obviously, model (6.1) can be rewritten as

$$\begin{aligned} \min_{x,y} \ & \alpha\|x\|_1 + \frac{1}{2}\|Dy - c\|^2 \\ \text{s.t.} \ & x - y = 0, \end{aligned} \tag{6.2}$$

which is a special case of model (1.2) with $f(x) = \alpha\|x\|_1$, $g(y) = \frac{1}{2}\|Dy - c\|^2$ and $m = d$, $\mathcal{X} = \mathbb{R}^d$, $A = I_{d\times d}$. For (6.2), it is easy to see that the assumptions posed in Section 2 are satisfied. Particulary, the Lipschitz continuity constant of $\nabla g$ is $\|D^TD\|$ and $\gamma = 1/\|D^TD\|$. Thus, applying the proposed faster ADMM (2.2) with $Q = 0$ to (6.2), we obtain the scheme

$$\begin{cases} x_{n+1} = \operatorname{argmin}_x \Big\{\alpha\|x\|_1 + \dfrac{\sigma_n}{2}\Big\|x - y_n + \dfrac{\lambda_n}{\sigma_n}\Big\|^2\Big\} \\ y_{n+1} = \operatorname{argmin}_y \Big\{\dfrac{1}{2}\|Dy - c\|^2 + \dfrac{\sigma_n}{2}\Big\|x_{n+1} - y + \dfrac{\lambda_n}{\sigma_n}\Big\|^2\Big\} \\ \lambda_{n+1} = \lambda_n + \sigma_n(x_{n+1} - y_{n+1}) \end{cases} \tag{6.3}$$

where the penalty parameter $\{\sigma_n\}$ is updated by

$$\sigma_{n+1} = \tilde{\sigma}_{\lfloor \frac{n}{\kappa} \rfloor}$$

with $\kappa$ being a given integer and

$$\tilde{\sigma}_{s+1} = \frac{\tilde{\sigma}_s}{\sqrt{1+\tilde{\sigma}_s/\|D^TD\|}},$$

starting from a given $\tilde{\sigma}_0 = \sigma_0$. As mentioned in many literatures, the $x$-subproblem in (6.3) has its closed-form solution given by

$$x_{n+1} = S_{\alpha/\sigma_n}\big(y_n - \frac{\lambda_n}{\sigma_n}\big),$$

where $S_\delta(x)$ is the soft-thresholding operator [10] defined as

$$(S_\delta(x))_i = (1-\delta/|x_i|)_+ \cdot x_i, \;\; i = 1, 2, \cdots, d,$$

and the $y$-subproblem has its solution given by

$$y_{n+1} = (\sigma_n I + D^TD)^{-1}(\sigma_n x_{n+1} + \lambda_n + D^Tc).$$

In our experiments, we specify the LASSO model as follows. We take $l = 1500$ and $d = 5000$; the matrix $D$ in (6.1) is generated by the MATLAB function `randn` with 1500 by 5000 entries; all columns are normalized afterwards; the sparse vector $x \in \mathbb{R}^{5000}$ is generated by the MATLAB function `sprandn` with 100 nonzero entries; the vector $c$ is set as $Dx + \eta$ with noise vector $\eta \sim N(0, 0.001)$; the regularization parameter $\alpha$ is set as $\|D^Tc\|_\infty/10$, $\gamma = 1/\|D^TD\|$. To implement the scheme (6.3) and avoid loss of efficiency possibly caused by coding skills, we use the widely-used MATLAB ADMM package downloaded at http://web.stanford.edu/~boyd/papers/admm/, with the only slight revision of using an adaptive penalty parameter.

For the iterate $w_{n+1} = (x_{n+1}, y_{n+1}, \lambda_{n+1})^T$ generated by (6.3), it is easy to see that it satisfies the variational inequality

(6.4)
$$\Theta(v) - \Theta(v_{n+1}) + \big(F(w_{n+1}), w - w_{n+1}\big) + \left(\begin{pmatrix}\sigma_n(y_{n+1}-y_n)\\ \frac{1}{\sigma_n}(\lambda_{n+1}-\lambda_n)\end{pmatrix}, \begin{pmatrix}x - x_{n+1}\\ \lambda - \lambda_{n+1}\end{pmatrix}\right) \geq 0,$$

for any $w \in \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}^d$, where

$$v = (x, y)^T, \;\; \Theta(v) = f(x) + g(y) \;\text{ and }\; F(w_{n+1}) = (\lambda_{n+1}, -\lambda_{n+1}, x_{n+1} - y_{n+1})^T.$$

Thus, $w_{n+1}$ is a solution of (6.2) if and only if $y_{n+1} = y_n$ and $\lambda_{n+1} = \lambda_n$, see similar analysis in [3, 24, 41]. Hence, we can use the stopping criterion

(6.5)
$$\max\Big\{\sigma_n\|y_{n+1} - y_n\|, \frac{1}{\sigma_n}\|\lambda_{n+1} - \lambda_n\|\Big\} \leq \sqrt{d}\varepsilon,$$

where $\varepsilon > 0$ is a tolerance.

### 6.2 Efficiency Comparison

We use the original ADMM (1.3) with a constant penalty parameter as the benchmark to verify the $O(1/n^2)$ convergence rate of Algorithm 1. We test different constant penalty parameters for the original ADMM (1.3) and Algorithm 1 also starts from the same constant for each comparison. In Figure 1, we plot evaluation of the objective function value of the LASSO model (6.1) with respect to the iteration number for the original ADMM (1.3)

with a constant penalty parameter $\sigma_0$ (denoted by "ADMM") and Algorithm 1 with varying penalty parameter starting from the same $\sigma_0$ (denoted by "FADMM"). We report the results for $\sigma_0 = 10, 20, \cdots$, up to 2000. We have tested a number of other cases of $\sigma_0$ and the comparison is similar except for some extreme cases where $\sigma_0$ is very small. For Algorithm 1, four choices of $\kappa = 1, 5, 10, 20$ are tested. For each case, we plot the evolution of primal-dual residual in (6.5) with log-log scale axis in Figure 1. These curves show clearly that Algorithm 1 converges absolutely faster than the original ADMM (1.3) with a given constant parameter. To discern the actual rate more clearly, we also set a benchmark rate of $100/n^2$ and plot its decay. As we can see, the speed of the decay of the primal-dual residuals is even faster than the benchmark rate of $100/n^2$; so the established $O(1/n^2)$ convergence rate is just a worse-case estimate of the speed and we can easily witness faster speed empirically. Moreover, the values of the objective function in each iteration are also plotted in Figure 1, where the decay of the objective function by Algorithm 1 is also much faster than that of the original ADMM, especially when the initial penalty parameter is large.

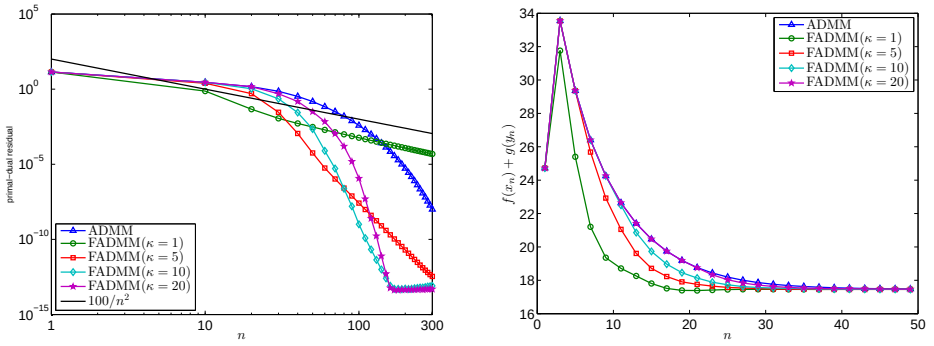
In Table 1, we compare the iteration numbers of these two ADMM schemes for some choices of $\sigma_0$ and different tolerance in the stopping criterion (6.5). In this table, "–" means the stopping criterion (6.5) is not satisfied after 5000 iterations. These data show that the original ADMM (1.3) with a given constant penalty parameter can easily fail, especially when pursuing high-precision solutions; while Algorithm 1 usually performs very well except for the extreme case where $\kappa = 1$. For most of the case, medium values of $\kappa$ such as $\kappa = 5$ or 10 are good choices for our experiments.

### 6.3 Sensitivity

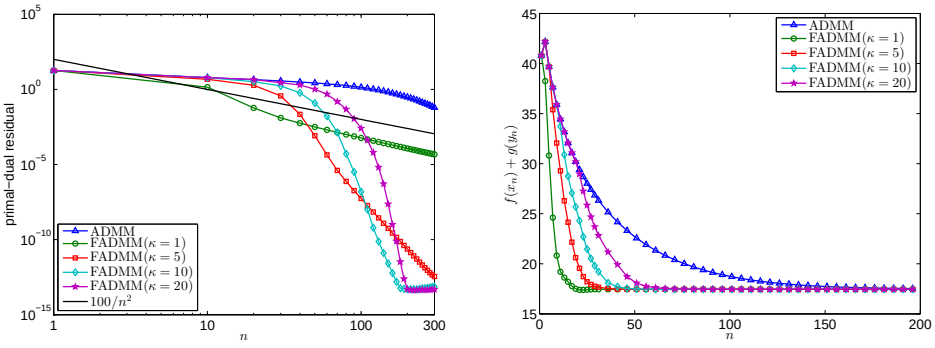
In the literature, it is well known that the efficiency of the original ADMM (1.3) with a constant penalty parameter heavily depends on the value of this parameter and it seems we still lack of any general strategy to tune this constant. This can also be seen in Table 1. Indeed, it is the main disadvantage of the ADMM (1.3) and it usually requires users to tune this parameter to find a specific value appropriate to a given problem. In this subsection, we test the sensitivity to the initial value of $\sigma_0$ of Algorithm 1. We only report the results when $\kappa = 10$ for succinctness. In Figure 2, for different cases of the tolerance $\epsilon$ in the stopping criterion (6.5), we plot the evaluations of the iteration numbers with respect to different choices of $\sigma_0$ whose values vary from $10^{-2}$ to $10^3$ with an equal distance of 0.1, where the horizontal axis is in log scale. It is clearly demonstrated that Algorithm 1 is much more robust to the value of the initial penalty parameter especially when $\sigma_0$ is larger equal than 10.0, even when high-precision solutions are pursued. This is a significant advantage for implementing ADMM-type algorithms.

### 6.4 Conclusions

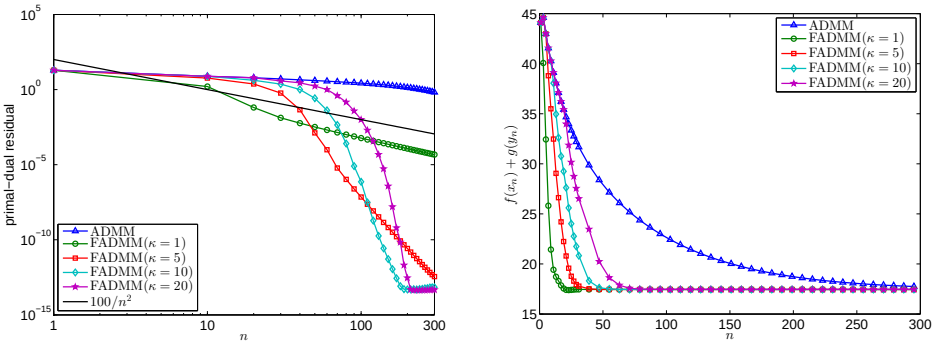
Based on the numerical experiments, we find that for most of the cases, Algorithm 1 with a given initial value of the penalty parameter outperforms the original ADMM (1.3) with the same constant penalty parameter; thus the theoretically faster $O(1/n^2)$ convergence rate is numerically verified. Moreover, our preliminary numerical results show that Algorithm 1 can pursue solutions in very high precisions with few iterations; this can be hardly achieved by the original ADMM (1.3) with a constant penalty parameter unless it is very well tuned. Last, in our experiments, we also show that Algorithm 1 is much less sensitive to the initial value of the penalty parameter. Indeed, for the LASSO model, Algorithm 1 is very robust with respect to the initial value of $\sigma_0$. These features indicate that compared with the




(a) Initial $\sigma_0 = 10.0$ (b) Initial $\sigma_0 = 10.0$



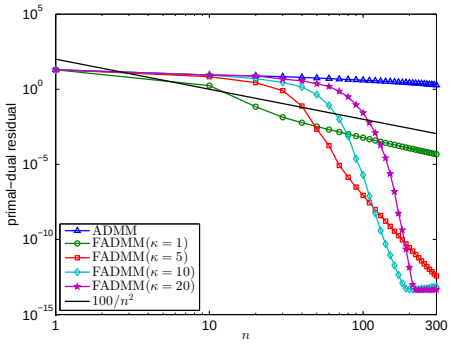

(c) Initial $\sigma_0 = 50.0$ (d) Initial $\sigma_0 = 50.0$



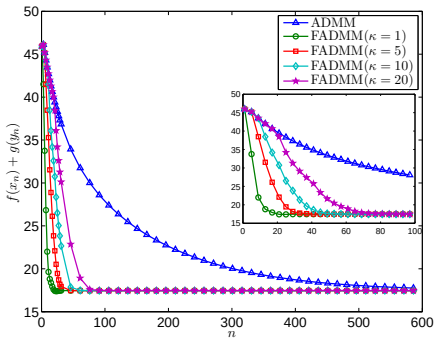

(e) Initial $\sigma_0 = 100.0$ (f) Initial $\sigma_0 = 100.0$

Figure 1: The decay of the primal-dual residual and objective function, for LASSO model by ADMM and faster ADMM with $\kappa = 1, 5, 10, 20$ and different initial penalty parameter $\sigma_0$, the tolerance $\varepsilon = 1.0 \times 10^{-4}$ in the stopping criterion (6.5).

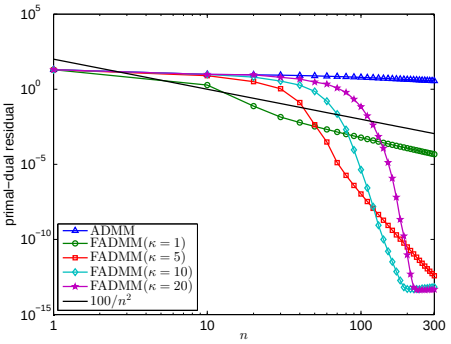
original ADMM (1.3) with a constant penalty parameter, theoretically Algorithm 1 has a higher order of worst-case convergence rate and numerically it performs more efficiently and robustly. These user-favorable features make Algorithm 1 more attractive to a number of applications.



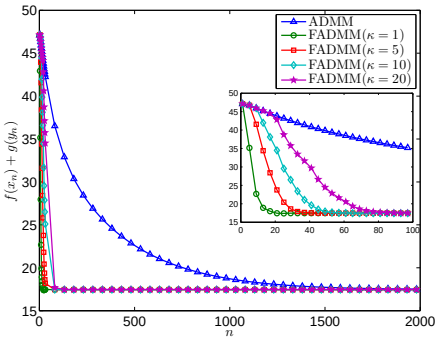

(g) Initial $\sigma_0 = 200.0$



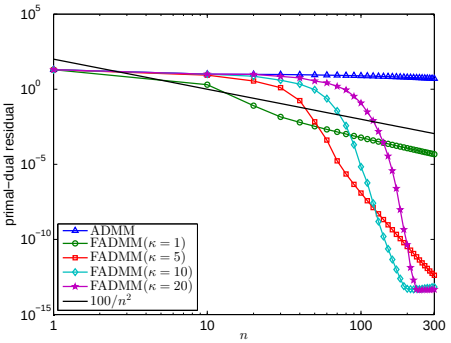

(h) Initial $\sigma_0 = 200.0$



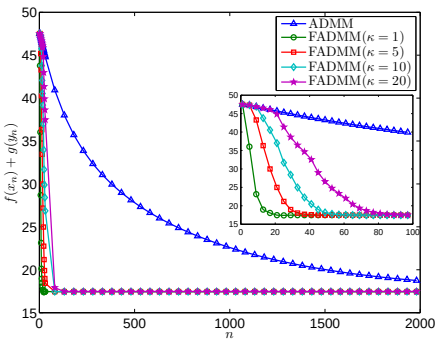

(i) Initial $\sigma_0 = 500.0$




(j) Initial $\sigma_0 = 500.0$




(k) Initial $\sigma_0 = 1000.0$



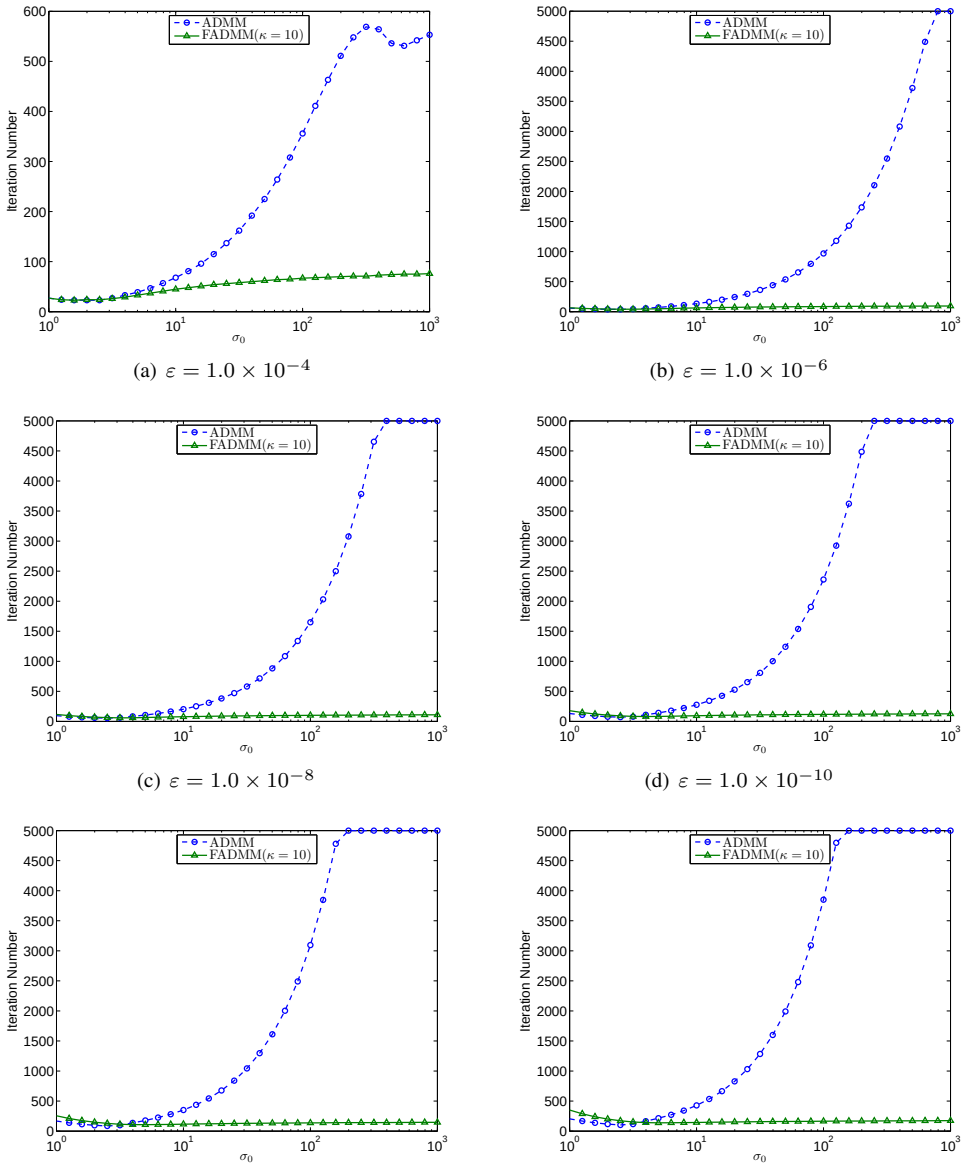

(l) Initial $\sigma_0 = 1000.0$

Figure 1: (con't) The decay of the primal-dual residual and objective function, for LASSO model by ADMM and faster ADMM with $\kappa = 1, 5, 10, 20$ and different initial penalty parameter $\sigma_0$, the tolerance $\varepsilon = 1.0 \times 10^{-4}$ in the stopping criterion (6.5).

## References


[1] H. H. BAUSCHKE AND P. L. COMBETTES, *Convex analysis and monotone operator theory in Hilbert spaces*, Springer, New York, 2011.

[2] D. BOLEY, *Local linear convergence of the alternating direction method of multipliers on quadratic or linear programs*, SIAM J. Optim., 23 (2013), pp. 2183–2207.




(a) $\varepsilon = 1.0 \times 10^{-4}$

(b) $\varepsilon = 1.0 \times 10^{-6}$

(c) $\varepsilon = 1.0 \times 10^{-8}$

(d) $\varepsilon = 1.0 \times 10^{-10}$

(e) $\varepsilon = 1.0 \times 10^{-12}$

(f) $\varepsilon = 1.0 \times 10^{-14}$

Figure 2: The iteration numbers of ADMM and faster ADMM with different initial parameter $\sigma_0$ and $\kappa = 10$ for solving LASSO model with different tolerance in the stopping criterion (6.5).

[3] S. BOYD, N. PARIKH, E. CHU, B. PELEATO, AND J. ECKSTEIN, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Found. Trends Mach. Learning, 3 (2010), pp. 1–122.

[4] A. CHAMBOLLE AND T. POCK, *A first-order primal-dual algorithm for convex problems with applications to imaging*, J. Math. Imaging Vision, 40 (2011), pp. 120–145.

[5] ——, *On the ergodic convergence rates of a first-order primal-dual algorithm*, Math. Program., (2015).

[6] T. F. CHAN AND R. GLOWINSKI, *Finite element approximation and iterative solution of a class of mildly nonlinear elliptic equations*, Technical Report, Computer Science Department, Stanford University, CA, 1978.

[7] E. CORMAN AND X. M. YUAN, *A generalized proximal point algorithm and its convergence rate*, SIAM J. Optim., 24 (2014), pp. 1614–1638.

[8] D. DAVIS AND W. YIN, *Convergence rate analysis of several splitting schemes*, arXiv:1406.4834, (2014).

[9] ——, *Faster convergence rates of relaxed Peaceman-Rachford and ADMM under regularity assumptions*, arXiv:1407.5210, (2014).

[10] D. L. DONOHO AND Y. TSAIG, *Fast solution of $\ell_1$-norm minimization problems when the solution may be sparse*, IEEE Trans. Inform. Theory, 54 (2008), pp. 4789–4812.

[11] J. DOUGLAS, JR. AND H. H. RACHFORD, JR., *On the numerical solution of heat conduction problems in two and three space variables*, Trans. Amer. Math. Soc., 82 (1956), pp. 421–439.

[12] J. ECKSTEIN, *Some saddle-function splitting methods for convex programming*, Optim. Methods Softw., 4 (1994), pp. 75–83.

[13] J. ECKSTEIN AND D. P. BERTSEKAS, *On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators*, Math. Program., 55 (1992), pp. 293–318.

[14] J. ECKSTEIN AND W. YAO, *Understanding the convergence of the alternating direction method of multipliers: Theoretical and computational perspectives*, Pac. J. Optim., 11 (2015), pp. 619–644.

[15] D. GABAY, *Applications of the method of multipliers to variational inequalities*, in Augmented Lagrangian Methods: Applications to the Solution of Boundary-Valued Problems, M. Fortin and R. Glowinski, eds., North-Holland Publishing Co., Amsterdam, 1983, pp. 299–331.

[16] D. GABAY AND B. MERCIER, *A dual algorithm for the solution of nonlinear variational problems via finite element approximation*, Comput. Math. Appl., 2 (1976), pp. 17–40.

[17] R. GLOWINSKI, *On alternating direction methods of multipliers: A historical perspective*, in Modeling, Simulation and Optimization for Science and Technology, W. Fitzgibbon, Y. A. Kuznetsov, P. Neittaanm?ki, and O. Pironneau, eds., Springer Netherlands, 2014, pp. 59–82.

[18] R. GLOWINSKI AND A. MARROCCO, *Sur l'approximation par éléments finis et la résolution par pénalisation-dualité d'une classe de problèmes de Dirichlet non linéaires*, R.A.I.R.O., R2 (1975), pp. 41–76.

[19] T. GOLDSTEIN, B. O'DONOGHUE, S. SETZER, AND R. BARANIUK, *Fast alternating direction optimization methods*, SIAM J. Imaging Sci., 7 (2014), pp. 1588–1623.

[20] D. HAN AND X. M. YUAN, *Local linear convergence of the alternating direction method of multipliers for quadratic programs*, SIAM J. Numer. Anal., 51 (2013), pp. 3446–3457.

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

[22] B. S. HE, Y. F. YOU, AND X. M. YUAN, *On the convergence of primal-dual hybrid gradient algorithm*, SIAM J. Imaging Sci., 7 (2014), pp. 2526–2537.

[23] B. S. HE AND X. M. YUAN, *Convergence analysis of primal-dual algorithms for a saddle-point problem: from contraction perspective*, SIAM J. Imaging Sci., 5 (2012), pp. 119–149.

[24] ——, *On the $O(1/n)$ convergence rate of the Douglas-Rachford alternating direction method*, SIAM J. Numer. Anal., 50 (2012), pp. 700–709.

[25] ——, *On non-ergodic convergence rate of Douglas-Rachford alternating direction method of multipliers*, Numer. Math., 130 (2015), pp. 567–577.

[26] M. R. HESTENES, *Multiplier and gradient methods*, J. Optim. Theory Appl., 4 (1969), pp. 303–320.

[27] P.-L. LIONS AND B. MERCIER, *Splitting algorithms for the sum of two nonlinear operators*, SIAM J. Numer. Anal., 16 (1979), pp. 964–979.

[28] B. MARTINET, *Régularisation dinéquations variationnelles par approximations successives*, Rev. Française Informat. Recherche Opérationnelle, 4 (1970), pp. 154–159.

[29] J. MOREAU, *Proximité et dualité dans un espace hilbertien*, Bull. Soc. Math. France, 93 (1965), pp. 273–299.

[30] A. NEMIROVSKI, *Prox-method with rate of convergence $O(1/t)$ for variational inequalities with Lipschitz continuous monotone operators and smooth convex-concave saddle point problems*, SIAM J. Optim., 15 (2004), pp. 229–251.

[31] Y. NESTEROV, *Gradient methods for minimizing composite functions*, Math. Program., 140 (2013), pp. 125–161.

[32] Y. E. NESTEROV, *A method for solving the convex programming problem with convergence rate $O(1/k^2)$*, Dokl. Akad. Nauk SSSR, 269 (1983), pp. 543–547. (In Russian. Translated in Soviet Math. Dokl., 27 (1983), pp. 372-376.).

[33] M. J. D. POWELL, *A method for nonlinear constraints in minimization problems*, in Optimization, R. Fletcher, ed., Academic Press, New York, 1969, pp. 283–298.

[34] R. T. ROCKAFELLAR, *Convex analysis*, Princeton University Press, Princeton, NJ, 1997.

[35] R. SHEFI, *Rate of convergence analysis for convex nonsmooth optimization algorithms*, PhD Thesis, Tel Aviv University, Israel, 2015.

[36] M. TAO AND X. M. YUAN, *On the optimal linear convergence rate of a generalized proximal point algorithm*, arXiv:1605.05474, (2015).

[37] W. Y. TIAN AND X. M. YUAN, *Convergence analysis of primal-dual based methods for total variation minimization with finite element approximation*, submited, (2014).

[38] R. TIBSHIRANI, *Regression shrinkage and selection via the lasso*, J. Roy. Statist. Soc. Ser. B, 58 (1996), pp. 267–288.

[39] X. F. WANG AND X. M. YUAN, *The linearized alternating direction method of multipliers for Dantzig selector*, SIAM J. Sci. Comput., 34 (2012), pp. A2792–A2811.

[40] J. F. YANG AND X. M. YUAN, *Linearized augmented Lagrangian and alternating direction methods for nuclear norm minimization*, Math. Comp., 82 (2012), pp. 301–329.

[41] X. M. YUAN, *Alternating direction method for covariance selection models*, J. Sci. Comput., 51 (2012), pp. 261–273.

[42] X. ZHANG, M. BURGER, AND S. OSHER, *A unified primal-dual algorithm framework based on Bregman iteration*, J. Sci. Comput., 46 (2011), pp. 20–46.

Table 1: Iteration numbers of ADMM and faster ADMM for solving the LASSO model with different initial parameter $\sigma_0$ and different tolerance $\varepsilon$ in the stopping criterion (6.5). ("–" means the iteration number is beyond 5000)

| $\varepsilon$ | $10^{-4}$ | $10^{-6}$ | $10^{-8}$ | $10^{-10}$ | $10^{-12}$ | $10^{-14}$ |
|---|---|---|---|---|---|---|
| | $\sigma_0 = 1.0$ | | | | | |
| ADMM | 27 | 59 | 95 | 131 | 167 | 204 |
| FADMM($\kappa = 1$) | 57 | 495 | 4061 | "–" | "–" | "–" |
| FADMM($\kappa = 5$) | 30 | 82 | 166 | 296 | 498 | "–" |
| FADMM($\kappa = 10$) | 28 | 69 | 122 | 189 | 274 | 379 |
| FADMM($\kappa = 20$) | 27 | 63 | 106 | 154 | 209 | 270 |
| | $\sigma_0 = 10.0$ | | | | | |
| ADMM | 92 | 160 | 232 | 306 | 382 | 459 |
| FADMM($\kappa = 1$) | 36 | 254 | 2051 | "–" | "–" | "–" |
| FADMM($\kappa = 5$) | 34 | 50 | 73 | 114 | 177 | 277 |
| FADMM($\kappa = 10$) | 46 | 61 | 77 | 93 | 114 | 143 |
| FADMM($\kappa = 20$) | 60 | 84 | 102 | 116 | 133 | 149 |
| | $\sigma_0 = 20.0$ | | | | | |
| ADMM | 182 | 319 | 462 | 610 | 762 | 916 |
| FADMM($\kappa = 1$) | 37 | 251 | 2021 | "–" | "–" | "–" |
| FADMM($\kappa = 5$) | 39 | 55 | 77 | 117 | 178 | 275 |
| FADMM($\kappa = 10$) | 56 | 71 | 87 | 103 | 123 | 151 |
| FADMM($\kappa = 20$) | 77 | 103 | 122 | 136 | 153 | 168 |
| | $\sigma_0 = 50.0$ | | | | | |
| ADMM | 450 | 791 | 1148 | 1516 | 1894 | 2277 |
| FADMM($\kappa = 1$) | 37 | 251 | 2012 | "–" | "–" | "–" |
| FADMM($\kappa = 5$) | 43 | 59 | 81 | 120 | 181 | 277 |
| FADMM($\kappa = 10$) | 64 | 80 | 96 | 112 | 131 | 159 |
| FADMM($\kappa = 20$) | 94 | 120 | 139 | 154 | 171 | 186 |
| | $\sigma_0 = 100.0$ | | | | | |
| ADMM | 897 | 1577 | 2289 | 3024 | 3778 | 4544 |
| FADMM($\kappa = 1$) | 38 | 251 | 2009 | "–" | "–" | "–" |
| FADMM($\kappa = 5$) | 46 | 61 | 83 | 122 | 183 | 279 |
| FADMM($\kappa = 10$) | 69 | 84 | 101 | 116 | 135 | 164 |
| FADMM($\kappa = 20$) | 103 | 129 | 148 | 163 | 180 | 197 |
| | $\sigma_0 = 200.0$ | | | | | |
| ADMM | 1790 | 3149 | 4571 | "–" | "–" | "–" |
| FADMM($\kappa = 1$) | 38 | 251 | 2010 | "–" | "–" | "–" |
| FADMM($\kappa = 5$) | 47 | 63 | 85 | 124 | 185 | 281 |
| FADMM($\kappa = 10$) | 72 | 88 | 104 | 120 | 139 | 167 |
| FADMM($\kappa = 20$) | 109 | 135 | 155 | 170 | 187 | 203 |
| | $\sigma_0 = 500.0$ | | | | | |
| ADMM | 4471 | "–" | "–" | "–" | "–" | "–" |
| FADMM($\kappa = 1$) | 38 | 251 | 2010 | "–" | "–" | "–" |
| FADMM($\kappa = 5$) | 49 | 65 | 87 | 126 | 187 | 283 |
| FADMM($\kappa = 10$) | 76 | 91 | 108 | 123 | 142 | 171 |
| FADMM($\kappa = 20$) | 117 | 143 | 163 | 177 | 194 | 208 |
| | $\sigma_0 = 1000.0$ | | | | | |
| ADMM | "–" | "–" | "–" | "–" | "–" | "–" |
| FADMM($\kappa = 1$) | 39 | 251 | 2009 | "–" | "–" | "–" |
| FADMM($\kappa = 5$) | 50 | 66 | 88 | 127 | 188 | 284 |
| FADMM($\kappa = 10$) | 78 | 93 | 110 | 125 | 145 | 173 |
| FADMM($\kappa = 20$) | 121 | 147 | 167 | 181 | 199 | 215 |
| | $\sigma_0 = 2000.0$ | | | | | |
| ADMM | "–" | "–" | "–" | "–" | "–" | "–" |
| FADMM($\kappa = 1$) | 39 | 252 | 2009 | "–" | "–" | "–" |
| FADMM($\kappa = 5$) | 51 | 67 | 89 | 128 | 189 | 285 |
| FADMM($\kappa = 10$) | 80 | 95 | 112 | 127 | 147 | 175 |
| FADMM($\kappa = 20$) | 125 | 151 | 171 | 185 | 203 | 219 |