MATHEMATICS OF COMPUTATION
Volume 86, Number 306, July 2017, Pages 1821–1845
http://dx.doi.org/10.1090/mcom/3145
Article electronically published on October 20, 2016

ACCELERATED UZAWA METHODS FOR CONVEX OPTIMIZATION

MIN TAO AND XIAOMING YUAN

Abstract. We focus on a linearly constrained strongly convex minimization model and discuss the application of the classical Uzawa method. Our principal goal is to show that some existing acceleration schemes can be used to accelerate the Uzawa method in the sense that the worst-case convergence rate (measured by the iteration complexity) of the resulting accelerated Uzawa schemes is $O(1/k^2)$ where k represents the iteration counter. Our discussion assumes that the objective function is given by a black-box oracle; thus an inexact version of the Uzawa method with a sequence of dynamically-chosen step sizes is implemented. A worst-case convergence rate of $O(1/k)$ is also shown for this inexact version. Some preliminary numerical results are reported to verify the acceleration effectiveness of the accelerated Uzawa schemes and their superiority over some existing methods.

1. Introduction

We consider the strongly convex minimization model with linear equality or inequality constraints

$$\begin{array}{ll} \min & f(x) \\ s.t. & Ax = \ (\text{or } \leq)\ b, \\ & x \in \mathcal{X}, \end{array} \tag{1.1}$$

where $f : \Re^n \to \Re$ is a strongly convex but not necessarily smooth function, $A \in \Re^{m\times n}$, $b \in \Re^m$ and $\mathcal{X} \subseteq \Re^n$ is a convex closed set. In our discussion, $\mathcal{X}$ is assumed to be simple in the sense that the projection onto it under the Euclidean norm is easy to compute. The objective function in (1.1) is assumed to be given by a black-box oracle (see, e.g., [37]); thus estimating the strong convexity modulus of $f(x)$ is not possible. The solution set of (1.1) is assumed to be nonempty.

To explain the rationale of assuming the strong convexity on $f(x)$, a good example is the application of the linearized Bregman scheme in [27] to some sparse or low-rank optimization models which have wide applications in varying domains; see, e.g., [14–16, 38]. For such an application whose original objective function is convex (e.g., $\|x\|_1 := \sum_{i=1}^n |x_i|$ in a vector space $\Re^n$ or $\|X\|_*$ in a matrix space $\Re^{m\times n}$ where the nuclear norm $\|X\|_*$ is defined as the sum of the absolute values of

Received by the editor March 31, 2015 and, in revised form, January 28, 2016.
2010 *Mathematics Subject Classification.* Primary 90C25; Secondary 94A08.
Key words and phrases. Convex optimization, Uzawa method, acceleration, convergence rate, black-box oracle.
The first author was supported by the Natural Science Foundation of China grant NSFC-11301280, and the Fundamental Research Funds for the Central Universities, 020314330019.
The second author was supported in part by the General Research Fund from Hong Kong Research Grants Council, HKBU 12300515.

all singular values of X), sometimes a thresholding operation is required to truncate the iterate in certain domains in order to produce a solution with the sparse or low-rank feature. As delineated in [14, 44], this truncation procedure amounts to adding a quadratic term to the original convex objective function; a strongly convex objective function is thus yielded. For instance, the new objective function is $\|x\|_1 + \frac{1}{2\tau}\|x\|^2$ if the original objective function is $\|x\|_1$, where τ is a threshold value used in the implementation of the linearized Bregman scheme. Other examples include the total-variational image denoising model (e.g., [17, 42]), matrix completion model in [14] and robust principal component analysis models in [45]. Also, we often encounter the model (1.1) as subproblems when implementing the proximal point algorithm [32, 41] in various contexts and the smoothing approach for nonsmooth optimization problems [36].

Let the Lagrangian function of (1.1) be

$$\mathcal{L}(x, \lambda) = f(x) + \lambda^\top (Ax - b), \tag{1.2}$$

with $\lambda \in \Re^m$ the Lagrange multiplier, and let

$$G(\lambda) := \min_{x \in \mathcal{X}} \mathcal{L}(x, \lambda). \tag{1.3}$$

Note that the minimum in (1.3) is attained because of the strong convexity assumption of $f(x)$ and the assumption on $\mathcal{X}$. Then, the dual problem of (1.1) is

$$\begin{array}{ll} \max & G(\lambda) \\ s.t. & \lambda \in \Lambda, \end{array} \tag{1.4}$$

where $\Lambda = \Re^m$ if $Ax = b$ and $\Lambda = \Re^m_+$ if $Ax \le b$ in (1.1). A point $(x^*, \lambda^*) \in \mathcal{X} \times \Lambda$ satisfying

$$x^* = \arg\min_{x \in \mathcal{X}} \mathcal{L}(x, \lambda^*) \ \text{ and } \ \lambda^* = \arg\max_{\lambda \in \Lambda} \mathcal{L}(x^*, \lambda)$$

is called a saddle point of $\mathcal{L}(x, \lambda)$. We denote by $\mathcal{X}^* \times \Lambda^*$ the set of all saddle points of $\mathcal{L}(x, \lambda)$.

The Uzawa method appearing early in [1] is probably the most fundamental method for finding a saddle point of $\mathcal{L}(x, \lambda)$. In fact, it has been playing a significant theoretical and algorithmic role in a variety of scientific computing areas. In particular, a number of celebrated algorithms are relevant to the Uzawa method, such as the modified Arrow-Hurwicz method in [39], extra-gradient method in [30], Douglas-Rachford operator splitting method in [31], alternating direction methods of multipliers (ADMM) in [25] (see also [24]), and a class of primal-dual algorithms in [18, 23, 28, 46, 47]. We refer to, e.g., [3, 11–13, 18, 22] for some applications of the Uzawa method in partial differential equations, [12, 26] for its convergence rate analysis, [4] for the relationship between the Uzawa method, and [29, 40] for the augmented Lagrangian method.

Applying the Uzawa method to (1.1), the iterative scheme reads as

$$\left\{ \begin{array}{l} \lambda^{k+1} = \arg\max_{\lambda \in \Lambda} \{\mathcal{L}(x^k, \lambda) - \frac{1}{2\beta}\|\lambda - \lambda^k\|^2\}, \\ x^{k+1} = \arg\min_{x \in \mathcal{X}} \{\mathcal{L}(x, \lambda^{k+1})\}, \end{array} \right. \tag{1.5}$$

where $\beta > 0$ is regarded as a step size. To ensure the convergence, the step size β in (1.5) should be chosen restrictively as $0 < \beta < 2/L(\nabla G)$, where ∇G denotes the gradient of G and $L(\nabla G)$ is the Lipschitz constant of ∇G. As we know (see Lemma 2.1 in Section 2), under the strong convexity assumption of $f(x)$, $G(\lambda)$ defined in

(1.3) is continuously differentiable and ∇G is Lipschitz continuous. Note that the λ-subproblem in (1.5) can be specified as

$$\lambda^{k+1} = P_\Lambda[\lambda^k + \beta(Ax^k - b)],$$

where P_Λ denotes the projection onto Λ under the Euclidean norm. Recall the simplicity of Λ (either $\Re^m$ or $\Re^m_+$). Thus, the λ-subproblem in (1.5) is easy. Moreover, with the simplicity assumption of $\mathcal{X}$, the difficulty of the x-subproblem in (1.5) depends essentially on the difficulty of $f(x)$ itself.

Since the constant $L(\nabla G)$ depends on the strong convexity modulus of $f(x)$ (see Lemma 2.1 in Section 2) while it is not possible to estimate this strong convexity modulus when $f(x)$ is given by a black-box oracle, we are interested in seeking a strategy to determine the step size β for the Uzawa method in absence of $L(\nabla G)$. We will propose such a strategy and show that the Uzawa method with this step size strategy (we call it an inexact Uzawa method) has a worst-case $O(1/k)$ convergence rate.[1] The main purpose of this paper is to show that this inexact Uzawa method can be accelerated by the acceleration schemes in [43]. Here, "acceleration" means the worst-case convergence rate of these accelerated inexact Uzawa methods is $O(1/k^2)$, which is considered as a faster convergence rate than $O(1/k)$.

The rest of this paper is organized as follows. In Section 2, we summarize some preliminary results which are useful for further analysis. Then, in Section 3, we propose an inexact Uzawa method whose step size is determined in absence of the strong convexity modulus of $f(x)$ and establish its worst-case $O(1/k)$ convergence rate. In Sections 4 and 5, we apply the acceleration schemes in [43] to accelerate the new inexact Uzawa method, and derive the worst-case $O(1/k^2)$ convergence rates for these accelerated Uzawa methods. In Section 6, we report some numerical results to verify the acceleration effectiveness of the accelerated Uzawa methods. Finally, we make some conclusions and mention some future work in Section 7.

2. Preliminaries

In this section, we summarize some known results which will be used in later analysis. We first summarize some properties of the function $G(\lambda)$ defined in (1.3).

Lemma 2.1. *Let $f(x)$ in (1.1) be strongly convex with the modulus σ (but σ is unknown because $f(x)$ is assumed to be given by a black-box oracle), and let $G(\lambda)$ be defined in (1.3). Then we have:*

1) *$G(\lambda)$ is concave and continuously differentiable at any $\lambda \in \Lambda$.*
2) *The gradient of $G(\lambda)$ is given by*

$$\nabla G(\lambda) = Ax(\lambda) - b, \tag{2.1}$$

where

$$x(\lambda) := \arg\min_{x\in\mathcal{X}} \mathcal{L}(x, \lambda). \tag{2.2}$$

3) *$\nabla G(\lambda)$ is Lipschitz continuous with the Lipschitz constant*

$$L(\nabla G) = \|A^\top A\|/\sigma.$$

[1] We follow the work [34, 35] and many others to measure the worst-case convergence rate in terms of the iteration complexity. That is, a worst-case $O(1/k)$ convergence rate means the accuracy to a solution point under certain criteria is of the order $O(1/k)$ after k iterations of an iterative scheme; or, equivalently, it requires at most $O(1/\epsilon)$ iterations to achieve an approximate solution with an accuracy of ϵ.

Proof. See, e.g., Theorem 1 in [36]. □

In the following lemmas, we summarize some well-known properties whose proofs can be found easily in the literature.

Lemma 2.2. *Let $\{a_k\}$ and $\{b_k\}$ be positive sequences of real numbers that satisfy*

$$a_k - a_{k+1} \geq b_{k+1} - b_k \quad \forall k \geq 1, \text{with} \quad a_1 + b_1 \leq c, \quad c > 0.$$

Then we have $a_k \leq c$ for every $k \geq 1$.

Proof. See, e.g., [5]. □

Lemma 2.3. *Let Ω be a closed convex subset in $\Re^n$ and P_Ω denote the projection onto Ω under the Euclidean norm. Then we have*

$$(y - P_\Omega(y))^\top (P_\Omega(y) - x) \geq 0, \quad \forall x \in \Omega, y \in \mathcal{R}^n. \tag{2.3}$$

Proof. See, e.g., [8]. □

3. An inexact Uzawa method with a worst-case $O(1/k)$ convergence rate

In this section, we propose the following inexact Uzawa method for finding a saddle point of $\mathcal{L}(x, \lambda)$ when the strong convexity modulus σ of $f(x)$ is unknown.

Algorithm 1: An inexact Uzawa method for (1.1)
Initialization. Take $\lambda^0 \in \Lambda$, set $x^0 = \arg\min_{x \in \mathcal{X}} \mathcal{L}(x, \lambda^0)$ and choose $\beta_0 > 0$.
Step k. Find $(\lambda^k, x^k, \beta_k)$ such that

$$\lambda^k = P_\Lambda[\lambda^{k-1} + \beta_k(Ax^{k-1} - b)], \tag{3.1a}$$

$$x^k = \arg\min_{x \in \mathcal{X}} \mathcal{L}(x, \lambda^k), \tag{3.1b}$$

and the following requirement is satisfied:

$$\beta_k |(\lambda^{k-1} - \lambda^k)^\top A(x^{k-1} - x^k)| \leq \frac{1}{2}\|\lambda^{k-1} - \lambda^k\|^2. \tag{3.1c}$$

We make some remarks on Algorithm 1.

Remark 3.1. Recall the equation (2.1). We easily see that Algorithm 1 can be written as

$$\lambda^k = P_\Lambda[\lambda^{k-1} + \beta_k \nabla G(\lambda^{k-1})],$$

where the step size β_k is subject to the criterion (3.1c). Therefore, it differs from the application of the projected gradient method to (1.1)'s dual problem (1.4) in that the step size β is chosen in absence of the Lipschitz constant of $\nabla G(\lambda)$. Note that the step size β in the original Uzawa method (1.5) is determined by knowing the Lipschitz constant of $\nabla G(\lambda)$. Algorithm 1 is thus an inexact version of the original Uzawa scheme (1.5).

Remark 3.2. Recall the Lipschitz continuity of ∇G and the concavity of G shown in Lemma 2.1. Then it is easy to see that the criterion (3.1c) can be fulfilled by reducing the value of β_k finitely many times. In fact, whenever $\beta_k < \sigma/(2\|A^\top A\|)$, the criterion (3.1c) is satisfied. Therefore, the sequence $\{\beta_k\}$ generated by Algorithm 1 has a uniform lower bound $\beta_{\min} > 0$. That is, $\beta_k \geq \beta_{\min} > 0$ for any integer k. This shares the same feature as some classical Armijo-like line-search rules for unconstrained optimization models such as those in [7]. On the other hand, since

the criterion (3.1c) is satisfied by reducing the value of β_k, obviously we can assume there exists a uniform upper bound $\beta_{\max} > 0$ such that $\beta_k \leq \beta_{\max}$ for any integer k. Thus, the sequence $\{\beta_k\}$ generated by Algorithm 1 can be regarded as bounded: $\beta_{\max} \geq \beta_k \geq \beta_{\min} > 0$ for any k. We will provide a specific strategy for choosing β_k in Section 6.1.

Remark 3.3. The criterion (3.1c) can be easily implemented; there is no need to evaluate any function value or to estimate $\|A^\top A\|$ and the strong convexity modulus of f. As will be shown by numerical experiments, the criterion (3.1c) has some numerical advantages such that it is robust to the initial choice of β_0. These features make it different from existing backtracking strategies of finding step sizes for some projected gradient type methods in, e.g., [5, 37].

In the following, we show that for the sequence $\{L(x^k, \lambda^k)\}$, where (x^k, λ^k) is generated by Algorithm 1, we have

$$\mathcal{L}(x^*, \lambda^*) - \mathcal{L}(x^k, \lambda^k) \leq \epsilon,$$

where $\epsilon = O(1/k)$ and (x^*, λ^*) is a saddle point of $\mathcal{L}(x, \lambda)$. In other words, the sequence $\{\mathcal{L}(x^k, \lambda^k)\}$ converges to $\mathcal{L}(x^*, \lambda^*)$ on a worst-case convergence rate of $O(1/k)$ where the rate is measured by the iteration complexity. Note that our analysis in this section is different from some existing results in the literature such as that in [18] where a general saddle point problem is considered. We first prove a lemma which is useful for establishing the worst-case $O(1/k)$ convergence rate for Algorithm 1.

Lemma 3.1. *Let $\{(x^k, \lambda^k)\}$ be generated by Algorithm* 1 *and let $x(\lambda)$ be given by* (2.2) *for $\lambda \in \Lambda$. Then we have*

$$2\beta_k(\mathcal{L}(x^k, \lambda^k) - \mathcal{L}(x(\lambda), \lambda)) \geq \|\lambda^k - \lambda\|^2 - \|\lambda^{k-1} - \lambda\|^2, \ \forall \lambda \in \Lambda. \tag{3.2}$$

Proof. First, for (x^{k-1}, λ^{k-1}) generated by Algorithm 1, it follows from (3.1b) that there exists $\bar{f}(x^{k-1}) \in \partial f(x^{k-1})$ such that

$$(x^k - x^{k-1})^\top (\bar{f}(x^{k-1}) + A^\top \lambda^{k-1}) \geq 0.$$

Together with this inequality, the convexity of $f(x)$ and the inequality in (3.1c) enable us to derive

$$\begin{aligned}
&\mathcal{L}(x^{k-1}, \lambda^{k-1}) - \mathcal{L}(x^k, \lambda^k) \\
&\quad = f(x^{k-1}) - f(x^k) + (\lambda^{k-1})^\top (Ax^{k-1} - b) - (\lambda^k)^\top (Ax^k - b) \\
&\leq (x^{k-1} - x^k)^\top \bar{f}(x^{k-1}) + (\lambda^{k-1})^\top (Ax^{k-1} - b) - (\lambda^k)^\top (Ax^k - b) \\
&\leq (x^k - x^{k-1})^\top A^\top \lambda^{k-1} + (\lambda^{k-1})^\top (Ax^{k-1} - b) - (\lambda^k)^\top (Ax^k - b) \\
&\leq \frac{1}{2\beta_k} \|\lambda^{k-1} - \lambda^k\|^2 - (\lambda^k)^\top A(x^{k-1} - x^k) + (\lambda^{k-1})^\top (Ax^{k-1} - b) \\
&- (\lambda^k)^\top (Ax^k - b) \\
&= \frac{1}{2\beta_k} \|\lambda^{k-1} - \lambda^k\|^2 + (\lambda^{k-1} - \lambda^k)^\top (Ax^{k-1} - b).
\end{aligned} \tag{3.3}$$

Similarly, for $\lambda \in \Lambda$, it follows from (3.1b) that there exists $\bar{f}(x(\lambda)) \in \partial f(x(\lambda))$ such that

$$(x^{k-1} - x(\lambda))^\top (\bar{f}(x(\lambda)) + A^\top \lambda) \geq 0.$$

We thus have

$$\begin{aligned} &\mathcal{L}(x^{k-1}, \lambda^{k-1}) - \mathcal{L}(x(\lambda), \lambda) \\ &\quad = f(x^{k-1}) - f(x(\lambda)) + (\lambda^{k-1})^\top (Ax^{k-1} - b) - \lambda^\top (Ax(\lambda) - b) \\ &\quad \geq (x^{k-1} - x(\lambda))^\top \bar{f}(x(\lambda)) + (\lambda^{k-1})^\top (Ax^{k-1} - b) - \lambda^\top (Ax(\lambda) - b) \\ &\quad \geq -(x^{k-1} - x(\lambda))^\top A^\top \lambda + (\lambda^{k-1})^\top (Ax^{k-1} - b) - \lambda^\top (Ax(\lambda) - b) \\ &\quad = (\lambda^{k-1} - \lambda)^\top (Ax^{k-1} - b), \ \ \forall \lambda \in \Lambda. \end{aligned} \tag{3.4}$$

Subtracting (3.4) by (3.3), we get

$$\mathcal{L}(x^k, \lambda^k) - \mathcal{L}(x(\lambda), \lambda) \geq -\frac{1}{2\beta_k} \|\lambda^{k-1} - \lambda^k\|^2 + (\lambda^k - \lambda)^\top (Ax^{k-1} - b), \ \ \forall \lambda \in \Lambda. \tag{3.5}$$

Rearranging the first-order optimality condition of (3.1a), we have

$$(\lambda^k - \lambda)^\top (Ax^{k-1} - b) \geq \frac{1}{\beta_k} (\lambda^k - \lambda)^\top (\lambda^k - \lambda^{k-1}), \ \ \forall \lambda \in \Lambda.$$

Substituting the above inequality into (3.5), we derive

$$\mathcal{L}(x^k, \lambda^k) - \mathcal{L}(x(\lambda), \lambda) \geq \frac{1}{2\beta_k} (\|\lambda^k - \lambda\|^2 - \|\lambda^{k-1} - \lambda\|^2), \ \ \forall \lambda \in \Lambda.$$

This implies the assertion (3.2) immediately. The proof is complete. □

Remark 3.4. In (3.2), if we take $(x(\lambda), \lambda) = (x^{k-1}, \lambda^{k-1})$ generated by Algorithm 1, then it follows that

$$\mathcal{L}(x^k, \lambda^k) - \mathcal{L}(x^{k-1}, \lambda^{k-1}) \geq \frac{1}{2\beta_k} \|\lambda^k - \lambda^{k-1}\|^2. \tag{3.6}$$

Thus, the sequence $\{\mathcal{L}(x^k, \lambda^k)\}$ is monotonically nondecreasing.

Now, we are ready to prove that the sequence $\{\mathcal{L}(x^k, \lambda^k)\}$ converges to $\mathcal{L}(x^*, \lambda^*)$, where (x^*, λ^*) is a saddle point of $\mathcal{L}(x, \lambda)$, on an $O(1/k)$ worst-case convergence rate. Hence, a sublinear convergence rate of Algorithm 1 in the worst case is proved in terms of the objective residual of the Lagrangian function $\mathcal{L}(x, \lambda)$. Recall the sequence $\{\beta_k\}$ generated by Algorithm 1 is bounded: $\beta_{\max} \geq \beta_k \geq \beta_{\min} > 0$ for any k.

Theorem 3.1. *Let $\{(x^k, \lambda^k)\}$ be the sequence generated by Algorithm* 1. *Then we have*

$$0 \leq \mathcal{L}(x^*, \lambda^*) - \mathcal{L}(x^k, \lambda^k) \leq \frac{\|\lambda^0 - \lambda^*\|^2}{2\beta_{\min} k}, \tag{3.7}$$

where (x^*, λ^*) is a saddle point of $\mathcal{L}(x, \lambda)$.

Proof. The first inequality is trivial based on the definition of a saddle point. Now, we prove the second inequality. Taking $k = n$ in inequality (3.6) and using the fact that the sequence $\{\beta_k\}$ is bounded, we have

$$2\beta_{\max}\Big((L(x^n,\lambda^n) - L(x^*,\lambda^*)) - (L(x^{(n-1)},\lambda^{(n-1)}) - L(x^*,\lambda^*))\Big) \geq \|\lambda^n - \lambda^{n-1}\|^2,$$

which implies the monotonicity of the sequence $\{(L(x^n,\lambda^n) - L(x^*,\lambda^*))\}$:

$$\begin{aligned} L(x^n,\lambda^n) - L(x^*,\lambda^*) &\geq L(x^{n-1},\lambda^{n-1}) - L(x^*,\lambda^*) \\ &\geq \cdots \geq L(x^0,\lambda^0) - L(x^*,\lambda^*). \end{aligned} \tag{3.8}$$

Invoking Lemma 3.1 with $x(\lambda) = x^*$, $\lambda = \lambda^*$, and $k = n$, we obtain

$$2\beta_{\min}(\mathcal{L}(x^n,\lambda^n) - \mathcal{L}(x^*,\lambda^*)) \geq \|\lambda^n - \lambda^*\|^2 - \|\lambda^{n-1} - \lambda^*\|^2.$$

Summarizing the inequality over $n = 1,\ldots,k$, we get

$$2\beta_{\min}\sum_{n=1}^{k}(\mathcal{L}(x^n,\lambda^n) - \mathcal{L}(x^*,\lambda^*)) \geq \|\lambda^k - \lambda^*\|^2 - \|\lambda^0 - \lambda^*\|^2. \tag{3.9}$$

Then, combining the fact (3.8) and inequality (3.9), we have

$$2\beta_{\min}k(\mathcal{L}(x^k,\lambda^k) - \mathcal{L}(x^*,\lambda^*)) \geq \|\lambda^k - \lambda^*\|^2 - \|\lambda^0 - \lambda^*\|^2.$$

This implies the assertion (3.7) immediately. □

4. Accelerated inexact Uzawa methods with worst-case $O(1/k^2)$ convergence rates

In this section, we show that the proposed Algorithm 1 can be accelerated by the acceleration schemes in [43]. As a result, some accelerated inexact Uzawa methods with worst-case $O(1/k^2)$ convergence rates are proposed. For the convenience of presenting these accelerated inexact Uzawa methods, from now on we use $(\tilde{x}^k,\tilde{\lambda}^k)$, rather than (x^k,λ^k), to denote the iterate generated by Algorithm 1.

As in [43], let $\{\theta_k\}$ be a scalar sequence satisfying

$$\frac{1-\theta_k}{\theta_k^2} \leq \frac{1}{\theta_{k-1}^2} \tag{4.1}$$

and $\theta_k \in (0,1]$. The choice (4.1) is a unified framework of the coefficients in some existing acceleration schemes, e.g., [2, 5, 35, 37].

4.1. **An algorithmic framework of accelerated inexact Uzawa methods.**

Let us first present an algorithmic framework of accelerated inexact Uzawa methods based on the combination of Algorithm 1 with the general acceleration scheme in [43].

Algorithm 2: An algorithmic framework of accelerated inexact Uzawa methods

Step 0. Take $\lambda^0 = \tilde{\lambda}^0 \in \Lambda$, set $x^0 = \arg\min_{x\in\mathcal{X}} \mathcal{L}(x, \lambda^0)$ and choose $\beta_0 > 0$. Let $\{\theta_k\}$ be a sequence satisfying (4.1).

Step k. Generate the new iterate (λ^k, x^k) by the following steps.

- Find $(\tilde{\lambda}^k, \tilde{x}^k, \beta_k)$ such that

$$\tilde{\lambda}^k = P_\Lambda[\lambda^{k-1} + \beta_k(Ax^{k-1} - b)], \tag{4.2a}$$

$$\tilde{x}^k = \arg\min_{x\in\mathcal{X}} \mathcal{L}(x, \tilde{\lambda}^k), \tag{4.2b}$$

and the following requirement is satisfied:

$$\beta_k|(\lambda^{k-1} - \tilde{\lambda}^k)^\top A(x^{k-1} - \tilde{x}^k)| \le \frac{1}{2}\|\lambda^{k-1} - \tilde{\lambda}^k\|^2. \tag{4.2c}$$

- Set

$$v^k = \tilde{\lambda}^{k-1} + \frac{1}{\theta_k}(\tilde{\lambda}^k - \tilde{\lambda}^{k-1}). \tag{4.3}$$

- Set

$$\lambda^k = (1 - \theta_{k+1})\tilde{\lambda}^k + \theta_{k+1}v^k. \tag{4.4}$$

$$x^k = \arg\min_{x\in\mathcal{X}} \mathcal{L}(x, \lambda^k). \tag{4.5}$$

The following lemma can be easily proved by an analysis similar to that in Lemma 3.1.

Lemma 4.1. *Let $\{(\tilde{x}^k, \tilde{\lambda}^k)\}$ be generated by Algorithm 2 and let $x(\lambda)$ be given by (2.2) for $\lambda \in \Lambda$. Then we have*

(4.6)
$$\mathcal{L}(\tilde{x}^k, \tilde{\lambda}^k) - \mathcal{L}(x(\lambda), \lambda) \ge -\frac{1}{2\beta_k}\|\tilde{\lambda}^k - \lambda^{k-1}\|^2 + \frac{1}{\beta_k}(\tilde{\lambda}^k - \lambda)^\top(\tilde{\lambda}^k - \lambda^{k-1}), \quad \forall \lambda \in \Lambda.$$

Proof. The proof is similar to Lemma 3.1, and is thus omitted. □

Now, we analyze how fast the sequence $\{\mathcal{L}(\tilde{x}^k, \tilde{\lambda}^k)\}$ converges to $\mathcal{L}(x^*, \lambda^*)$ where the sequence $\{(\tilde{x}^k, \tilde{\lambda}^k)\}$ is generated by Algorithm 2. We first prove two lemmas.

Lemma 4.2. *Let $\{(\tilde{x}^k, \tilde{\lambda}^k)\}$ be generated by Algorithm 2. Then we have*

$$\begin{aligned}&(1 - \theta_k)(\mathcal{L}(x^*, \lambda^*) - \mathcal{L}(\tilde{x}^{k-1}, \tilde{\lambda}^{k-1})) - (\mathcal{L}(x^*, \lambda^*) - \mathcal{L}(\tilde{x}^k, \tilde{\lambda}^k))\\ &\quad\ge \frac{\theta_k^2}{2\beta_k}\{\|v^k - \lambda^*\|^2 - \|v^{k-1} - \lambda^*\|^2\}.\end{aligned} \tag{4.7}$$

Proof. Using Lemma 4.1 for k, setting $(x(\lambda), \lambda) = (\tilde{x}^{k-1}, \tilde{\lambda}^{k-1})$ and $(x(\lambda), \lambda) = (x^*, \lambda^*)$, we get

$$\mathcal{L}(\tilde{x}^k, \tilde{\lambda}^k) - \mathcal{L}(\tilde{x}^{k-1}, \tilde{\lambda}^{k-1}) \ge -\frac{1}{2\beta_k}\|\tilde{\lambda}^k - \lambda^{k-1}\|^2 + \frac{1}{\beta_k}(\tilde{\lambda}^k - \tilde{\lambda}^{k-1})^\top(\tilde{\lambda}^k - \lambda^{k-1}) \tag{4.8}$$

and

$$\mathcal{L}(\tilde{x}^k, \tilde{\lambda}^k) - \mathcal{L}(x^*, \lambda^*) \ge -\frac{1}{2\beta_k}\|\tilde{\lambda}^k - \lambda^{k-1}\|^2 + \frac{1}{\beta_k}(\tilde{\lambda}^k - \lambda^*)^\top(\tilde{\lambda}^k - \lambda^{k-1}), \tag{4.9}$$

respectively. Multiplying (4.8) by $(1-\theta_k)$ and (4.9) by θ_k and by a simple manipulation, we obtain

(4.10)
$$\begin{aligned}
&(1-\theta_k)(\mathcal{L}(x^*,\lambda^*)-\mathcal{L}(\tilde{x}^{k-1},\tilde{\lambda}^{k-1}))-(\mathcal{L}(x^*,\lambda^*)-\mathcal{L}(\tilde{x}^k,\tilde{\lambda}^k))\\
&\quad=(1-\theta_k)(\mathcal{L}(\tilde{x}^k,\tilde{\lambda}^k)-\mathcal{L}(\tilde{x}^{k-1},\tilde{\lambda}^{k-1}))+\theta_k(\mathcal{L}(\tilde{x}^k,\tilde{\lambda}^k)-\mathcal{L}(x^*,\lambda^*))\\
&\quad\geq-\frac{1}{2\beta_k}\|\tilde{\lambda}^k-\lambda^{k-1}\|^2+\frac{1}{\beta_k}(\tilde{\lambda}^k-\lambda^{k-1})^\top[(1-\theta_k)(\tilde{\lambda}^k-\tilde{\lambda}^{k-1})+\theta_k(\tilde{\lambda}^k-\lambda^*)]\\
&\quad=\frac{1}{2\beta_k}\{\|\tilde{\lambda}^k-(1-\theta_k)\tilde{\lambda}^{k-1}-\theta_k\lambda^*\|^2-\|\lambda^{k-1}-(1-\theta_k)\tilde{\lambda}^{k-1}-\theta_k\lambda^*\|^2\}.
\end{aligned}$$

It follows from (4.3) that

$$\theta_k v^k=\tilde{\lambda}^k-(1-\theta_k)\tilde{\lambda}^{k-1}.$$

Then, rewriting (4.4) as

$$\theta_k v^{k-1}=\lambda^{k-1}-(1-\theta_k)\tilde{\lambda}^{k-1}$$

and substituting the above two equalities into (4.10), we prove the assertion (4.7). □

For the convenience of further analysis, we use the notation

$$\xi_k:=\mathcal{L}(x^*,\lambda^*)-\mathcal{L}(\tilde{x}^k,\tilde{\lambda}^k). \tag{4.11}$$

Lemma 4.3. *Let $\{(\tilde{x}^k,\tilde{\lambda}^k)\}$ be generated by Algorithm 2. Then we have*

$$2\Big(\frac{\beta_{k-1}}{\theta_{k-1}^2}\xi_{k-1}-\frac{\beta_k}{\theta_k^2}\xi_k\Big)\geq\|v^k-\lambda^*\|^2-\|v^{k-1}-\lambda^*\|^2, \tag{4.12}$$

where ξ_k is defined in (4.11).

Proof. First, dividing inequality (4.7) by θ_k^2 and using (4.1), we have

$$\begin{aligned}
&\frac{1}{\theta_{k-1}^2}(\mathcal{L}(x^*,\lambda^*)-\mathcal{L}(\tilde{x}^{k-1},\tilde{\lambda}^{k-1}))-\frac{1}{\theta_k^2}(\mathcal{L}(x^*,\lambda^*)-\mathcal{L}(\tilde{x}^k,\tilde{\lambda}^k))\\
&\qquad\geq\frac{1}{2\beta_k}\{\|v^k-v^*\|^2-\|v^{k-1}-v^*\|^2\}.
\end{aligned}$$

Note that $\{\beta_k\}$ is a nonincreasing sequence. Then, using the notation ξ_k in (4.11), we prove the assertion (4.12). □

Now, we are able to find a bound for $\mathcal{L}(x^*,\lambda^*)-\mathcal{L}(\tilde{x}^k,\tilde{\lambda}^k)$, and this bound only depends on a constant and θ_k. This result enables us to choose specific θ_k and thus to establish a worst-case $O(1/k^2)$ convergence rate for Algorithm 2.

Theorem 4.1. *Let $\{(\tilde{x}^k,\tilde{\lambda}^k)\}$ be generated by Algorithm 2. Then we have*

$$0\leq\mathcal{L}(x^*,\lambda^*)-\mathcal{L}(\tilde{x}^k,\tilde{\lambda}^k)\leq\frac{\theta_k^2\|\lambda^0-\lambda^*\|^2}{2\beta_{\min}}, \tag{4.13}$$

where (x^*,λ^*) is a saddle point of $\mathcal{L}(x,\lambda)$.

Proof. The first inequality is trivial. Now, we prove the second inequality. We first define

$$a_k := \frac{2\beta_k\xi_k}{\theta_k^2} \text{ and } b_k := \|v^k - \lambda^*\|^2.$$

Applying Lemma 4.1 to the case where $(x(\lambda), \lambda) = (x^*, \lambda^*)$ with $k = 1$, we get

$$2\beta_1(\mathcal{L}(\tilde{x}^1, \tilde{\lambda}^1) - \mathcal{L}(x^*, \lambda^*)) \geq \|\tilde{\lambda}^1 - \lambda^*\|^2 - \|\lambda^0 - \lambda^*\|^2.$$

Recall $\theta_1 = 1$, $a_1 = 2\beta_1\xi_1$, $b_1 = \|v^1 - \lambda^*\|^2$ and $v^1 = \tilde{\lambda}^1$. Thus, it follows from the above inequality that

$$a_1 + b_1 \leq \|\lambda^0 - \lambda^*\|^2.$$

It follows from Lemma 2.2 with $c := \|\lambda^0 - \lambda^*\|^2$ and Lemma 4.3 that

$$\frac{2\beta_k\xi_k}{\theta_k^2} \leq \|\lambda^0 - \lambda^*\|^2.$$

Using the definition of ξ_k in (4.11) and $\beta_k \geq \beta_{\min}$, the assertion (4.13) is proved. □

4.2. **Specification of θ_k.** According to Theorem 4.1, a worst-case $O(1/k^2)$ convergence rate of the proposed Algorithm 2 can be ensured as long as $\theta_k \sim O(1/k)$. In this subsection, we specify some choices of the sequence $\{\theta_k\}$ for Algorithm 2. Some concrete accelerated inexact Uzawa methods with worst-case $O(1/k^2)$ convergence rates are thus derived for (1.1). As in [43], we can choose θ_k as follows.

1) Let $\{t_k\}$ be a sequence with positive real numbers defined as

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2} \text{ with } t_0 = 1, \tag{4.14}$$

and choose

$$\theta_k = \frac{1}{t_k}. \tag{4.15}$$

This acceleration scheme was originally proposed in [35] and then was used by many others (e.g., [6]).

2) Let $M \geq 2$ be a constant. Then we can choose θ_k as

$$\theta_k = M/(k + M - 1). \tag{4.16}$$

For both the choices (4.14)–(4.15) and (4.16), we have the following assertion by using mathematical induction and elementary calculations. We thus omit its proof.

Lemma 4.4. *The sequence $\{\theta_k\}$ defined in* (4.14)–(4.15) *or* (4.16) *satisfies*

$$\theta_k \leq \frac{2}{k+1}, \quad \forall\, k \geq 1. \tag{4.17}$$

With the specific choices (4.14)–(4.15) and (4.16), Algorithm 2 can be specified as two concrete algorithms for solving (1.1). In the following subsections, we list the details of Algorithm 2 with the choices (4.14)–(4.15) and (4.16); and specify their convergence rate results in Theorems 4.2 and 4.3, respectively.

4.3. **Algorithm 2 with (4.14)–(4.15).** With the choice (4.14)–(4.15), substituting (4.3) into (4.4) in Algorithm 2, we derive that

$$\begin{aligned}\lambda^k &= (1-\theta_{k+1})\tilde{\lambda}^k + \theta_{k+1}v^k \\ &= (1-\theta_{k+1}+\frac{\theta_{k+1}}{\theta_k})\tilde{\lambda}^k + (\theta_{k+1}-\frac{\theta_{k+1}}{\theta_k})\tilde{\lambda}^{k-1} \\ &= \tilde{\lambda}^k + (\frac{\theta_{k+1}}{\theta_k}-\theta_{k+1})(\tilde{\lambda}^k-\tilde{\lambda}^{k-1}) \\ &= \tilde{\lambda}^k + \frac{t_k-1}{t_{k+1}}(\tilde{\lambda}^k-\tilde{\lambda}^{k-1}),\end{aligned}$$

where the last equality is because $\theta_k = \frac{1}{t_k}$ as defined in (4.15). Thus, Algorithm 2 with (4.15) can be summarized in Algorithm 2A.

Algorithm 2A: An accelerated inexact Uzawa method with (4.14)–(4.15)

Step 0. Take $\lambda^0 = \tilde{\lambda}^0 \in \Lambda$, set $x^0 = \arg\min_{x\in\mathcal{X}} \mathcal{L}(x,\lambda^0)$ and choose $\beta_0 > 0$. Let θ_k be chosen as in (4.14)–(4.15).

Step k. Generate the new iterate (λ^k, x^k) by the following steps.

- Find $(\tilde{\lambda}^k, \tilde{x}^k, \beta_k)$ such that

$$\begin{aligned}\tilde{\lambda}^k &= P_\Lambda[\lambda^{k-1} + \beta_k(Ax^{k-1}-b)], \\ \tilde{x}^k &= \arg\min_{x\in\mathcal{X}} \mathcal{L}(x,\tilde{\lambda}^k),\end{aligned}$$

  and the following requirement is satisfied:

$$\beta_k|(\lambda^{k-1}-\tilde{\lambda}^k)^\top A(x^{k-1}-\tilde{x}^k)| \le \frac{1}{2}\|\lambda^{k-1}-\tilde{\lambda}^k\|^2.$$

- Set $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$.
- Compute

$$\begin{aligned}\lambda^k &= \tilde{\lambda}^k + \Big(\frac{t_k-1}{t_{k+1}}\Big)(\tilde{\lambda}^k-\tilde{\lambda}^{k-1}), \\ x^k &= \arg\min_{x\in\mathcal{X}} \mathcal{L}(x,\lambda^k).\end{aligned}$$

Combining Theorem 4.1 and Lemma 4.4, we immediately show a worst-case $O(1/k^2)$ convergence rate of Algorithm 2A in the following theorem and the proof is omitted.

Theorem 4.2. *Let $\{(\tilde{x}^k,\tilde{\lambda}^k)\}$ be generated by Algorithm* 2A. *Then we have*

$$0 \le \mathcal{L}(x^*,\lambda^*) - \mathcal{L}(\tilde{x}^k,\tilde{\lambda}^k) \le \frac{2\|\lambda^0-\lambda^*\|^2}{\beta_{\min}(k+1)^2}, \tag{4.18}$$

where (x^*,λ^*) is a saddle point of $\mathcal{L}(x,\lambda)$.

4.4. **Algorithm 2 with (4.16).** Similarly, if θ_k takes the value in (4.16), Algorithm 2 can be specified in Algorithm 2B.

Algorithm 2B: An accelerated inexact Uzawa method with (4.16)
Step 0. Take $\lambda^0 \in \Lambda$, set $\tilde{\lambda}^0 = \lambda^0$ and $x^0 \in \arg\min_{x\in\mathcal{X}} \mathcal{L}(x, \lambda^0)$, and choose $M \geq 2$ and $\beta_0 > 0$. Let θ_k be chosen as in (4.16).
Step k. Generate the new iterate (λ^k, x^k) by the following steps.

- Find $(\tilde{\lambda}^k, \tilde{x}^k, \beta_k)$ such that

$$\tilde{\lambda}^k = P_\Lambda[\lambda^{k-1} + \beta_k(Ax^{k-1} - b)],$$
$$\tilde{x}^k = \arg\min_{x\in\mathcal{X}} \mathcal{L}(x, \tilde{\lambda}^k),$$

  and the following requirement is satisfied:

$$\beta_k|(\lambda^{k-1} - \tilde{\lambda}^k)^\top A(x^{k-1} - \tilde{x}^k)| \leq \frac{1}{2}\|\lambda^{k-1} - \tilde{\lambda}^k\|^2.$$

- Set $v^k = \tilde{\lambda}^{k-1} + \frac{k+M-1}{M}(\tilde{\lambda}^k - \tilde{\lambda}^{k-1})$.
- Compute

$$\lambda^k = \frac{k}{k+M}\tilde{\lambda}^k + \frac{M}{k+M}v^k,$$
$$x^k = \arg\min_{x\in\mathcal{X}} \mathcal{L}(x, \lambda^k).$$

Using Theorem 4.1 and the definition of θ_k in (4.16), we can show a worst-case $O(1/k^2)$ convergence rate of Algorithm 2B easily. The proof is thus omitted.

Theorem 4.3. *Let $\{(\tilde{x}^k, \tilde{\lambda}^k)\}$ be generated by Algorithm* 2B*. Then we have*

$$0 \leq \mathcal{L}(x^*, \lambda^*) - \mathcal{L}(\tilde{x}^k, \tilde{\lambda}^k) \leq \frac{M^2\|\lambda^0 - \lambda^*\|^2}{2\beta_{\min}(k+M-1)^2},$$

where (x^*, λ^*) is a saddle point of $\mathcal{L}(x, \lambda)$.

5. Another algorithmic framework of accelerated inexact Uzawa methods

In this section, we show that Algorithm 1 can be accelerated by another scheme in [43] and thus some other specific accelerated inexact Uzawa methods can be easily obtained. The algorithmic framework can be summarized in the following Algorithm 3.

As for Algorithm 2, we can specify Algorithm 3 by choosing the parameter θ_k according to (4.14)–(4.15) or (4.16). Some specific accelerated inexact Uzawa methods can be derived based on Algorithm 3. Parallel to Section 4.1, we can easily prove some similar worst-case $O(1/k^2)$ convergence rate results for the resulting accelerated inexact Uzawa methods based on Algorithm 3. For succinctness, we skip the details.

Algorithm 3: Another algorithmic framework of accelerated inexact Uzawa methods

Step 0. Take $\lambda^0 \in \Lambda$, choose $z^0 = \tilde{\lambda}^0 = \lambda^0$, set $x^0 = \arg\min_{x\in\mathcal{X}} \mathcal{L}(x, \lambda^0)$, and choose $\beta_0 > 0$. Let $\{\theta_k\}$ be a sequence satisfying (4.1).

Step k. Generate the new iterate (λ^k, x^k) via:

- Find $(\tilde{\lambda}^k, z^k, x^k)$ and β_k such that

$$\tilde{\lambda}^k = P_\Lambda[\tilde{\lambda}^{k-1} + \frac{\beta_k}{\theta_k}(Ax^{k-1} - b)], \tag{5.1a}$$

$$z^k = (1-\theta_k)z^{k-1} + \theta_k\tilde{\lambda}^k, \tag{5.1b}$$

$$\tilde{x}^k = \arg\min_{x\in\mathcal{X}} \mathcal{L}(x, z^k), \tag{5.1c}$$

and the following requirement is satisfied:

$$\beta_k|(z^k - \lambda^{k-1})^\top A(\tilde{x}^k - x^{k-1})| \le \frac{1}{2}\|z^k - \lambda^{k-1}\|^2. \tag{5.1d}$$

- Set

$$\lambda^k = (1-\theta_{k+1})z^k + \theta_{k+1}\tilde{\lambda}^k, \tag{5.2}$$

$$x^k = \arg\min_{x\in\mathcal{X}} \mathcal{L}(x, \lambda^k). \tag{5.3}$$

6. Numerical results

In this section, we report some numerical results to show the efficiency of the proposed algorithms. We will verify: 1) the efficiency of Algorithm 1; 2) the acceleration effectiveness of the proposed accelerated Uzawa algorithms; and 3) the proposed accelerated Uzawa methods are competitive with some efficient solvers in the literature. For succinctness, we only report the numerical results of the accelerated Uzawa methods based on Algorithm 2 and omit those based on Algorithm 3.

The proposed algorithms were coded by MATLAB v7.4 (R2007a) and all our experiments were performed on a desktop with Windows XP system and a Pentium (R) 4 CPU processor (2.80GH) with a 2GB memory.

6.1. **Implementation details.** We first specify how to choose the step size β_k for the proposed algorithms. For succinctness, we only elaborate on the implementation of Algorithm 1, and the details of implementing Algorithms 2A and 2B are similar.

Remark 6.1. With the implementation detail of Algorithm 1 given below, we see that

$$\beta_k > \beta_{\min} := \frac{1}{2L(G)} * \min\{1, \frac{1}{2\beta_0 L(G)}\} * 0.45.$$

Recall that the criterion (3.1c) is guaranteed to be satisfied when β_k is reduced finitely many times. In the implementation, we can choose a constant value for β_k after the iteration counter k is sufficiently large. In our numerical experiments, we found that the "While-Do" inner loop is usually terminated after one step for the tested scenarios. Note that our adjustment rule for choosing the step size β_k also guarantees that β_k is not too small. Thus, the sequence $\{\beta_k\}$ generated by Algorithm 1 with the implementation detail above is both lower and upper bounded. In addition, the computational load of determining appropriate step

Implementation of Algorithm 1.
Initialization. Take $\lambda^0 \in \Lambda$. Set $x^0 = \arg\min_{x\in\mathcal{X}} \mathcal{L}(x, \lambda^0)$ and $\beta_0 > 0$.
Step k. Compute:

$$\tilde{\lambda}^k = P_\Lambda[\lambda^{k-1} + \beta_k(Ax^{k-1} - b)],$$
$$\tilde{x}^k = \arg\min_{x\in\mathcal{X}} \mathcal{L}(x, \tilde{\lambda}^k),$$
$$r_k := \frac{\beta_k|(\lambda^{k-1} - \tilde{\lambda}^k)^\top A(x^{k-1} - \tilde{x}^k)|}{\|\lambda^{k-1} - \tilde{\lambda}^k\|^2},$$
$$\beta_k^{(0)} = \beta_k, \quad i = 0.$$

While "$r_k > 1/2$" **Do:**

$$i := i + 1,$$
$$\beta_k^{(i)} = \beta_k^{(i-1)} * 0.45 * \min\{1, \frac{1}{r_k}\},$$
$$\tilde{\lambda}^k = P_\Lambda[\lambda^{k-1} + \beta_k^{(i)} * (Ax^{k-1} - b)],$$
$$\tilde{x}^k = \arg\min_{x\in\mathcal{X}} \mathcal{L}(x, \tilde{\lambda}^k),$$
$$r_k := \frac{\beta_k^{(i)}|(\lambda^{k-1} - \tilde{\lambda}^k)^\top A(x^{k-1} - \tilde{x}^k)|}{\|\lambda^{k-1} - \tilde{\lambda}^k\|^2},$$
$$\text{if } r_k < 0.3, \ \beta = \beta * 1.5.$$

End Do
Set $\beta_k = \beta_k^{(i)}$, $x^k = \tilde{x}^k$ and $\lambda^k = \tilde{\lambda}^k$.

sizes for Algorithm 1 in the absence of σ and $\|A^\top A\|$ is not too much. In Section 6.2.1, we will compare this strategy with some classical Armijo-like rules in the literature.

6.2. **An image deblurring model with a box constraint.** In this subsection, we apply the proposed algorithms to solve the following constrained linear least-squares problem:

$$\min_{l\le x\le u} \{\frac{1}{2}\|Kx - c\|^2 + \frac{\mu}{2}\|Dx\|^2\}, \tag{6.1}$$

where $K, D \in \Re^{n\times n}$, $c \in \Re^n$, $l, u \in \Re^n$, $\mu > 0$, and $\|\cdot\|$ denotes the 2-norm. The box constraint $l \le x \le u$ is interpreted entry-wise, i.e., $l_i \le x_i \le u_i$ for any $i \in \{1, 2, \ldots, n\}$. This model captures the application of an image deblurring problem with box constraints, where x is the vector representation of a digital image to be restored, K is a blurring operator (integral operator), D is a regularization operator (differential operator), c is the vector representation of a blurred image, and l and u are respectively the lower and upper bounds of pixel values. As in the literature, we assume that $\mathcal{N}(K) \cap \mathcal{N}(D) = \{\mathbf{0}\}$, where $\mathcal{N}(K)$ and $\mathcal{N}(D)$ are the null spaces of K and D, respectively. With this assumption, the objective function in (6.1) is strongly convex; see, e.g., [9]. We refer to [20] for the effectiveness of considering the box constraint $l \le x \le u$ on pixel values for debluring a blurred image.

It is easy to see that (6.1) can be reformulated as

$$\begin{aligned} &\min \frac{1}{2}\|Kx-c\|^2+\frac{\mu}{2}\|Dx\|^2,\\ &s.t.\left(\begin{array}{c} -I\\ I\end{array}\right)x+\left(\begin{array}{c} l\\ -u\end{array}\right)\le 0, \end{aligned} \tag{6.2}$$

which is a special case of the model (1.1). Let the Lagrangian function of (6.2) be

$$\mathcal{L}(x,\lambda_1,\lambda_2)=\frac{1}{2}\|Kx-c\|^2+\frac{\mu}{2}\|Dx\|^2+\lambda_1^\top(-x+l)+\lambda_2^\top(x-u),$$

with λ_1 and λ_2 the Lagrange multipliers associated with the first and second inequality constraint in (6.2), respectively.

Now, let us elucidate the subproblems when the proposed algorithms are implemented to solve (6.2). For succinctness, we only analyze the Uzawa step at each iteration of the proposed algorithms and ignore the acceleration steps. More specifically, to implement Algorithm 2 for (6.2), the main step (4.2a)–(4.2b) of each iteration reduces to

$$\left\{\begin{array}{l} \tilde{\lambda}_1^k=P_{R_+^n}(\lambda_1^{k-1}+\beta_k(-x^{k-1}+l)),\\ \tilde{\lambda}_2^k=P_{R_+^n}(\lambda_2^{k-1}+\beta_k(x^{k-1}-u)),\\ (K^\top K+\mu D^\top D)\tilde{x}^k=K^\top c+\tilde{\lambda}_1^k-\tilde{\lambda}_2^k. \end{array}\right. \tag{6.3}$$

Thus, the resulting subproblems at each iteration are easy to solve.

Now, we specify the stopping criterion to implement the proposed algorithms. Let $(x^*,\lambda_1^*,\lambda_2^*)$ be a saddle point of $L(x,\lambda_1,\lambda_2)$. Then, solving (6.2) is equivalent to

$$\left\{\begin{array}{l} K^\top(Kx^*-c)+\mu D^\top Dx^*-\lambda_1^*+\lambda_2^*=0,\\ 0\le\lambda_1^*\perp x^*-l\ge 0,\\ 0\le\lambda_2^*\perp u-x^*\ge 0. \end{array}\right. \tag{6.4}$$

In other words, we can measure the accuracy of an iterate $(x^k,\lambda_1^k,\lambda_2^k)$ to a saddle point of $L(x,\lambda_1,\lambda_2)$ by the violation of the conditions in (6.4). Because of (6.3) and (6.4), the accuracy of an iterate $(x^k,\lambda_1^k,\lambda_2^k)$ to a saddle point of $L(x,\lambda_1,\lambda_2)$ can be measured by the quantity of `infeasibility` defined as:

$$\texttt{infeasibility}=\max\{|(\lambda_1^k)^\top(-x^k+l)|,|(\lambda_2^k)^\top(x^k-u)|\}. \tag{6.5}$$

We thus use

$$\texttt{infeasibility}\le 10^{-2} \tag{6.6}$$

as the stopping criterion when implementing the proposed algorithms for (6.2).

We test two 256×256 images: satellite.pgm and chart.tiff, as shown in Figure 1. Accordingly, $n=65,536$ in the model (6.1) for these images. As in [21, 33], the blurring matrix K is chosen to be the out-of-focus blur and the matrix D is taken to be the gradient matrix. Under the periodic boundary condition for x, both $D^\top D$ and $K^\top K$ are block circulant matrices with circulant blocks. Thus they are diagonalizable by the 2D discrete Fourier transforms (see, e.g., [19]), and hence the third subproblem in (6.3) can be solved with an order of

$O(n \log n)$ operations. The observed image c is expressed as $c = K\bar{x} + \eta\mathbf{r}$, where $\bar{x}$ is the true image, $\mathbf{r}$ is a random vector with entries distributed in the standard normal, and η is the level of Gaussian noise. The box constraint is set as $l_i = 0$ and $u_i = 255$ for all $i = 1, \ldots, n$. We employ the MATLAB scripts: `K = fspecial('average',alpha)` and `C = imfilter(X,K,'circular','conv') + η*randn(m,n)` to produce the blurred images corrupted by the *average* kernel of different sizes. Here, `alpha` is the kernel size of the blurring operator, `X` denotes the original image, and `C` represents the observed image. The quality of a restored image (denoted by x) is measured by the peak signal-to-noise ratio (PSNR) in decible (dB):

$$\text{PSNR}(x) = 20\log_{10}\frac{x_{\max}}{\sqrt{\text{Var}(x,\bar{x})}} \quad \text{with} \quad \text{Var}(x,\bar{x}) = \frac{\sum_{j=1}^{n^2}[\bar{x}(j) - x(j)]^2}{n^2}. \tag{6.7}$$

Here, $\bar{x}$ is the true image and $\bar{x}_{\max}$ is the maximum possible pixel value of the image. On the other hand, we measure the accuracy of the recovered quality by the relative error defined as

$$\texttt{relerr} := \|x^k - \bar{x}\|_F / \|\bar{x}\|_F. \tag{6.8}$$

In all the experiments, we set $\mu = 0.01$ in model (6.1) as suggested by the numerical results in [21, 33].

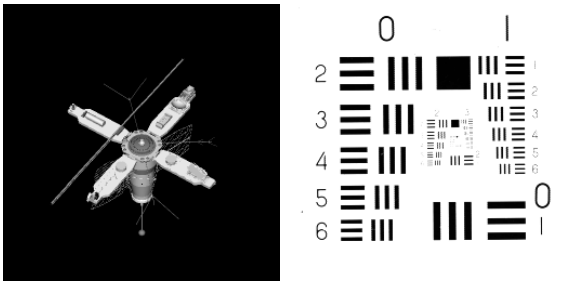

FIGURE 1. Original images: satellite.pgm (left) and chart.tiff (right)

6.2.1. *Verifying the efficiency of Algorithm* 1. For Algorithm 1, we can also use some standard Armijo-like line-search rules in the literature to choose the step size in (3.1a). In this subsection, we verify the efficiency of Algorithm 1 by comparing it with its variant which uses an Armijo-like line-search rule, rather than the criterion (3.1c), to determine the step size β_k in (3.1a). Particularly, we compare the one proposed in [7]:

$$\begin{cases} \beta_k = \kappa^{m_k}\beta_0, \\ -G(\lambda^k) \le -G(\lambda^{k-1}) - \mu(\lambda^k - \lambda^{k-1})^\top \nabla G(\lambda^{k-1}), \end{cases} \tag{6.9}$$

where κ and μ in $(0, 1)$ are given, β_0 is the initial step size and m_k is the smallest nonnegative integer such that the second inequality is satisfied. In the following, we will show the difference when (3.1c) and (6.9) are used in (3.1) for Algorithm 1, respectively.

TABLE 1. Numerical comparison

alpha	Algorithm	Iter.	PSNR (dB)	infeasiblity	Time (s)	relerr	Total_adj
satellite, $\eta = 3$							
11	Algo1	98	25.89	9.97e-003	3.62	2.44e-001	25
	DPJ_Armijo	133	25.89	9.92e-003	6.79	2.44e-001	26
15	Algo1	115	24.76	9.99e-003	4.85	2.78e-001	36
	DPJ_Armijo	152	24.76	1.00e-002	9.03	2.78e-001	32
19	Algo1	75	24.06	9.93e-003	3.31	3.01e-001	16
	DPJ_Armijo	89	24.06	9.89e-003	5.65	3.01e-001	32
23	Algo1	110	23.65	9.97e-003	4.12	3.15e-001	28
	DPJ_Armijo	142	23.65	9.98e-003	6.47	3.15e-001	39
chart, $\eta = 3$							
11	Algo1	112	19.74	9.98e-003	5.37	1.10e-001	26
	DPJ_Armijo	145	19.74	9.87e-003	6.32	1.10e-001	26
15	Algo1	136	18.37	9.95e-003	6.01	1.29e-001	32
	DPJ_Armijo	173	18.37	9.95e-003	10.16	1.29e-001	32
19	Algo1	90	17.65	9.92e-003	3.28	1.40e-001	25
	DPJ_Armijo	105	17.65	9.99e-003	7.41	1.40e-001	32
23	Algo1	154	16.78	9.98e-003	5.49	1.55e-001	42
	DPJ_Armijo	200	16.78	9.99e-003	10.61	1.55e-001	39

The initial values are set as $\beta_0 = 0.1$ and $\lambda^0 = \mathbf{0}$ for the comparison in this subsection. For the Armijo rule (6.9), we choose $\kappa = 0.9$ and $\mu = 1/2$ (some other choices of these two parameters lead to similar performance; we thus only report the results with this choice for succinctness). We test different cases of (6.2) where the kernel size `alpha` is 11, 15, 19 and 23, respectively. The Gaussian noise level is fixed as $\eta = 3$. As mentioned in Remark 3.1, if the criterion (3.1c) is replaced by an Armijo rule, we can regard the resulting scheme as an application of the project gradient method to the dual of (1.1). We thus denote by "DPJ_Armijo" the resulting scheme by combining the Uzawa step (3.1) with the Armijo-like line-search rule (6.9). Algorithm 1 is further abbreviated as "Algo1" in the following. In Table 1, we report the numerical results when they are implemented to solve these different cases of (6.2). In this table, the iteration number ("Iter."), computing time in seconds ("Time (s)"), PSNR values of the recovered images ("PSNR (dB)"), quantity of `infeasibility` defined in (6.5), `relerr` defined in (6.8) and total numbers of adjusting the step size β_k ("Total_adj") are reported when the stopping criterion (6.6) is satisfied. Data in this table clearly show the efficiency of the proposed criterion (3.1c).

To verify Remark 6.1, we focus on the case where the kernel size is `alpha` $= 11$ and the Gaussian noise level is $\eta = 3$ for the images in Figure 1; and plot the respective numbers of adjustment of β required by the practical version of Algorithm 1 and the dual projected gradient method at each iteration in Figure 2. We observe that at each iteration of Algorithm 1, the number of adjusting the step size β_k is at most one; while for the rule (6.9), it adjusts many more times at the beginning of the iteration although it becomes stable later.

6.2.2. *Verifying the acceleration effectiveness of Algorithm* 2. We then verify the acceleration effectiveness of the proposed Algorithms 2A and 2B over Algorithm 1. The initial values are set as $\beta_0 = 0.1$ and $\lambda^0_{1,2} = \mathbf{0}$ for all the algorithms to be tested. For Algorithm 2B, we take $M = 2$.

We test different cases of (6.2) where the kernel size `alpha` is $11, 15, 19, 23$, respectively; and the Gaussian noise level is fixed as $\eta = 3$. In Table 2, we report the numerical results when all these algorithms are implemented to solve these different cases of (6.2). As before, we report "Iter.", "Time (s)", "PSNR (dB)",

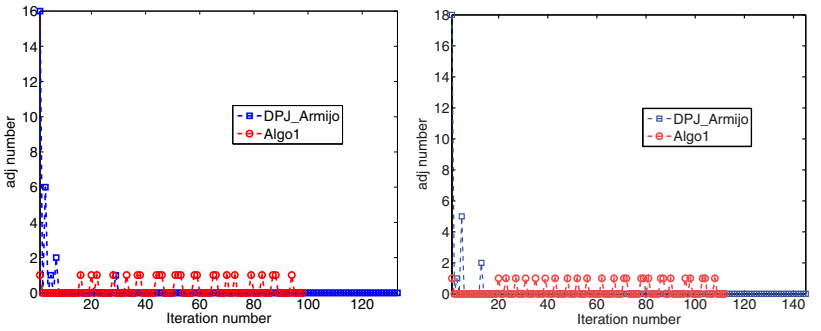

FIGURE 2. Numbers of adjustment of β: satellite.pgm (left) and chart.jpg (right)

TABLE 2. Numerical comparison

alpha	Algorithm	Iter.	PSNR (dB)	infeasiblity	Time (s)	relerr
satellite, $\eta = 3$						
11	Algo1	98	25.89	9.97e-003	4.60	2.44e-001
	Algo2A	33	25.91	9.49e-003	2.37	2.43e-001
	Algo2B	29	25.89	9.51e-003	1.65	2.44e-001
15	Algo1	115	24.76	9.99e-003	4.46	2.78e-001
	Algo2A	33	24.77	9.69e-003	1.58	2.77e-001
	Algo2B	33	24.76	9.82e-003	1.79	2.77e-001
19	Algo1	75	24.06	9.93e-003	3.65	3.01e-001
	Algo2A	22	24.07	9.93e-003	2.22	3.01e-001
	Algo2B	25	24.06	9.62e-003	2.53	3.01e-001
23	Algo1	110	23.65	9.97e-003	4.17	3.15e-001
	Algo2A	28	23.65	9.97e-003	1.95	3.15e-001
	Algo2B	30	23.65	9.98e-003	1.51	3.15e-001
chart, $\eta = 3$						
11	Algo1	112	19.74	9.98e-003	4.49	1.10e-001
	Algo2A	47	19.74	9.33e-003	3.45	1.10e-001
	Algo2B	34	19.74	9.65e-003	2.50	1.10e-001
15	Algo1	136	18.37	9.95e-003	5.04	1.29e-001
	Algo2A	32	18.38	9.86e-003	1.54	1.29e-001
	Algo2B	36	18.37	9.99e-003	2.01	1.29e-001
19	Algo1	90	17.65	9.92e-003	3.23	1.40e-001
	Algo2A	28	17.66	9.40e-003	1.51	1.40e-001
	Algo2B	28	17.65	9.47e-003	1.67	1.40e-001
23	Algo1	154	16.78	9.98e-003	5.12	1.55e-001
	Algo2A	38	16.78	9.82e-003	2.00	1.55e-001
	Algo2B	36	16.78	9.66e-003	1.67	1.55e-001

`infeasibility` defined in (6.5) and `relerr` defined in (6.8) when the stopping criterion (6.6) is satisfied. Data in this table show that the proposed accelerated Uzawa methods are much faster than Algorithm 1 to achieve the same level of restoration quality (i.e., optimality).

To further demonstrate the comparison of different algorithms, we focus on the satellite.pgm image and the case where the kernel size is `alpha` $= 19$ and the Gaussian noise level is $\eta = 3$. In Figure 3, we plot the evolution of the PSNR value with respect to the iteration number and computing time, respectively; and the evolution of the quantity of `infeasibility` and `relerr` with respect to the iteration number. The recovered images are displayed in Figure 4.

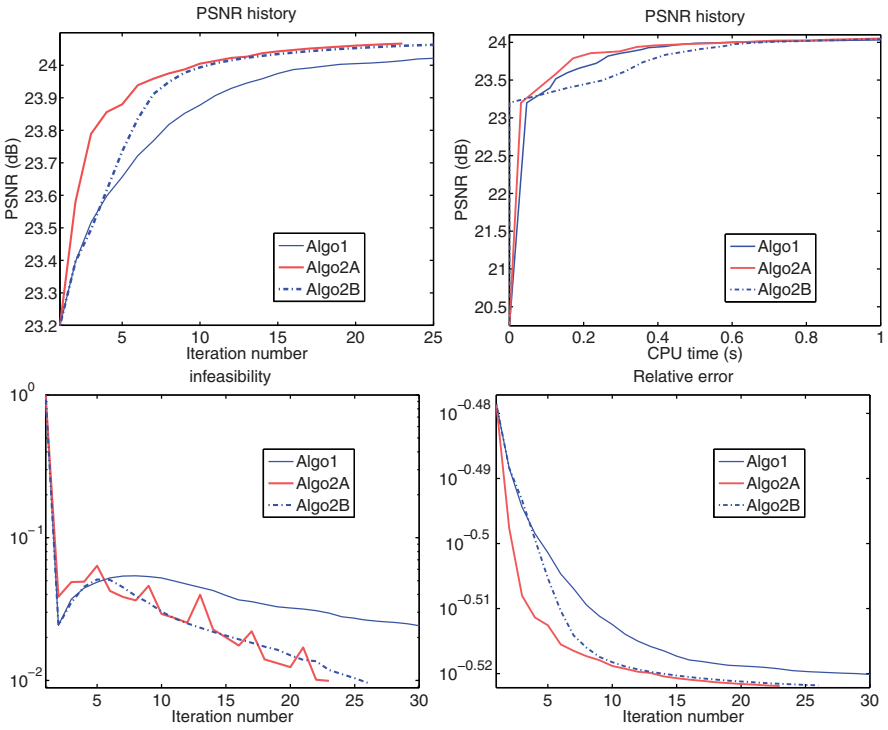

FIGURE 3. Top row: PSNR w.r.t. iteration number (left) and CPU time (right); Bottom row: quantity of `infeasibility` (left) and `relerr` (right) w.r.t. iteration number for satellite.pgm with *average* blur, `alpha` $= 19$, $\eta = 3$.

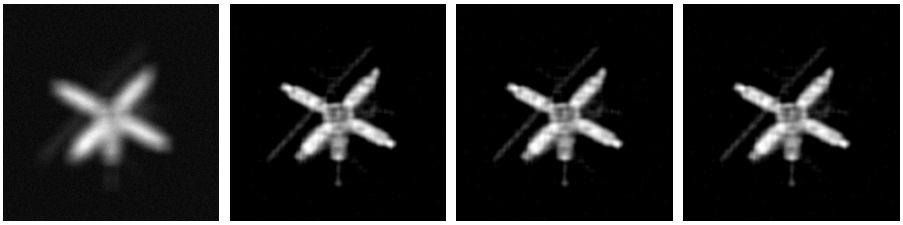

FIGURE 4. Recovered images of satellite.pgm with *average* blur, `alpha` $= 19$, $\eta = 3$. Blurred image (left); recovered image by Algorithm 1 (middle left); recovered image by Algorithm 2A (middle right); recovered image by Algorithm 2B (right).

6.2.3. *Comparison with other existing methods.* In the literature, there are some efficient methods applicable to the model (6.1), including the reduced Newton method in [33] (denoted by RN), affine scaling method with a BB step in [21] (denoted by AS-BB), and ADMM in [24]. In this subsection, we compare the proposed accelerated Uzawa methods with these existing algorithms and further show their efficiency.

TABLE 3. Numerical comparison between the proposed algorithms and some existing ones

satellite, $\eta = 3$										
alpha	Time (s)					PSNR (dB)				
	RN	AS-BB	ADMM	Algo2A	Algo2B	RN	AS-BB	ADMM	Algo2A	Algo2B
15	30.37	15.85	5.05	4.14	4.70	24.74	24.53	24.76	24.77	24.77
19	19.16	15.53	5.83	3.13	3.39	24.09	24.00	24.10	24.10	24.10
23	31.72	17.60	9.23	3.53	3.80	23.67	23.55	23.65	23.65	23.65
chart, $\eta = 3$										
	RN	AS-BB	ADMM	Algo2A	Algo2B	RN	AS-BB	ADMM	Algo2A	Algo2B
15	42.74	16.17	4.00	3.97	4.42	18.39	18.26	18.37	18.40	18.40
19	34.25	16.21	4.06	2.94	3.70	17.65	17.54	17.62	17.65	17.65
23	36.55	17.53	4.75	4.06	4.94	16.78	16.58	16.76	16.80	16.80

To apply ADMM, we need to reformulate (6.1) as

$$\begin{array}{ll} \min & \left\{\frac{1}{2}\|Kx-c\|^2+\frac{\mu}{2}\|Dx\|^2\right\}, \\ s.t. & x-y=0, \\ & x\in\Re^n, y\in\Omega:=[l,u], \end{array} \tag{6.10}$$

where $y \in \Re^n$ is an auxiliary variable. Let the augmented Lagrangian function of (6.10) be

$$\mathcal{L}_\varrho(x,y,p)=\frac{1}{2}\|Kx-c\|^2+\frac{\mu}{2}\|Dx\|^2+p^\top(x-y)+\frac{\varrho}{2}\|x-y\|^2,$$

with $p \in \Re^n$ the Lagrange multiplier and $\varrho > 0$ a penalty parameter. Then, the iterative scheme of ADMM reads as

$$\left\{\begin{array}{l} x^{k+1}=\arg\min L_\varrho(x,y^k,p^k), \\ y^{k+1}=\arg\min L_\varrho(x^{k+1},y,p^k), \\ p^{k+1}=p^k+\varrho(x^{k+1}-y^{k+1}). \end{array}\right.$$

Obviously, the subproblems at each iteration of ADMM for (6.10) all have closed-form solutions:

$$\left\{\begin{array}{l} (K^\top K+\mu D^\top D)x^{k+1}=K^\top c-p^k+\varrho y^k, \\ y^{k+1}=P_\Omega[x^{k+1}+\frac{p^k}{\varrho}], \\ p^{k+1}=p^k+\varrho(x^{k+1}-y^{k+1}). \end{array}\right.$$

To implement ADMM, the penalty parameter ϱ is set as 0.1, the initial iterate y^0 is taken as the blurry image and $p^0 = \mathbf{0}$.

As in Section 6.2, we test the images satellite.pgm (256×256) and chart.tiff (256×256) shown in Figure 1. We test the case where the kernel size of the blur is $\texttt{alpha} = 15, 19, 23$ and the Gaussian noise level is fixed as $\eta = 3$. In Table 3, we list the numerical comparison of these algorithms in terms of "Time (s)" and "PSNR (dB)". Data in this table show that Algorithms 2A and 2B are both faster than the other methods to achieve the same level of restoration. In Figure 5, we plot the evolution of the PSNR value with respect to the computing time in seconds for some scenarios of the tested images.

6.2.4. *Robustness to the initial step size.* In the previous subsections, we have shown the efficiency of the proposed accelerated Uzawa methods. In particular, the numerical results reported in Section 6.2.3 show that Algorithms 2A and 2B are competitive with the ADMM. In this subsection, we will show that the proposed Uzawa methods are very robust to the initial choice of the step size β_0. This represents another advantage of the proposed algorithms over some existing methods

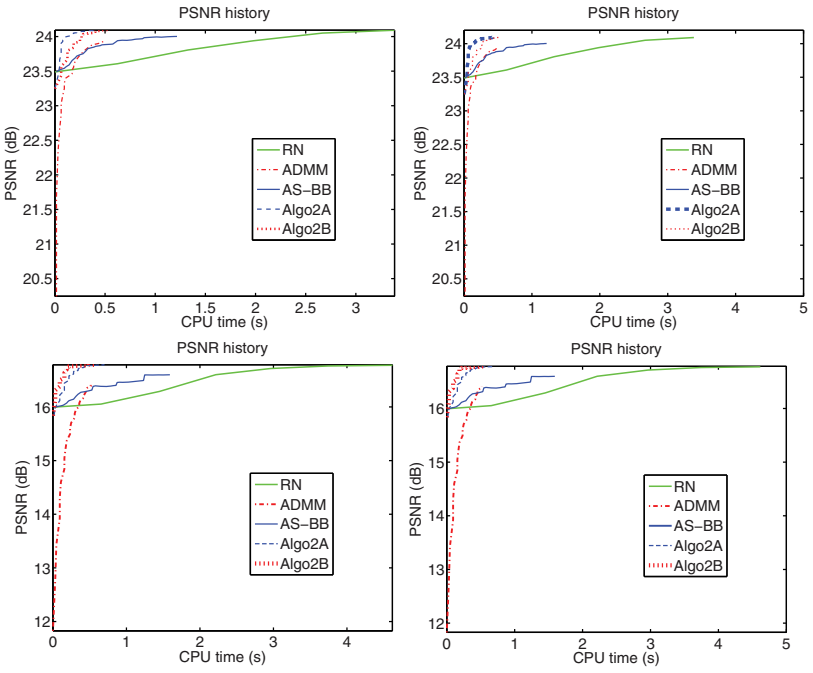

FIGURE 5. First row: satellite.pgm with *average* blur, `alpha` $= 19$, $\eta = 3$; PSNR w.r.t. computing time (left) and zoom-in for the first 5 seconds (right). Second row: chart.tiff with *average* blur, `alpha` $= 23$, $\eta = 3$; PSNR w.r.t. computing time (left) and zoom-in for the first 5 seconds (right).

such as the ADMM whose numerical performance is well known to be sensitive to the initial choice of its penalty parameter ϱ.

We focus on the satellite.pgm image with an *average* blur of kernel size `alpha` $=$ 13 and the Gaussian noise level is $\eta = 3$. To see the sensitivity of the proposed algorithms to β_0, we test two sets of values for β_0: β_0 increasing from 0.1 to 1 with an equal distance of 0.1 and from 1 to 15 with an equal distance of 1. For each choice of β_0, we implement the proposed algorithms and record their recovered PSNR values and computing time in seconds when the stopping criterion

$$\frac{\|x^{k+1} - x^k\|}{\|x^k\|} < 1e-3$$

is satisfied. In Figure 6, we plot the PSNR values of recovered images and the computing time in seconds with respect to different choices of β_0. These plots clearly show that the proposed algorithms are all robust to different choices of the initial value of β_0. This is an important convenience for implementing the proposed algorithms.

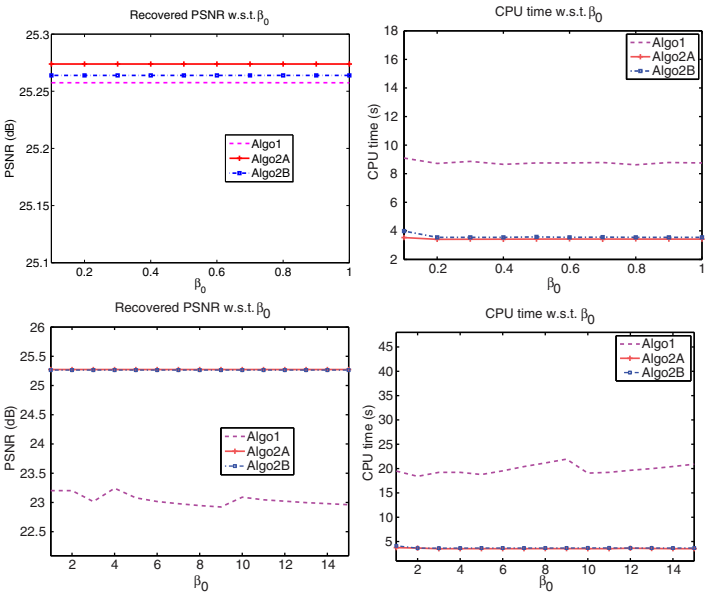

FIGURE 6. Robustness of the proposed algorithms on initial value of β_0. First row: $\beta_0 = \{0.1, 0.2, \ldots, 1\}$. Second row: $\beta_0 = \{1, 2, \ldots, 15\}$.

7. Conclusions and future work

We focus on a linearly constrained strongly convex minimization model, whose objective function is given by a black-box oracle and thus its strong convexity modulus is unknown. Our goal is to discuss a customized and more effective application of the classical Uzawa method to finding a saddle point of the Lagrangian function of the model under consideration. We mainly show that an inexact version of the Uzawa method without knowing the strong convexity modulus of the objective function can be accelerated by some existing acceleration schemes in the literature — the convergence rate measured by the iteration complexity can be improved to $O(1/k^2)$ from $O(1/k)$ where k denotes the iteration counter and the accuracy of an iterate is measured by the residual of the Lagrangian function. This work can be regarded as the combination of a rather new technique (the acceleration scheme in [43]) and the old Uzawa method, and it easily yields a number of implementable accelerated inexact Uzawa methods with $O(1/k^2)$ worst-case convergence rates. We show the efficiency of the accelerated Uzawa methods by an image deblurring problem. Our numerical experiments demonstrate that the proposed accelerated Uzawa methods outperform some popular solvers for the tested examples and thus they have potential applications in some areas where the original Uzawa method is used. Moreover, the proposed accelerated Uzawa methods are robust to the initial choice of the step size; this represents a superiority to some existing methods. For future work, it is interesting to extend our analysis to the generic saddle point problem and obtain accelerated Uzawa schemes with a worst-case $O(1/k^2)$ convergence rate in more general settings.

References

[1] K. J. Arrow, L. Hurwicz, and H. Uzawa, *Studies in Linear and Non-Linear Programming*, With contributions by H. B. Chenery, S. M. Johnson, S. Karlin, T. Marschak, R. M. Solow. Stanford Mathematical Studies in the Social Sciences, vol. II, Stanford University Press, Stanford, Calif., 1958. MR0108399

[2] A. Auslender and M. Teboulle, *Interior gradient and proximal methods for convex and conic optimization*, SIAM J. Optim. **16** (2006), no. 3, 697–725, DOI 10.1137/S1052623403427823. MR2197553

[3] I. Babuška, *The finite element method with Lagrangian multipliers*, Numer. Math. **20** (1972/73), 179–192. MR0359352

[4] C. Bacuta, *A unified approach for Uzawa algorithms*, SIAM J. Numer. Anal. **44** (2006), no. 6, 2633–2649, DOI 10.1137/050630714. MR2272609

[5] A. Beck and M. Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM J. Imaging Sci. **2** (2009), no. 1, 183–202, DOI 10.1137/080716542. MR2486527

[6] A. Beck and M. Teboulle, *Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems*, IEEE Trans. Image Process. **18** (2009), no. 11, 2419–2434, DOI 10.1109/TIP.2009.2028250. MR2722312

[7] D. P. Bertsekas, *On the Goldstein-Levitin-Polyak gradient projection method*, IEEE Trans. Automatic Control **AC-21** (1976), no. 2, 174–184. MR0416017

[8] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation, Numerical Methods*, Prentice-Hall, Englewood Cliffs, NJ, 1989.

[9] Å. Björck, *Numerical Methods for Least Squares Problems*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1996. MR1386889

[10] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, Cambridge, 2004. MR2061575

[11] J. H. Bramble, *The Lagrange multiplier method for Dirichlet's problem*, Math. Comp. **37** (1981), no. 155, 1–11, DOI 10.2307/2007496. MR616356

[12] J. H. Bramble, J. E. Pasciak, and A. T. Vassilev, *Analysis of the inexact Uzawa algorithm for saddle point problems*, SIAM J. Numer. Anal. **34** (1997), no. 3, 1072–1092, DOI 10.1137/S0036142994273343. MR1451114

[13] F. Brezzi and M. Fortin, *Mixed and Hybrid Finite Element Methods*, Springer Series in Computational Mathematics, vol. 15, Springer-Verlag, New York, 1991. MR1115205

[14] J.-F. Cai, E. J. Candès, and Z. Shen, *A singular value thresholding algorithm for matrix completion*, SIAM J. Optim. **20** (2010), no. 4, 1956–1982, DOI 10.1137/080738970. MR2600248

[15] J.-F. Cai, S. Osher, and Z. Shen, *Linearized Bregman iterations for compressed sensing*, Math. Comp. **78** (2009), no. 267, 1515–1536, DOI 10.1090/S0025-5718-08-02189-3. MR2501061

[16] J.-F. Cai, S. Osher, and Z. Shen, *Linearized Bregman iterations for frame-based image deblurring*, SIAM J. Imaging Sci. **2** (2009), no. 1, 226–252, DOI 10.1137/080733371. MR2486529

[17] A. Chambolle, *An algorithm for total variation minimization and applications*, J. Math. Imaging Vision **20** (2004), no. 1-2, 89–97, DOI 10.1023/B:JMIV.0000011320.81911.38. Special issue on mathematics and image analysis. MR2049783

[18] A. Chambolle and T. Pock, *A first-order primal-dual algorithm for convex problems with applications to imaging*, J. Math. Imaging Vision **40** (2011), no. 1, 120–145, DOI 10.1007/s10851-010-0251-1. MR2782122

[19] R. H. Chan and M. K. Ng, *Conjugate gradient methods for Toeplitz systems*, SIAM Rev. **38** (1996), no. 3, 427–482, DOI 10.1137/S0036144594276474. MR1409592

[20] R. H. Chan, M. Tao, and X. Yuan, *Constrained total variation deblurring models and fast algorithms based on alternating direction method of multipliers*, SIAM J. Imaging Sci. **6** (2013), no. 1, 680–697, DOI 10.1137/110860185. MR3036992

[21] R. H. Chan, B. Morini and M. Porcelli, *Affine scaling methods for image deblurring problems*, American Institute of Physics Conference Proceedings **1281** (2010), no. 11, 1043–1046.

[22] H. C. Elman and G. H. Golub, *Inexact and preconditioned Uzawa algorithms for saddle point problems*, SIAM J. Numer. Anal. **31** (1994), no. 6, 1645–1661, DOI 10.1137/0731085. MR1302679

[23] E. Esser, X. Zhang, and T. F. Chan, *A general framework for a class of first order primal-dual algorithms for convex optimization in imaging science*, SIAM J. Imaging Sci. **3** (2010), no. 4, 1015–1046, DOI 10.1137/09076934X. MR2763706

[24] D. Gabay and B. Mercier, *A dual algorithm for the solution of nonlinear variational problems via finite-element approximations*, Comput. Math. Appl. **2** (1976), no. 1, 17–40.

[25] R. Glowinski and A. Marrocco, *Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité, d'une classe de problèmes de Dirichlet non linéaires* (French, with Loose English summary), Rev. Française Automat. Informat. Recherche Opérationnelle Sér. Rouge Anal. Numér. **9** (1975), no. R-2, 41–76. MR0388811

[26] R. Glowinski and P. Le Tallec, *Augmented Lagrangian and Operator-Splitting Methods in Nonlinear Mechanics*, SIAM Studies in Applied Mathematics, vol. 9, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1989. MR1060954

[27] T. Goldstein and S. Osher, *The split Bregman method for L1-regularized problems*, SIAM J. Imaging Sci. **2** (2009), no. 2, 323–343, DOI 10.1137/080725891. MR2496060

[28] B. He and X. Yuan, *Convergence analysis of primal-dual algorithms for a saddle-point problem: from contraction perspective*, SIAM J. Imaging Sci. **5** (2012), no. 1, 119–149, DOI 10.1137/100814494. MR2902659

[29] M. R. Hestenes, *Multiplier and gradient methods*, J. Optimization Theory Appl. **4** (1969), 303–320. MR0271809

[30] G. M. Korpelevič, *An extragradient method for finding saddle points and for other problems* (Russian), Èkonom. i Mat. Metody **12** (1976), no. 4, 747–756. MR0451121

[31] P.-L. Lions and B. Mercier, *Splitting algorithms for the sum of two nonlinear operators*, SIAM J. Numer. Anal. **16** (1979), no. 6, 964–979, DOI 10.1137/0716071. MR551319

[32] B. Martinet, *Régularisation d'inéquations variationnelles par approximations successives* (French), Rev. Française Informat. Recherche Opérationnelle **4** (1970), no. Ser. R-3, 154–158. MR0298899

[33] B. Morini, M. Porcelli, and R. H. Chan, *A reduced Newton method for constrained linear least-squares problems*, J. Comput. Appl. Math. **233** (2010), no. 9, 2200–2212, DOI 10.1016/j.cam.2009.10.006. MR2577759

[34] A. S. Nemirovsky and D. B. Yudin, *Problem Complexity and Method Efficiency in Optimization*, Wiley-Interscience Series in Discrete Mathematics, John Wiley & Sons, Inc., New York, 1983. Translated from the Russian and with a preface by E. R. Dawson. MR702836

[35] Yu. E. Nesterov, *A method for solving the convex programming problem with convergence rate $O(1/k^2)$* (Russian), Dokl. Akad. Nauk SSSR **269** (1983), no. 3, 543–547. MR701288

[36] Yu. E. Nesterov, *Smooth minimization of non-smooth functions*, Math. Program. **103** (2005), no. 1, Ser. A, 127–152, DOI 10.1007/s10107-004-0552-5. MR2166537

[37] Yu. E. Nesterov, *Gradient methods for minimizing composite functions*, Math. Program. **140** (2013), no. 1, Ser. B, 125–161, DOI 10.1007/s10107-012-0629-5. MR3071865

[38] S. Osher, Y. Mao, B. Dong, and W. Yin, *Fast linearized Bregman iteration for compressive sensing and sparse denoising*, Commun. Math. Sci. **8** (2010), no. 1, 93–111. MR2655902

[39] L. D. Popov, *A modification of the Arrow-Hurwitz method of search for saddle points* (Russian), Mat. Zametki **28** (1980), no. 5, 777–784, 803. MR599872

[40] M. J. D. Powell, *A method for nonlinear constraints in minimization problems*, Optimization (Sympos., Univ. Keele, Keele, 1968), Academic Press, London, 1969, pp. 283–298. MR0272403

[41] R. T. Rockafellar, *Monotone operators and the proximal point algorithm*, SIAM J. Control Optimization **14** (1976), no. 5, 877–898. MR0410483

[42] L. I. Rudin, S. Osher, and E. Fatemi, *Nonlinear total variation based noise removal algorithms*, Phys. D **60** (1992), no. 1-4, 259–268. Experimental mathematics: computational issues in nonlinear science (Los Alamos, NM, 1991). MR3363401

[43] P. Tseng, *On the accelerated proximal gradient methods for convex-concave optimization*, manuscript, 2008, available at http://www.math.washington.edu/ tseng/papers/apgm.pdf.

[44] W. Yin, *Analysis and generalizations of the linearized Bregman model*, SIAM J. Imaging Sci. **3** (2010), no. 4, 856–877, DOI 10.1137/090760350. MR2735964

[45] H. Zhang, J.-F. Cai, L. Cheng, and J. Zhu, *Strongly convex programming for exact matrix completion and robust principal component analysis*, Inverse Probl. Imaging **6** (2012), no. 2, 357–372, DOI 10.3934/ipi.2012.6.357. MR2942744

[46] X. Zhang, M. Burger, and S. Osher, *A unified primal-dual algorithm framework based on Bregman iteration*, J. Sci. Comput. **46** (2011), no. 1, 20–46, DOI 10.1007/s10915-010-9408-8. MR2753250

[47] M. Zhu and T. F. Chan, *An efficient primal-dual hybrid gradient algorithm*, CAM Report 08-34, UCLA, Los Angeles, CA, 2008.

Department of Mathematics, Nanjing University, Nanjing, 210093, People's Republic of China

E-mail address: taom@nju.edu.cn

Department of Mathematics, Hong Kong Baptist University, Hong Kong, People's Republic of China

E-mail address: xmyuan@hkbu.edu.hk