Commun Nonlinear Sci Numer Simulat 142 (2025) 108565

Contents lists available at ScienceDirect

Communications in Nonlinear Science and Numerical Simulation

journal homepage: www.elsevier.com/locate/cnsns

Research paper

Perturbed reflected forward backward splitting algorithm for monotone inclusion

Bing Tan [a], Yekini Shehu [b], Tiexiang Li [c,d,e], Xiaolong Qin [f,*]

[a] School of Mathematics and Statistics, Southwest University, Chongqing 400715, China
[b] School of Mathematical Sciences, Zhejiang Normal University, Jinhua 321004, China
[c] School of Mathematics and Shing-Tung Yau Center, Southeast University, Nanjing, China
[d] Nanjing Center for Applied Mathematics, Nanjing 211135, China
[e] Shanghai Institute for Mathematics and Interdisciplinary Sciences, Shanghai 200433, China
[f] School of Mathematical Sciences, Hangzhou Normal University, Hangzhou 311121, China

ARTICLE INFO

MSC:
47J25
65K15
90C25

Keywords:
Monotone inclusion
Variational inequality problem
Reflected forward–backward algorithm
Image deblurring
Signal processing

ABSTRACT

In this paper, we investigate a new reflected forward–backward splitting algorithm with self-adaptive step sizes to solve monotone inclusion problems. The implementation of our algorithm does not require the knowledge of the Lipschitz constant for the Lipschitz continuous monotone operator, unlike existing reflected forward–backward splitting algorithms, which necessitate this information during implementation. The weak convergence theorem of the proposed algorithm is given under standard conditions. We compare numerically our algorithm with other related ones in the literature through its applications in signal processing and image deblurring.

1. Introduction

Let $\mathcal{H}$ be a real Hilbert space with inner product $\langle\cdot,\cdot\rangle$ and induced norm $\|\cdot\|$. We study the following monotone inclusion problem:

$$\text{Find } x^* \in \mathcal{H} \text{ such that } \mathbf{0} \in Ax^* + Bx^*, \tag{IP}$$

where $A : \mathcal{H} \to 2^{\mathcal{H}}$ is a set-valued maximal monotone operator, and $B : \mathcal{H} \to \mathcal{H}$ is a single-valued monotone and Lipschitz continuous operator. Inclusion problem (IP) is closely related to variational inequalities and split feasibility problems. It is widely applied in signal processing, image restoration, and machine learning problems (see, e.g., [1–3]).

Numerous splitting algorithms have been proposed in the literature to solve monotone inclusion problem (IP) recently. One of the celebrated splitting algorithms for solving the monotone inclusion problem (IP) with the condition that B is β-cocoercive (also called β-inverse-strongly monotone), is the forward–backward splitting algorithm, devised by Lions and Mercier [4], which features one forward evaluation of B and a backward evaluation of A at each iteration:

$$x_{n+1} = J_{\lambda_n A}(x_n - \lambda_n B x_n), \tag{1}$$

where $J_{\lambda_n A} := (I + \lambda_n A)^{-1}$ is the resolvent operator (see [5]) and λ_n is a positive constant. This method is known to converge weakly to a point in $(A+B)^{-1}(\mathbf{0})$ provided that $\lambda_n \in (0, 2\beta)$. Other than the cocoercivity property of B, strong monotonicity of $A+B$ (see [6])

* Corresponding author.
E-mail addresses: bingtan@swu.edu.cn (B. Tan), yekini.shehu@zjnu.edu.cn (Y. Shehu), txli@seu.edu.cn (T. Li), qxlxajh@163.com (X. Qin).

https://doi.org/10.1016/j.cnsns.2024.108565
Received 14 July 2024; Received in revised form 5 December 2024; Accepted 19 December 2024
Available online 25 December 2024

also guarantees the convergence of Algorithm (1). We observe that monotonicity and Lipschitz continuity conditions are weaker conditions on B than cocoercivity and strong monotonicity.

If the cocoercivity condition on B in monotone inclusion problem (IP) is relaxed such that B is monotone and Lipschitz continuous, then the forward–backward–forward splitting algorithm (also called Tseng splitting algorithm), proposed by Tseng [7], is as:

$$\begin{cases} y_n = J_{\lambda_n A}(x_n - \lambda_n B x_n), \\ x_{n+1} = y_n - \lambda_n(B y_n - B x_n). \end{cases} \tag{2}$$

Although forward–backward–forward splitting algorithm (2) requires an additional forward evaluation of B, the weak convergence result can be obtained when B is monotone and L-Lipschitz continuous. To achieve the strong convergence of Algorithm (2), Gibali and Thong [8] proposed two strongly convergent forward–backward–forward algorithms for solving monotone inclusion problems in real Hilbert spaces by utilizing the Mann-type method and the viscosity method. Their algorithms are stated as follows:

$$\begin{cases} y_n = J_{\lambda_n A}(x_n - \lambda_n B x_n), \\ z_n = y_n - \lambda_n(B y_n - B x_n), \\ x_{n+1} = (1-\alpha_n-\beta_n)x_n + \beta_n z_n, \\ \lambda_{n+1} = \begin{cases} \min\left\{ \dfrac{\mu \|x_n - y_n\|}{\|B x_n - B y_n\|}, \lambda_n \right\}, & \text{if } B x_n - B y_n \neq 0, \\ \lambda_n, & \text{otherwise}, \end{cases} \end{cases} \tag{3}$$

where $\lambda_0 > 0$, $\mu \in (0,1)$, $\{\alpha_n\}$ and $\{\beta_n\}$ are two real sequences in $(0,1)$ such that $\{\beta_n\} \subset (a,b) \subset (0, 1-\alpha_n)$ for some $a > 0, b > 0$ and $\lim_{n\to\infty} \alpha_n = 0, \sum_{n=1}^{\infty} \alpha_n = \infty$.

$$\begin{cases} y_n = J_{\lambda_n A}(x_n - \lambda_n B x_n), \\ z_n = y_n - \lambda_n(B y_n - B x_n), \\ x_{n+1} = \alpha_n f(x_n) + (1-\alpha_n) z_n, \\ \lambda_{n+1} = \begin{cases} \min\left\{ \dfrac{\mu \|x_n - y_n\|}{\|B x_n - B y_n\|}, \lambda_n \right\}, & \text{if } B x_n - B y_n \neq 0, \\ \lambda_n, & \text{otherwise}, \end{cases} \end{cases} \tag{4}$$

where $\lambda_0 > 0$, $\mu \in (0,1)$, $\{\alpha_n\}$ is a real sequences in $(0,1)$ such that $\lim_{n\to\infty} \alpha_n = 0, \sum_{n=1}^{\infty} \alpha_n = \infty$, and $f : \mathcal{H} \to \mathcal{H}$ is a contraction mapping with constant $\rho \in (0,1)$. Under the conditions that operator A is maximally monotone and operator B is monotone and Lipschitz continuous, they established strong convergence theorems for the proposed Algorithm (3) and Algorithm (4). It is noted that Algorithms (3) and (4) employ an adaptive step size rule, allowing them to operate without prior knowledge of the Lipschitz constant. To accelerate the convergence speed of the algorithm, Polyak [9] introduced a method known as the inertial technique, which is derived from a second-order dissipative dynamical system. Recently, Alakoya et al. [10] suggested an inertial version of Algorithm (4), as described below:

$$\begin{cases} \theta_n = \begin{cases} \min\left\{ \dfrac{\epsilon_n}{\|x_n - x_{n-1}\|}, \theta \right\}, & \text{if } x_n \neq x_{n-1}, \\ \theta, & \text{otherwise}, \end{cases} \\ w_n = x_n + \theta_n \left(x_n - x_{n-1}\right), \\ y_n = J_{\lambda_n A}(w_n - \lambda_n B w_n), \\ z_n = y_n - \lambda_n(B y_n - B w_n), \\ x_{n+1} = \alpha_n f(w_n) + (1-\alpha_n) z_n, \\ \lambda_{n+1} = \begin{cases} \min\left\{ \dfrac{\mu \|w_n - y_n\|}{\|B w_n - B y_n\|}, \lambda_n + \xi_n \right\}, & \text{if } B w_n - B y_n \neq 0, \\ \lambda_n + \xi_n, & \text{otherwise}, \end{cases} \end{cases} \tag{5}$$

where $\theta > 0$, $\lambda_0 > 0$, $\mu \in (0,1)$, $\{\alpha_n\} \subset (0,1)$, $\lim_{n\to\infty} \alpha_n = 0$, $\sum_{n=1}^{\infty} \alpha_n = +\infty$, $\{\epsilon_n\}$ is a positive sequence satisfying $\lim_{n\to\infty}(\epsilon_n/\alpha_n) = 0$, $\{\xi_n\}$ is a nonnegative sequence such that $\sum_{n=1}^{\infty} \xi_n < +\infty$, $f : \mathcal{H} \to \mathcal{H}$ is a contraction mapping with coefficient $\rho \in (0,1)$, operator B is monotone and uniformly continuous, and operator A is maximal monotone. It should be pointed out that Algorithm (5) uses a non-monotonic step size rule, which is computationally superior to the non-increasing step size rule of Algorithms (3) and (4). Alakoya et al. [10] demonstrated the strong convergence of the iterative sequence generated by the proposed Algorithm (5) and provided several numerical examples to showcase its computational advantages over some known adaptive algorithms.

It is noted that the algorithms in [7,8,10] use the same step size in each iteration. In 2020, Gibali et al. [11] introduced a modified inertial projection and contraction algorithm to address monotone inclusion problem (IP), which uses different step sizes in each iteration to compute the values of the sequence. Their iterative procedure is as follows:

$$\begin{cases} w_n = x_n + \theta_n \left(x_n - x_{n-1}\right), \\ y_n = J_{\lambda_n A}(w_n - \lambda_n B w_n), \\ d_n = w_n - y_n - \lambda_n \left(B w_n - B y_n\right), \\ \beta_n = \langle w_n - y_n, d_n \rangle / \|d_n\|^2, \\ x_{n+1} = w_n - \gamma \beta_n d_n. \end{cases} \tag{6}$$

The weak convergence of Algorithm (6) was obtained under the conditions that operator A satisfies maximal monotonicity, operator B satisfies monotonicity and L-Lipschitz continuity, and the parameters meet the conditions of $\gamma \in (1,2)$, $\{\theta_n\} \subset [0,1)$, and $\{\lambda_n\} \subset (0, 1/L)$. Note that Algorithm (6) also requires two forward evaluations of B and one backward evaluation of A in each iteration. Thus, its computational complexity is the same as that of Algorithms (2), (3), (4), and (5). Other related splitting algorithms to solve monotone inclusion problem (IP) can be found in [12,13], which also only needs two forward evaluation of B and one backward evaluation of set-valued maximal monotone operator A per iteration. To reduce the number of evaluations of the operator B in each iteration, Malitsky and Tam [14] proposed a novel fixed-step algorithm for finding solutions of monotone inclusion problems, now known as the forward-reflected-backward splitting algorithm. Their algorithm requires only one forward evaluation and one backward evaluation per iteration, whereas Algorithm (2) needs two forward evaluations. Moreover, they proved the weak convergence theorem of the algorithm under the condition that operator B is monotone and Lipschitz continuous, which is the same condition as in Algorithm (2). Therefore, the computational complexity of Malitsky and Tam's algorithm is the same as Algorithm (1), but the former guarantees convergence under weaker conditions. On the other hand, Malitsky and Tam also introduced a forward-reflected-backward splitting algorithm with a linesearch technique, as described below:

$$\begin{cases} \text{Having } x_n, \lambda_{n-1}, \text{ and } Bx_{n-1}, \text{ choose } \rho \in \left\{1, \sigma^{-1}\right\} \text{ and compute} \\ x_{n+1} = J_{\lambda_n A}\left(x_n - \lambda_n B x_n - \lambda_{n-1}(B x_n - B x_{n-1})\right), \\ \text{where } \lambda_n = \rho \lambda_{n-1} \sigma^i \text{ with } i \text{ being the smallest nonnegative integer} \\ \text{satisfying } \lambda_n \left\|B x_{n+1} - B x_n\right\| \le \mu/2 \cdot \left\|x_{n+1} - x_n\right\|, \end{cases} \tag{7}$$

where $\lambda_0, \lambda_{-1} > 0$, $\mu \in (0,1)$, and $\sigma \in (0,1)$. Under the conditions that operator B is monotone and locally Lipschitz continuous, they proved a weak convergence theorem for Algorithm (7). It is worth noting that Algorithm (7) requires at least two forward evaluations per iteration due to its use of the Armijo linesearch criterion to determine an appropriate step size, which increases the computational burden when B is complex.

Recently, Cevher and Vũ [15] proposed and studied the reflected forward–backward splitting algorithm for monotone inclusion problem (IP) when B is L-Lipschitz continuous and monotone operator:

$$\begin{cases} y_n = 2x_n - x_{n-1}, \\ x_{n+1} = J_{\lambda_n A}(x_n - \lambda_n B y_n), \end{cases} \tag{8}$$

and obtained weak convergence results when $\lambda_n \in (0, (\sqrt{2}-1)/L)$. Reflected forward–backward splitting algorithm (8) features one forward evaluation of B and one backward evaluation of set-valued maximal monotone operator A per iteration. This property can reduce computational complexity encountered in Algorithms (2)–(7). Furthermore, reflected forward–backward splitting algorithm (8) extends the results of [16] from monotone variational inequalities to monotone inclusion problem (IP). However, it is required to know the Lipschitz constant L of monotone operator B during its implementation. This is a drawback.

The requirement of the knowledge of the Lipschitz constant of B in implementing reflected forward–backward splitting algorithm (8) makes us to ask this question: *Can one devise a new version of reflected forward–backward splitting algorithm such that step sizes are implemented self-adaptively without the knowledge of the Lipschitz constant L of B?*

In this paper, we answer the question above affirmatively. To be more precise, we propose a modified version of reflected forward–backward splitting algorithm (8) such that the step sizes λ_n are implemented adaptive without the knowledge of the Lipschitz constant of B. In this case, our algorithm would be more applicable to solve monotone inclusion problem (IP). Specifically, our contributions manifest in the following three aspects:

(i) we modify the reflected forward–backward splitting algorithm (8) such that step sizes $\{\lambda_n\}$ are self-adaptively generated;
(ii) we establish a weak convergence result of our algorithm;
(iii) we apply our results to signal processing and image deblurring problems, and we compare our algorithm with Algorithms (2)–(8).

The remainder of the paper is organized as follows. In Section 2, we provide some relevant definitions and useful lemmas for the convergence analysis. The proposed splitting algorithm and its weak convergence result are stated in Section 3. Section 4 gives numerical experiments on signal processing and image deblurring, where we demonstrate the superiority of the proposed algorithm over related existing splitting algorithms in [7,8,10,11,14,15]. The paper is finalized with some concluding remarks in Section 5.

2. Preliminaries

The graph of $A : \mathcal{H} \to 2^{\mathcal{H}}$, denoted by $\mathrm{gph}(A)$, is defined by

$$\mathrm{gph}(A) := \{(x,u) \in \mathcal{H} \times \mathcal{H} : x \in \mathrm{dom}(A),\ u \in A(x)\}.$$

We say that the operator A is

(i) *monotone* if $\langle u - v, x - y\rangle \geq 0$ for all (x,u), $(y,v) \in \mathrm{gph}(A)$;
(ii) *maximal monotone* if it is monotone and $\mathrm{gph}(A) \supset \mathrm{gph}(B)$, where B is any other monotone operator;
(iii) *η-strongly monotone* if there exists $\eta > 0$ such that $\langle u - v, x - y\rangle \geq \eta\|x - y\|^2$ for all (x,u), $(y,v) \in \mathrm{gph}(A)$.

Observe that the maximal monotonicity of A is equivalent to: it is monotone and if, for any $(x,u) \in \mathcal{H} \times \mathcal{H}$, $\langle u - v, x - y\rangle \geq 0$ for all $(y,v) \in \mathrm{gph}(A)$, then it follows that $u \in A(x)$. For a given maximal monotone operator A, the resolvent $J_{\lambda A}(x) := (I + \lambda A)^{-1}(x)$ for $x \in \mathcal{H}$ and $\lambda > 0$ is a single-valued mapping, where I is the identity operator on $\mathcal{H}$. Furthermore, $\|J_{\lambda A}(x) - J_{\lambda A}(y)\| \leq \|x - y\|$ for all $x, y \in \mathcal{H}$.

An operator $B : \mathcal{H} \to \mathcal{H}$ is called:

(i) L-Lipschitz continuous if there exists a number $L > 0$ such that $\|Bx - By\| \leq L\|x - y\|$ for all $x, y \in \mathcal{H}$;
(ii) Uniformly continuous if, for every $\epsilon > 0$, there exists $\delta = \delta(\epsilon) > 0$, such that $\|Bx - By\| < \epsilon$ for all $x, y \in \mathcal{H}$ whenever $\|x - y\| < \delta$;
(iii) β-cocoercive or β-inverse-strongly monotone, if there exists $\beta > 0$ such that $\langle Bx - By, x - y\rangle \geq \beta\|Bx - By\|^2$ for all $x, y \in \mathcal{H}$.

Note that cocoercivity and strong monotonicity are dual: B is β-cocoercive $\Leftrightarrow$ B^{-1} is β-strongly monotone. The following result gives the maximal monotonicity of the sum of two monotone operators.

Lemma 2.1 ([17]). *Let $A : \mathcal{H} \to 2^{\mathcal{H}}$ be a maximal monotone operator, and let $B : \mathcal{H} \to \mathcal{H}$ be a Lipschitz continuous and monotone operator. Then $A + B$ is a maximal monotone operator.*

We use the following trivial identities in our convergence analysis.

Lemma 2.2. *Let $x, y, z \in \mathcal{H}$ and $\alpha \in \mathbb{R}$. Then*

(i) $2\langle x, y\rangle = \|x\|^2 + \|y\|^2 - \|x - y\|^2 = \|x + y\|^2 - \|x\|^2 - \|y\|^2$.
(ii) $\|\alpha x + (1 - \alpha)y\|^2 = \alpha\|x\|^2 + (1 - \alpha)\|y\|^2 - \alpha(1 - \alpha)\|x - y\|^2$.

The lemma below plays an important role in proving weak convergence of sequences in a Hilbert space.

Lemma 2.3 ([18]). *Let C be a nonempty subset of a Hilbert space $\mathcal{H}$, and let $\{x_n\}$ be a bounded sequence in $\mathcal{H}$. Assume that the following two conditions are satisfied:*

(i) $\lim_{n\to\infty} \|x_n - x\|$ *exists for each* $x \in C$,
(ii) every weak cluster point of $\{x_n\}$ *belongs to* C.

Then $\{x_n\}$ *converges weakly to a point in* C.

3. Main results

In this section, we propose an adaptive reflected forward–backward splitting algorithm to solve monotone inclusion problem (IP) and analyze its weak convergence. The advantage of the proposed algorithm lies in its ability to adaptively operate without requiring prior knowledge of the Lipschitz constant of the operator B. Moreover, we apply our results to variational inequality problems.

3.1. Weak convergence analysis

We now introduce our adaptive splitting algorithm for the inclusion problem (IP) with self-adaptive step sizes.

Remark 3.1. We have the following comments on the suggested Algorithm 1.

(i) Unlike Algorithms (2), (6), and (8), the proposed Algorithm 1 does not require the Lipschitz constant estimate of Lipschitz continuous monotone operator B as an input parameter. Moreover, our algorithm employs a non-monotonic step size rule, which generates a non-monotonic sequence of step sizes. This is superior to the non-increasing step size criteria used in [11] and the Armijo-type linesearch step size rule applied in [14].
(ii) Note that Algorithm (1) and Algorithm (8) each require one evaluation of operators A and B per iteration, Algorithm (2) to Algorithm (6) need two evaluations of operator B and one evaluation of operator A, while Algorithm (7) performs at least two evaluations of operator B in each iteration. It should be point out that the terms Bx_{n+1} and By_n computed in the current iteration are the same as Bx_n and By_{n-1} used in the next iteration. Therefore, our Algorithm calls two forward evaluations and one backward evaluation at each iteration, which means its computational complexity is the same as the Tseng's algorithm [7].

Algorithm 1 The perturbed reflected forward-backward with adaptive step sizes

Initialization: Pick $\lambda_{-1} = \lambda_0 > 0$ and $x_{-2} = x_{-1} = x_0 \in \mathcal{H}$. Select $\mu \in (0, 1/5)$. Let $\{\xi_n\}$ be a nonnegative real numbers sequence such that $\sum_{n=0}^{\infty} \xi_n < +\infty$. Set $n = 0$.

Iteration: With x_{n-1}, x_n, λ_{n-1}, λ_n, Bx_n, and By_{n-1}, compute

$$\begin{cases} y_n = 2x_n - x_{n-1}, \\ x_{n+1} = J_{\lambda_n A}\big(x_n - \lambda_n By_n - \lambda_{n-1}(Bx_n - By_{n-1})\big), \end{cases}$$

where

$$\lambda_{n+1} = \begin{cases} \min\left\{ \dfrac{\mu\|y_n - x_{n+1}\|}{\|By_n - Bx_{n+1}\|},\ \lambda_n + \xi_n \right\}, & \text{if } Bx_n \neq Bx_{n+1}, \\ \lambda_n + \xi_n, & \text{otherwise.} \end{cases} \tag{9}$$

To analyze the convergence of Algorithm 1, we need the following assumption.

Assumption 3.2.

(i) $A : \mathcal{H} \to 2^{\mathcal{H}}$ is set-valued maximal monotone;
(ii) $B : \mathcal{H} \to \mathcal{H}$ is a single-valued monotone and uniformly continuous;
(iii) The solution set $(A + B)^{-1}(\mathbf{0})$ of inclusion problem (IP) is nonempty.

We mention here that there are a lot of examples which satisfies the (ii) in Assumption 3.2 but is not Lipschitz continuous, such as $f(x) := exp(x)$ for all $x \in [0, \infty)$. Then f is uniformly continuous and monotone. However, f is not Lipschitz continuous. In addition, $f(x) := \sqrt{x}$ for all $x \in [0, \infty)$. Then f is also uniformly continuous and monotone. However, f is not Lipschitz continuous. Indeed, there are a lot of examples, which satisfy Assumption 3.2 (ii) but are not Lipschitz continuous. The following two lemmas are crucial for the convergence analysis of Algorithm 1.

Lemma 3.3 (*[10, Lemma 4.1]*). *Let $\{\lambda_n\}$ be the sequence generated by (9). Then $\{\lambda_n\}$ is well-defined and $\lim_{n\to\infty} \lambda_n = \lambda \in \left[\min\{\lambda_0, \mu/Q\}, \lambda_0 + \sum_{n=0}^{\infty} \xi_n\right]$ for some $Q > 0$.*

Lemma 3.4. *The sequences $\{x_n\}$ generated by Algorithm 1 are bounded when Assumption 3.2 is satisfied.*

Proof. By using the definition of x_{n+1}, one obtains

$$x_n - x_{n+1} - \lambda_n By_n - \lambda_{n-1}(Bx_n - By_{n-1}) \in \lambda_n Ax_{n+1}.$$

Pick $x^* \in (A + B)^{-1}(\mathbf{0})$. It follows that $-\lambda_n Bx^* \in \lambda_n Ax^*$. From the monotonicity of A, one has

$$\langle x_n - x_{n+1} - \lambda_n By_n - \lambda_{n-1}(Bx_n - By_{n-1}) + \lambda_n Bx^*, x_{n+1} - x^* \rangle \geq 0.$$

Thus

$$\begin{aligned} 0 &\leq 2\langle x_n - x_{n+1}, x_{n+1} - x^* \rangle - 2\lambda_n \langle By_n - Bx^*, x_{n+1} - x^* \rangle \\ &\quad + 2\lambda_{n-1}\langle By_{n-1} - Bx_n, x_{n+1} - x^* \rangle \\ &= 2\langle x_n - x_{n+1}, x_{n+1} - x^* \rangle - 2\lambda_n \langle By_n - Bx_{n+1}, x_{n+1} - x^* \rangle \\ &\quad - 2\lambda_n \langle Bx_{n+1} - Bx^*, x_{n+1} - x^* \rangle + 2\lambda_{n-1}\langle By_{n-1} - Bx_n, x_{n+1} - x_n \rangle \\ &\quad + 2\lambda_{n-1}\langle By_{n-1} - Bx_n, x_n - x^* \rangle, \end{aligned}$$

which together with the monotonicity of B implies that

$$\begin{aligned} 0 &\leq 2\langle x_n - x_{n+1}, x_{n+1} - x^* \rangle - 2\lambda_n \langle By_n - Bx_{n+1}, x_{n+1} - x^* \rangle \\ &\quad + 2\lambda_{n-1}\langle By_{n-1} - Bx_n, x_n - x^* \rangle + 2\lambda_{n-1}\langle By_{n-1} - Bx_n, x_{n+1} - x_n \rangle. \end{aligned}$$

On account of Lemma 2.2 (i), we obtain

$$\begin{aligned} 0 &\leq \|x_n - x^*\|^2 - \|x_n - x_{n+1}\|^2 - \|x_{n+1} - x^*\|^2 \\ &\quad - 2\lambda_n \langle By_n - Bx_{n+1}, x_{n+1} - x^* \rangle + 2\lambda_{n-1}\langle By_{n-1} - Bx_n, x_n - x^* \rangle \\ &\quad + 2\lambda_{n-1}\langle By_{n-1} - Bx_n, x_{n+1} - x_n \rangle. \end{aligned} \tag{10}$$

By using the Cauchy–Schwarz inequality and (9), we see that

$$\begin{aligned} &2\lambda_{n-1}\langle By_{n-1} - Bx_n, x_{n+1} - x_n\rangle \\ &\leq 2\lambda_{n-1}\|By_{n-1} - Bx_n\|\|x_{n+1} - x_n\| \\ &\leq 2\lambda_{n-1}\frac{\mu\|y_{n-1} - x_n\|}{\lambda_n}\|x_{n+1} - x_n\| \\ &\leq \frac{\mu\lambda_{n-1}}{\lambda_n}\|y_{n-1} - x_n\|^2 + \frac{\mu\lambda_{n-1}}{\lambda_n}\|x_{n+1} - x_n\|^2. \end{aligned} \tag{11}$$

Plugging (11) into (10), we arrive at

$$\begin{aligned} &\|x_{n+1} - x^*\|^2 + 2\lambda_n\langle By_n - Bx_{n+1}, x_{n+1} - x^*\rangle \\ &\leq \|x_n - x^*\|^2 - \|x_n - x_{n+1}\|^2 + 2\lambda_{n-1}\langle By_{n-1} - Bx_n, x_n - x^*\rangle \\ &\quad + \frac{\mu\lambda_{n-1}}{\lambda_n}\|y_{n-1} - x_n\|^2 + \frac{\mu\lambda_{n-1}}{\lambda_n}\|x_{n+1} - x_n\|^2. \end{aligned}$$

Consequently,

$$\begin{aligned} &\|x_{n+1} - x^*\|^2 + 2\lambda_n\langle By_n - Bx_{n+1}, x_{n+1} - x^*\rangle + \frac{\mu\lambda_n}{\lambda_{n+1}}\|x_{n+1} - y_n\|^2 \\ &\leq \|x_n - x^*\|^2 + 2\lambda_{n-1}\langle Bx_n - By_{n-1}, x^* - x_n\rangle + \frac{\mu\lambda_{n-1}}{\lambda_n}\|x_n - y_{n-1}\|^2 \\ &\quad - \left(1 - \frac{\mu\lambda_{n-1}}{\lambda_n}\right)\|x_{n+1} - x_n\|^2 + \frac{\mu\lambda_n}{\lambda_{n+1}}\|x_{n+1} - y_n\|^2. \end{aligned}$$

Let us define

$$t_n := \|x_n - x^*\|^2 + 2\lambda_{n-1}\langle Bx_n - By_{n-1}, x^* - x_n\rangle + \frac{\mu\lambda_{n-1}}{\lambda_n}\|x_n - y_{n-1}\|^2.$$

Hence

$$t_{n+1} \leq t_n - \left(1 - \frac{\mu\lambda_{n-1}}{\lambda_n}\right)\|x_{n+1} - x_n\|^2 + \frac{\mu\lambda_n}{\lambda_{n+1}}\|x_{n+1} - y_n\|^2. \tag{12}$$

Note that

$$\begin{aligned} \|x_{n+1} - y_n\|^2 &= \|(x_{n+1} - x_n) - (x_n - x_{n-1})\|^2 \\ &\leq 2\|x_{n+1} - x_n\|^2 + 2\|x_{n-1} - x_n\|^2. \end{aligned} \tag{13}$$

Plugging (13) into (12), we include that

$$\begin{aligned} t_{n+1} &\leq t_n - \left(1 - \frac{\mu\lambda_{n-1}}{\lambda_n}\right)\|x_{n+1} - x_n\|^2 \\ &\quad + \frac{2\mu\lambda_n}{\lambda_{n+1}}\|x_{n+1} - x_n\|^2 + \frac{2\mu\lambda_n}{\lambda_{n+1}}\|x_n - x_{n-1}\|^2. \end{aligned}$$

Therefore,

$$\begin{aligned} &t_{n+1} + \frac{2\mu\lambda_{n+1}}{\lambda_{n+2}}\|x_{n+1} - x_n\|^2 \\ &\leq t_n + \frac{2\mu\lambda_n}{\lambda_{n+1}}\|x_n - x_{n-1}\|^2 - \left(1 - \frac{\mu\lambda_{n-1}}{\lambda_n} - \frac{2\mu\lambda_n}{\lambda_{n+1}} - \frac{2\mu\lambda_{n+1}}{\lambda_{n+2}}\right)\|x_{n+1} - x_n\|^2. \end{aligned} \tag{14}$$

From the definition of t_n, the Cauchy–Schwarz inequality, and (9), we deduce that

$$\begin{aligned} t_n &= \|x_n - x^*\|^2 + 2\lambda_{n-1}\langle Bx_n - By_{n-1}, x^* - x_n\rangle + \frac{\mu\lambda_{n-1}}{\lambda_n}\|x_n - y_{n-1}\|^2 \\ &\geq \|x_n - x^*\|^2 - 2\lambda_{n-1}\|Bx_n - By_{n-1}\|\|x^* - x_n\| + \frac{\mu\lambda_{n-1}}{\lambda_n}\|x_n - y_{n-1}\|^2 \\ &\geq \|x_n - x^*\|^2 - \frac{\mu\lambda_{n-1}}{\lambda_n}\|x_n - y_{n-1}\|^2 - \frac{\mu\lambda_{n-1}}{\lambda_n}\|x_n - x^*\|^2 + \frac{\mu\lambda_{n-1}}{\lambda_n}\|x_n - y_{n-1}\|^2 \\ &= \left(1 - \frac{\mu\lambda_{n-1}}{\lambda_n}\right)\|x_n - x^*\|^2. \end{aligned}$$

Since $\lim\left(1 - \frac{\mu\lambda_{n-1}}{\lambda_n}\right) = 1 - \mu > 0$, we see that there exists $n_1 \in \mathbb{N}$ such that $1 - \frac{\mu\lambda_{n-1}}{\lambda_n} > 0$ for all $n \geq n_1$. Consequently, $t_n \geq 0$ for all $n \geq n_1$. Let

$$s_n := t_n + \frac{2\mu\lambda_n}{\lambda_{n+1}}\|x_n - x_{n-1}\|^2.$$

Note that $s_n \geq 0$ for all $n \geq n_1$. It follows from (14) that

$$s_{n+1} \leq s_n - \Big(1 - \frac{\mu\lambda_{n-1}}{\lambda_n} - \frac{2\mu\lambda_n}{\lambda_{n+1}} - \frac{2\mu\lambda_{n+1}}{\lambda_{n+2}}\Big)\|x_{n+1} - x_n\|^2. \tag{15}$$

Since $\lim_{n\to\infty}\Big(1 - \frac{\mu\lambda_{n-1}}{\lambda_n} - \frac{2\mu\lambda_n}{\lambda_{n+1}} - \frac{2\mu\lambda_{n+1}}{\lambda_{n+2}}\Big) = 1 - 5\mu > 0$, then there exists $n_2 \in \mathbb{N}$, where $n_2 \geq n_1$ such that

$$1 - \frac{\mu\lambda_{n-1}}{\lambda_n} - \frac{2\mu\lambda_n}{\lambda_{n+1}} - \frac{2\mu\lambda_{n+1}}{\lambda_{n+2}} > 0, \quad \forall n \geq n_2.$$

Then $\{s_n\}$ is non-increasing, $\lim_{n\to\infty} s_n$ exists, and $\{s_n\}$ is bounded. We can then easily obtain that $\{x_n\}$ is bounded. □

Next, we give our weak convergence result.

Theorem 3.5. *The sequence $\{x_n\}$ generated by Algorithm 1 converge weakly to a point in $(A+B)^{-1}(\mathbf{0})$ provided that Assumption 3.2 is satisfied.*

Proof. We obtain from (15) that $\lim_{n\to\infty}\|x_{n+1} - x_n\| = 0$. Consequently, $\lim_{n\to\infty}\|x_n - y_n\| = 0$. The two limits imply that $\lim_{n\to\infty}\|x_{n+1} - y_n\| = 0$. From the fact that $\{x_n\}$ is bounded, one deduces that $\{x_n\}$ has a weakly convergent subsequence $\{x_{n_m}\}$ such that $\{x_{n_m}\}$ converges weakly to $\bar{x} \in \mathcal{H}$. It follows from the definition of x_{n+1} that

$$x_{n_m} - \lambda_{n_m} B y_{n_m} - \lambda_{n_m-1}(Bx_{n_m} - By_{n_m-1}) \in (I + \lambda_{n_m} A)x_{n_m+1}.$$

Therefore,

$$\frac{x_{n_m} - x_{n_m+1}}{\lambda_{n_m}} + Bx_{n_m+1} - By_{n_m} - \frac{\lambda_{n_m-1}}{\lambda_{n_m}}(Bx_{n_m} - By_{n_m-1}) \in (A+B)x_{n_m+1}. \tag{16}$$

Since B is uniformly continuous, we have $\|Bx_{n_m+1} - By_{n_m}\| \to 0$ as $m \to \infty$. By Lemma 2.1, one obtains that $A + B$ is maximal monotone. Therefore, the graph of $A + B$ is demiclosed. Passing to the limit in (16) (noting that $\lambda_{n_m} \to \lambda$ as $m \to \infty$), one arrives at $\bar{x} \in (A+B)^{-1}(\mathbf{0})$. Now, one defines

$$d_n := -\frac{\mu\lambda_{n-1}}{\lambda_n}\|x_n - y_{n-1}\|^2 - \frac{2\mu\lambda_n}{\lambda_{n+1}}\|x_n - x_{n-1}\|^2 - 2\lambda_{n-1}\langle Bx_n - By_{n-1}, x^* - x_n\rangle.$$

From the fact that B is uniformly continuous and $\{x_n\}$ is bounded, $\lim_{n\to\infty}\|x_n - x_{n-1}\| = 0$, and $\lim_{n\to\infty}\|x_n - y_{n-1}\| = 0$, one has $\lim_{n\to\infty} d_n = 0$. From the definitions of s_n and d_n, one has $\|x_n - x^*\|^2 = s_n + d_n$. By using the existence of the limit of $\{s_n\}$, one asserts that $\lim_{n\to\infty}\|x_n - x^*\|$ exists. Therefore, $\{x_n\}$ converges weakly to a point in $(A+B)^{-1}(\mathbf{0})$ by means of Lemma 2.3. □

3.2. Application to variational inequalities

In this subsection, we apply the proposed Algorithm 1 to the variational inequality problem. Let C be a nonempty, closed, and convex subset of a real Hilbert space $\mathcal{H}$, and let $M : \mathcal{H} \to \mathcal{H}$ be an operator. The variational inequality problem is described as:

$$\text{Find } x^* \in C \text{ such that } \langle Mx^*, x - x^*\rangle \geq 0, \quad \forall x \in C. \tag{VI}$$

The solution set of problem (VI) is denoted by $\mathrm{VI}(M, C)$.

Let $f : \mathcal{H} \to (-\infty, +\infty]$ be a proper, lower semicontinuous, and convex function. The subdifferential of f at $x \in \mathrm{dom}\, f$ is defined by $\partial f(x) := \{z \in \mathcal{H} : f(y) - f(x) \geq \langle z, y - x\rangle\}$. The normal cone of C at $v \in C$ is given as $N_C(v) := \{d \in \mathcal{H} : \langle d, y - v\rangle \leq 0, \forall y \in C\}$. Let $\delta_C(x)$ be the indicator function of C at x, that is,

$$\delta_C(x) := \begin{cases} 0, & \text{if } x \in C, \\ \infty, & \text{if } x \notin C. \end{cases}$$

We can frame variational inequality problem (VI) within the context of monotone inclusion problem (IP) by setting $A = \partial\delta_C$ and $B = M$. That is, the solution to problem (VI) is equivalent to the solution of the monotone inclusion problem: find $x^* \in \mathcal{H}$ such that $\mathbf{0} \in (\partial\delta_C + M)(x^*)$. Indeed, it can be seen that $\delta_C : \mathcal{H} \to (-\infty, +\infty]$ is a proper, lower semicontinuous, and convex function. Moreover, one knows that ∂f is maximal monotone and $\partial\delta_C(x) = N_C(x)$. Let $A = \partial\delta_C$. Then

$$\begin{aligned} v = J_{\lambda_n A}(x) &\iff x \in v + \lambda_n N_C(v) \\ &\iff \langle x - v, y - v\rangle \leq 0, \ \forall y \in C \iff v = P_C(x), \end{aligned}$$

where P_C is the projection operator from $\mathcal{H}$ into C, which is defined by $P_C(x) = \min\{y \in C : \|x - y\|\}$ for a fixed $x \in \mathcal{H}$.

Now, we consider (VI) as an application of Theorem 3.5.

Theorem 3.6. *Let M be a monotone and uniformly continuous operator, and let $\mathrm{VI}(M,C)\neq\emptyset$. Pick $\lambda_{-1}=\lambda_0>0$ and $x_{-2}=x_{-1}=x_0\in\mathcal{H}$. Select $\mu\in(0,1/5)$. Let $\{\xi_n\}$ be a nonnegative real numbers sequence such that $\sum_{n=0}^{\infty}\xi_n<+\infty$. Let $\{x_n\}$ be the sequence generated by the following iterative procedure:*

$$\begin{cases} y_n = 2x_n - x_{n-1}, \\ x_{n+1} = P_C\big(x_n - \lambda_n M y_n - \lambda_{n-1}(Mx_n - My_{n-1})\big), \\ \lambda_{n+1} = \begin{cases} \min\left\{\dfrac{\mu\|y_n - x_{n+1}\|}{\|My_n - Mx_{n+1}\|},\ \lambda_n + \xi_n\right\}, & \text{if } Mx_n \neq Mx_{n+1}, \\ \lambda_n + \xi_n, & \text{otherwise.} \end{cases} \end{cases} \tag{17}$$

Then the sequence $\{x_n\}$ generated by Algorithm (17) converges weakly to a solution of problem (VI).

4. Numerical experiments

In this section, we explore the problems related to signal processing and image deblurring, which have generated considerable interest in the field of contemporary science and technology. The primary goal is to recover a signal/an image from the available data. A task that can be conceptualized as the subsequent inverse problem described by the equation:

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{z}, \tag{18}$$

where $\mathbf{y}$ represents the observed data, $\mathbf{x}$ corresponds to the true (original) underlying data, $\mathbf{z}\in\mathbb{R}^m$ accounts for the measurement error, and $\mathbf{A}$ is an operator that can be either linear or nonlinear. This operator can take the form of a convolution operator in the context of deblurring, a Radon transform in X-ray computer tomography (CT), or a sampling mask applied in the Fourier domain for magnetic resonance imaging (MRI), as detailed in [19–21]. To solve Eq. (18), we can address the following constrained minimization problem:

$$\min \frac{1}{2}\|\mathbf{A}\mathbf{x} - P_Q\mathbf{A}\mathbf{x}\|^2 \quad \text{subject to} \quad \mathbf{x}\in C, \tag{19}$$

where $Q := \{\mathbf{y}\}$ in the signal processing problem, and $Q := \{\mathbf{w} : \|\mathbf{w}-\mathbf{y}+\mathbf{z}\|_2 \le \epsilon\}$ for a small ϵ in the image deblurring problem. In fact, solving problem (19) is equivalent to addressing the split feasibility problem: find $\mathbf{x}\in C$ such that $\mathbf{A}\mathbf{x}\in Q$. We can transform problem (19) into monotone inclusion problem (IP) by setting $A=\partial\delta_C$ and $B=\mathbf{A}^{\top}(\mathbf{A}\mathbf{x}-P_Q\mathbf{A}\mathbf{x})$. Next, we consider the application of the proposed Algorithm 1 in signal processing and image deblurring, and compare its computational performance with some known fixed-step and adaptive algorithms. All programs were written and implemented on a MacBook Air 2023 with 8 GB of memory.

Example 4.1.

Consider recovering the original signal from a degraded signal. Our setup is as follows: the original signal $\mathbf{x}\in\mathbb{R}^n$ is randomly generated, with its values being $\{-1,0,1\}$ and the number of non-zero elements being k. The matrix $\mathbf{A}\in\mathbb{R}^{n\times m}$ is first randomly generated from a normal distribution and then orthogonalized row-wise. The noise vector $\mathbf{z}\in\mathbb{R}^m$ is randomly generated from a normal distribution with a mean of 0 and a standard deviation of 0.01. The obtained degraded signal $\mathbf{y}\in\mathbb{R}^m$ is generated by Eq. (18). We compare the performance of the proposed Algorithm 1 (shortly, Our Alg.) with other fixed-step and adaptive methods in the literature, including Algorithm (2) (shortly, Tseng Alg.), Algorithm (3) (shortly, GT Alg. 1), Algorithm (4) (shortly, GT Alg. 2), Algorithm (5) (shortly, AOM Alg.), Algorithm (6) (shortly, GTV Alg.), Algorithm (7) (shortly, MT Alg.), and Algorithm (8) (shortly, CV Alg.). In all experiments, we test the algorithms by using the following parameters:

- For Our Alg., we take $\lambda_{-1}=\lambda_0=0.2$, $\xi_n=1000/(n+1)^{1.05}$, and $\mu=0.19$.
- For Tseng Alg. [7], we select $\lambda=0.15/L$.
- For GTV Alg. [11], we pick $\alpha_n=0.2$, $\lambda_n=0.15/L$, and $\gamma=1.2$.
- For CV Alg. [15], we choose $\lambda_n=0.2/L$.
- For GT Alg. 1 [8], we take $\alpha_n=0.1/(n+1)$, $\beta_n=0.8(1-\alpha_n)$, $\lambda_0=0.2$, and $\mu=0.19$.
- For GT Alg. 2 [8], we choose $\alpha_n=0.1/(n+1)$, $f(x)=0.9x$, $\lambda_0=0.2$, and $\mu=0.19$.
- For AOM Alg. 1 [10], we pick $\alpha_n=1/(n+1)$, $\theta=0.4$, $\epsilon_n=100/(n+1)^2$, $f(x)=0.9x$, $\xi_n=1000/(n+1)^{1.05}$, $\lambda_0=0.2$, and $\mu=0.19$.
- For MT Alg. [14], we select $\sigma=0.15$, $\rho=\sigma^{-1}$, $\lambda_{-1}=0.2$, and $\mu=0.19$.

We use the mean square error $\mathrm{MSE}=\|\hat{x}-x\|^2/n$ (where $\hat{x}$ is the recovered signal generated by the algorithm at the nth iteration) as the iteration error of the algorithm at the nth step. The stopping criterion for all algorithms is a maximum of 300 iterations. Considering four different dimensions and sparsity levels, the recovery results of all algorithms are shown in Figs. 1, 2, 3, and 4. The trends of their MSE with respect to the number of iterations are shown in Fig. 5. Additionally, Table 1 presents the execution time (in seconds) and the MSE values for all algorithms.

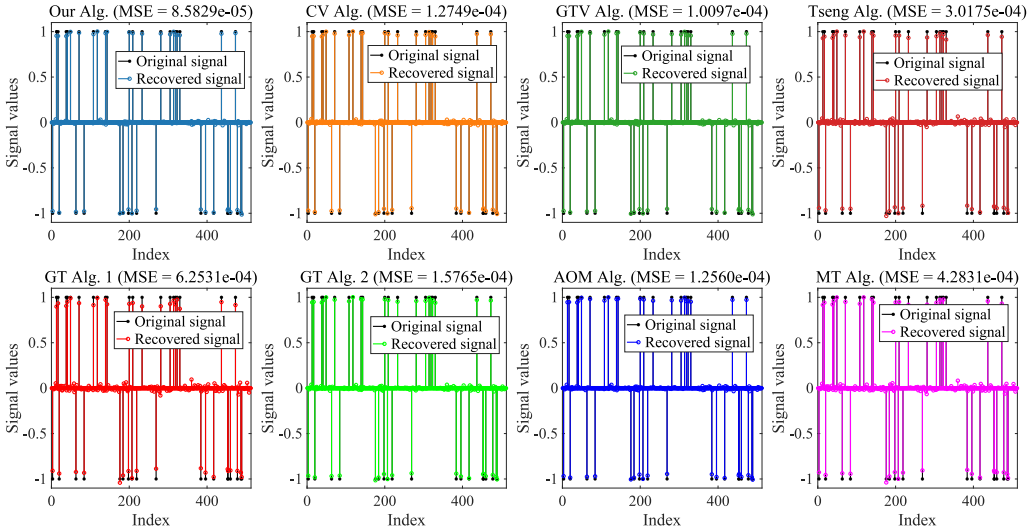

Fig. 1. The comparison of the original signal and the recovered signal for all algorithms when $m = 256$, $n = 512$, and $k = 40$.

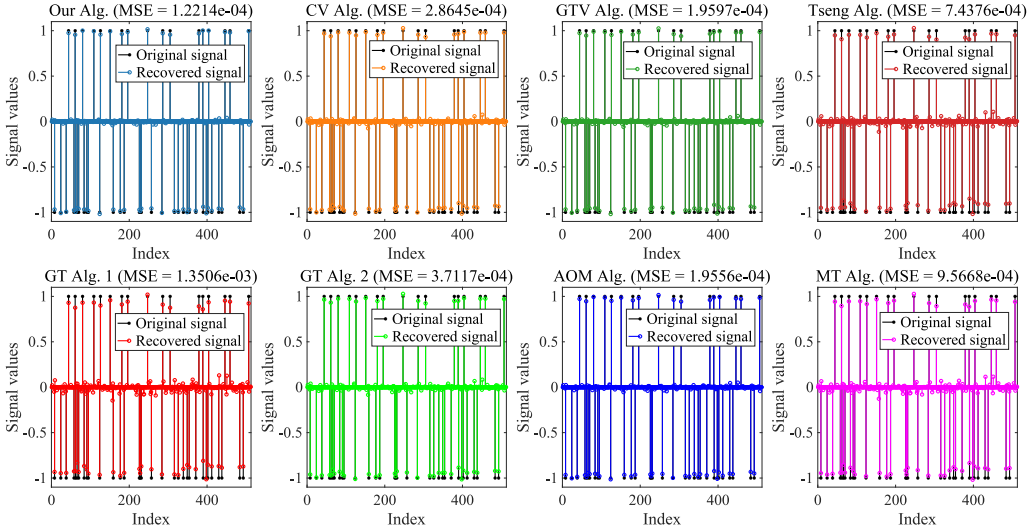

Fig. 2. The comparison of the original signal and the recovered signal for all algorithms when $m = 256$, $n = 512$, and $k = 50$.

Table 1
Numerical results of all algorithms under different cases for Example 4.1.

Algorithms	$m = 256, n = 512$ $k = 40$		$m = 256, n = 512$ $k = 50$		$m = 512, n = 1024$ $k = 60$		$m = 512, n = 1024$ $k = 80$	
	Time (s)	MSE	Time (s)	MSE	Time (s)	MSE	Time (s)	MSE
Our Alg.	0.066	8.583E−05	0.042	1.221E−04	0.162	7.456E−05	0.158	8.198E−05
CV Alg.	0.094	1.275E−04	0.023	2.864E−04	0.099	8.529E−05	0.097	1.323E−04
GTV Alg.	0.056	1.010E−04	0.049	1.960E−04	0.136	7.827E−05	0.140	1.012E−04
Tseng Alg.	0.058	3.017E−04	0.059	7.438E−04	0.202	1.395E−04	0.194	3.316E−04
GT Alg. 1	0.128	6.253E−04	0.068	1.351E−03	0.320	2.917E−04	0.315	6.610E−04
GT Alg. 2	0.115	1.577E−04	0.071	3.712E−04	0.414	9.562E−05	0.315	1.659E−04
AOM Alg.	0.158	1.256E−04	0.072	1.956E−04	0.328	1.018E−04	0.322	1.191E−04
MT Alg.	0.079	4.283E−04	0.044	9.567E−04	0.146	1.761E−04	0.151	4.995E−04

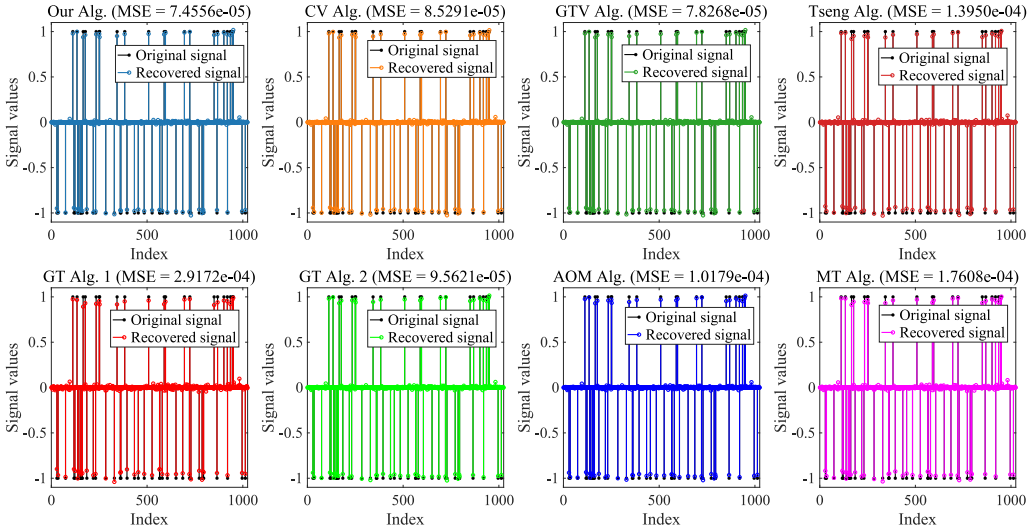

Fig. 3. The comparison of the original signal and the recovered signal for all algorithms when $m = 512$, $n = 1024$, and $k = 60$.

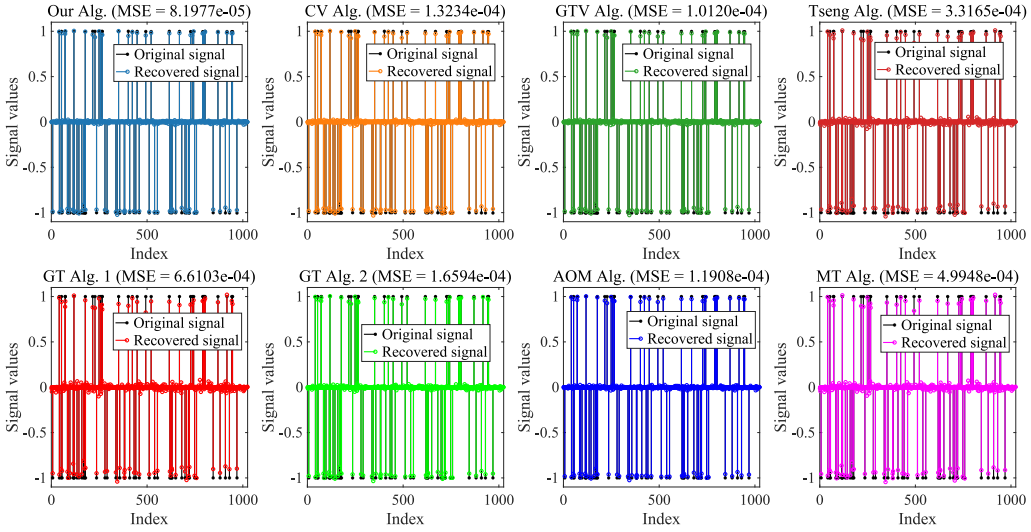

Fig. 4. The comparison of the original signal and the recovered signal for all algorithms when $m = 512$, $n = 1024$, and $k = 80$.

Example 4.2.

Let $\mathbf{A}$ be a convolution matrix and adopt the following settings: the convolution matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ is generated by using a 9×9 Gaussian blur kernel with a standard deviation of 4. The noise vector $\mathbf{z} \in \mathbb{R}^{n\times n}$ is generated from a normal distribution with a mean of 0 and a standard deviation of 10^{-4}. We test four original images $\mathbf{x}$ of size 512×512. The pixel intensity of all original images is scaled into the range between 0 and 1. The degraded image $\mathbf{y}$ is generated using Eq. (18). We use signal-to-noise ratio (SNR in dB), peak signal-to-noise ratio (PSNR in dB), and structural similarity index (SSIM) to evaluate the performance of the algorithms in image recovery. The SNR is calculated as follows:

$$\mathrm{SNR} = 20\log\frac{\|\mathbf{x}\|_2}{\|\mathbf{x}-\hat{\mathbf{x}}\|_2},$$

where $\hat{\mathbf{x}}$ the restored image and $\mathbf{x}$ is the original image. The PSNR and SSIM are calculated using the functions defined in MATLAB. We also select the same algorithms and parameter settings as in Example 4.1 for this example. Typically, higher values of SNR, PSNR, and SSIM indicate better recovery performance.

The recovery results of all algorithms for four different original images are shown in Figs. 6, 7, 8, and 9. The trends of their SNR, PSNR, and SSIM values with the number of iterations are presented in Figs. 10, 11, and 12. Finally, Table 2 summarizes the values of all metrics for all algorithms upon reaching the stopping condition.

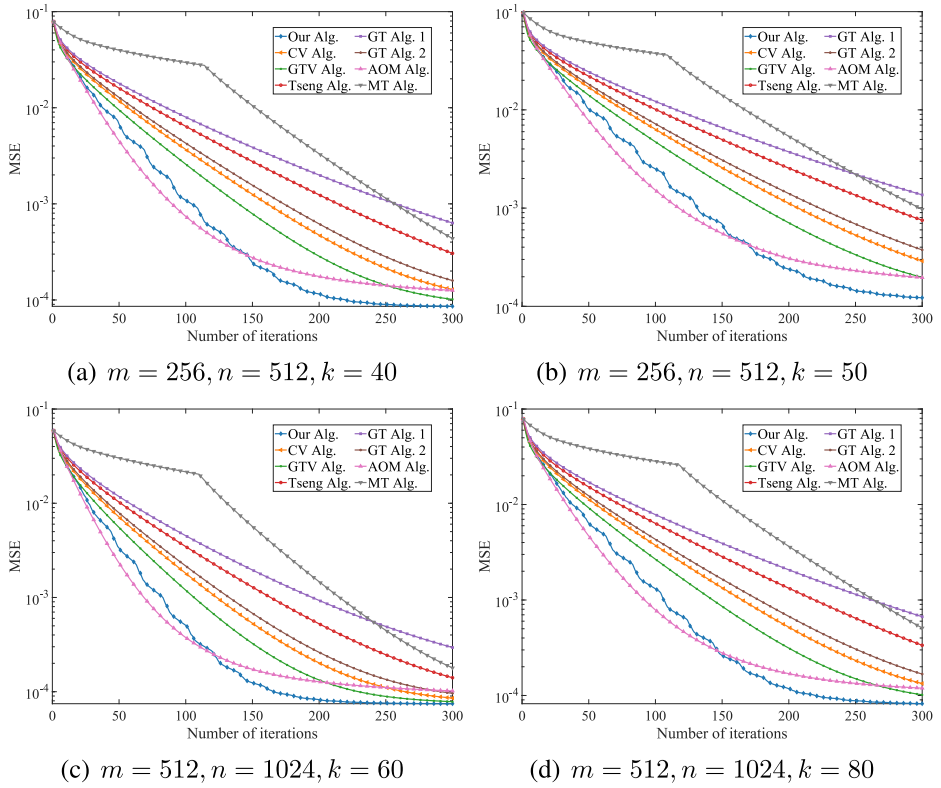

Fig. 5. The variation of MSE for all algorithms under different cases (Example 4.1).

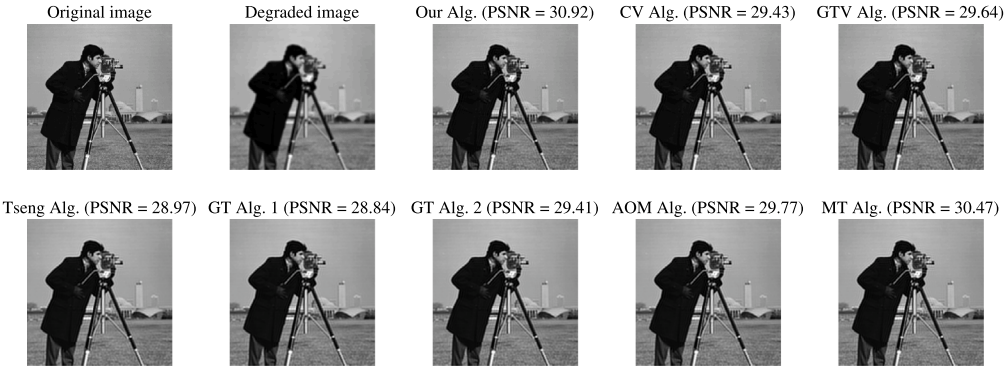

Fig. 6. The recovery results of all algorithms for the image "Cameraman".

Remark 4.3. For the experimental results in Examples 4.1 and 4.2, we have the following observations:

(i) From Figs. 1, 2, 3, 4, 6, 7, 8, and 9, it can be seen that our algorithm can be used to solve signal processing and image deblurring problems.

(ii) From Fig. 5 and Table 1, it is clear that our algorithm outperforms the fixed-step algorithms in [7,11,15] and the adaptive methods in [8,10,14] in terms of MSE. Moreover, Figs. 10, 11, 12, and Table 2 demonstrate that our algorithm also outperforms these fixed-step and adaptive methods in SNR, PSNR, and SSIM.

(iii) It is important to note that the algorithms in [7,11,15] require the prior knowledge of the Lipschitz constant of operator B to perform iterations, while our algorithm and the methods in [8,10,14] can run adaptively without this prior information.

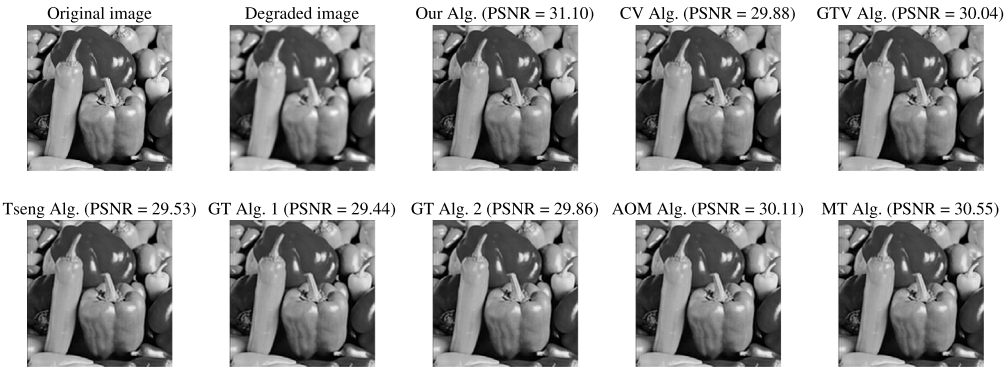

Fig. 7. The recovery results of all algorithms for the image "Peppers".

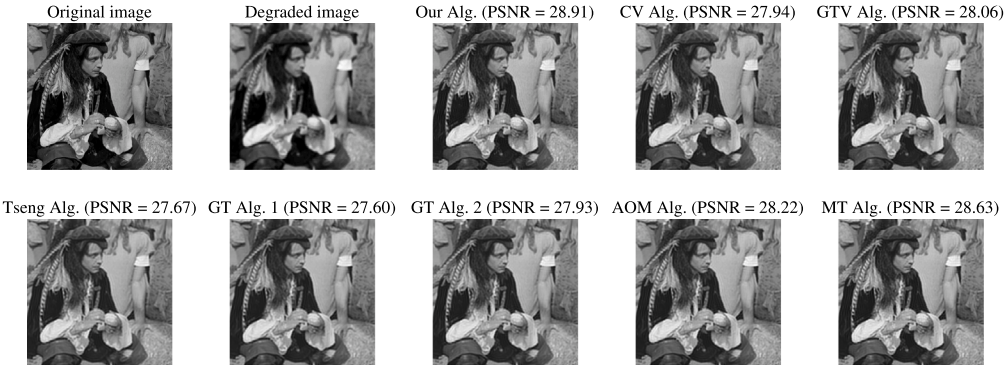

Fig. 8. The recovery results of all algorithms for the image "Pirate".

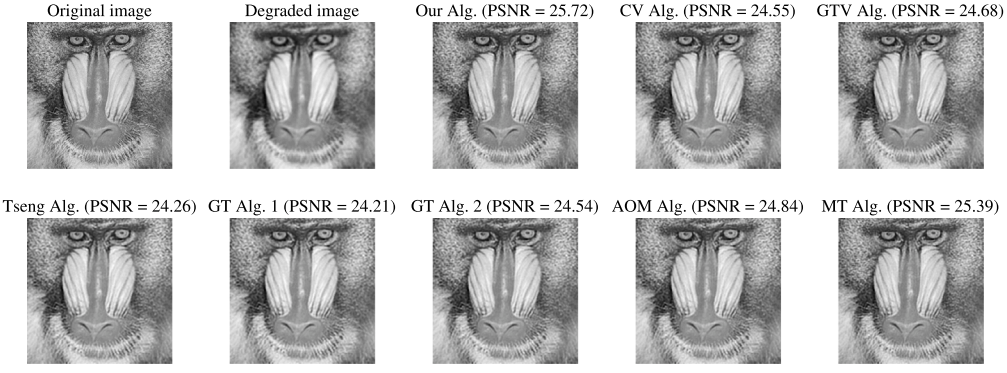

Fig. 9. The recovery results of all algorithms for the image "Mandril".

(iv) As shown in Table 2, Algorithm (8) requires the least amount of time, which is related to the fact that it only requires one forward evaluation and one backward evaluation. Moreover, it can be observed from Tables 1 and 2 that the proposed algorithm has a similar computation time compared to the methods in [7,8,10,11]. These methods require only two evaluations of B and one evaluation of A per iteration, which is consistent with the computational complexity of our algorithm.

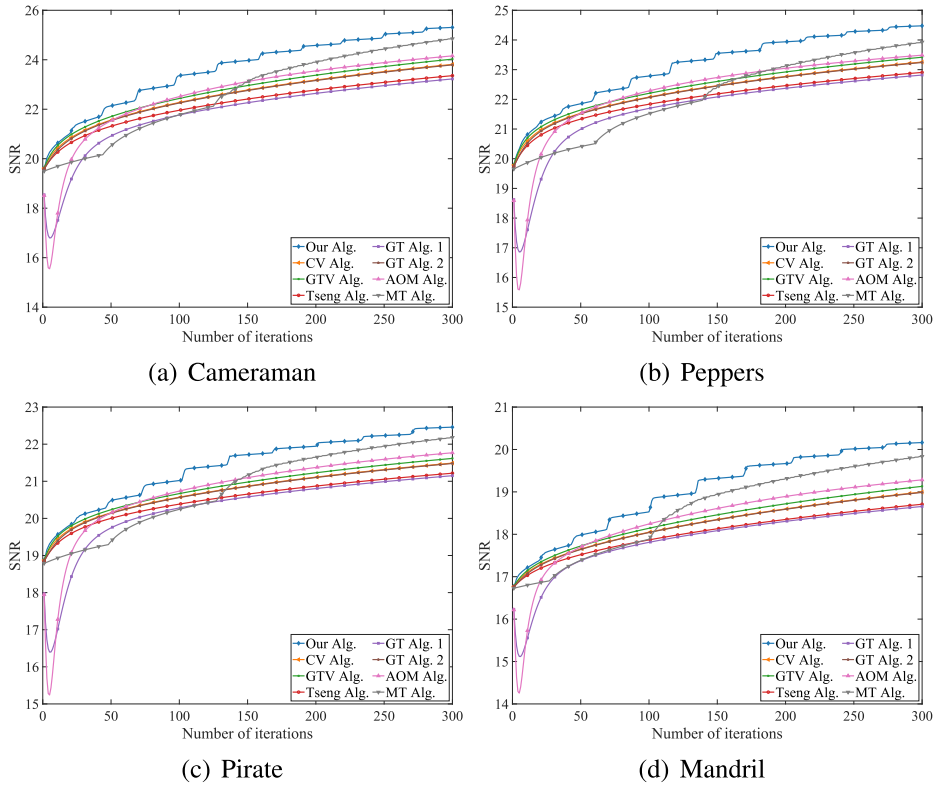

Fig. 10. The variations in SNR with the number of iterations for all algorithms (Example 4.2).

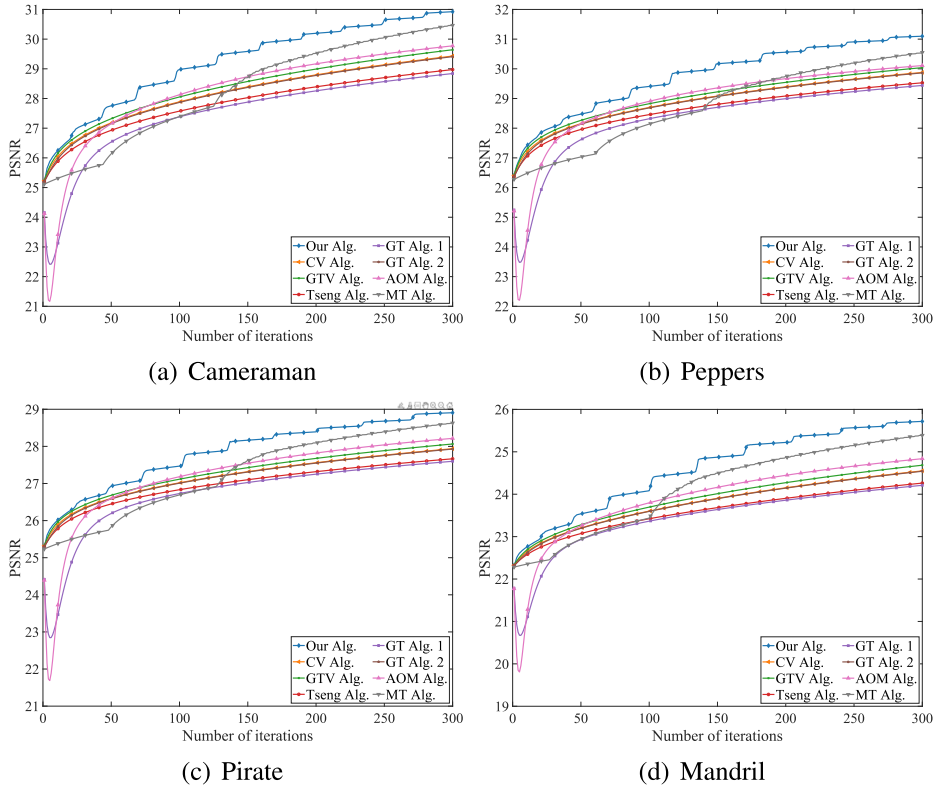

Fig. 11. The variations in PSNR with the number of iterations for all algorithms (Example 4.2).

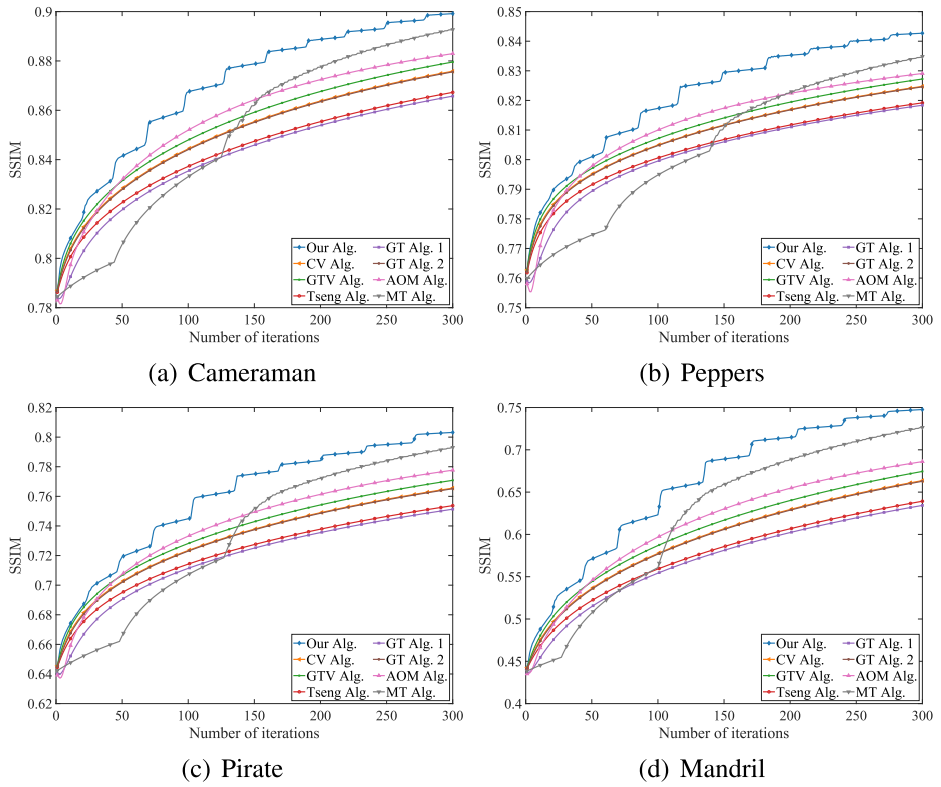

Fig. 12. The variations in SSIM with the number of iterations for all algorithms (Example 4.2).

Table 2
The numerical results of all algorithms with different images in Example 4.2.

Algorithms	Cameraman				Peppers			
	Time (s)	SNR	PSNR	SSIM	Time (s)	SNR	PSNR	SSIM
Our Alg.	16.1596	25.3075	30.9239	0.8992	14.3681	24.4764	31.0982	0.8427
CV Alg.	10.7919	23.8137	29.4302	0.8758	8.6311	23.2562	29.8780	0.8248
GTV Alg.	15.7152	24.0255	29.6419	0.8796	13.6604	23.4182	30.0400	0.8272
Tseng Alg.	14.2786	23.3572	28.9736	0.8673	13.1219	22.9071	29.5288	0.8192
GT Alg. 1	14.4274	23.2247	28.8411	0.8657	13.5009	22.8178	29.4396	0.8184
GT Alg. 2	12.8622	23.7929	29.4094	0.8756	13.8175	23.2410	29.8628	0.8246
AOM Alg.	31.8661	24.1572	29.7736	0.8830	30.9717	23.4874	30.1092	0.8291
MT Alg.	14.5816	24.8490	30.4655	0.8927	14.0328	23.9284	30.5501	0.8348
	Pirate				Mandril			
	Time(s)	SNR	PSNR	SSIM	Time(s)	SNR	PSNR	SSIM
Our Alg.	15.7683	22.4587	28.9067	0.8032	14.9817	20.1633	25.7171	0.7476
CV Alg.	8.8194	21.4901	27.9381	0.7656	8.9056	18.9968	24.5506	0.6636
GTV Alg.	14.1027	21.6163	28.0643	0.7708	14.2182	19.1304	24.6842	0.6745
Tseng Alg.	13.3691	21.2172	27.6652	0.7537	13.9963	18.7083	24.2621	0.6392
GT Alg. 1	13.7963	21.1501	27.5981	0.7513	14.0294	18.6563	24.2101	0.6343
GT Alg. 2	14.7656	21.4791	27.9271	0.7651	14.0032	18.9872	24.5410	0.6628
AOM Alg.	31.1726	21.7679	28.2159	0.7777	32.0870	19.2838	24.8376	0.6861
MT Alg.	14.1984	22.1812	28.6292	0.7930	14.2638	19.8354	25.3892	0.7264

5. Conclusion

In this paper, we introduced a perturbed reflected forward–backward splitting algorithm to solve a monotone inclusion problem such that its step sizes are generated self-adaptively unlike existing reflected forward–backward splitting algorithms where the Lipschitz constant of one of the operators must be known. We obtained weak convergence results, and also demonstrated that our

algorithm is more efficient numerically than some existing splitting algorithms in the literature through experimental results in signal processing and image deblurring. In our future work, we would further explore ways to relax the monotonicity assumption of the single-valued operator.

CRediT authorship contribution statement

Bing Tan: Writing – original draft, Visualization, Software, Investigation. **Yekini Shehu:** Writing – original draft, Validation, Methodology, Investigation, Conceptualization. **Tiexiang Li:** Writing – review & editing. **Xiaolong Qin:** Writing – review & editing, Writing – original draft, Validation, Supervision, Methodology, Investigation, Conceptualization.

Ethical approval and consent to participate

All the authors gave the ethical approval and consent to participate in this article.

Consent for publication

All the authors gave consent for the publication of identifiable details to be published in the journal and article.

Code availability

The MATLAB codes employed to run the numerical experiments are available on request.

Funding

B. Tan acknowledges support from the Natural Science Foundation of Chongqing, China (No. CSTB2024NSCQ-MSX0354), the Fundamental Research Funds for the Central Universities, China (No. SWU-KQ24052), and the Natural Science Foundation of China (No. 12471473). T. Li was partially supported by the National Natural Science Foundation of China under Grant No. 12371377, the Jiangsu Provincial Scientific Research Center of Applied Mathematics, China under Grant No. BK20233002 and the SIMIS under Grant SIMIS-ID-2024-LG.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors are deeply grateful to the reviewers for their thorough reading of the manuscript and for providing constructive suggestions, which greatly helped us improve the quality of the initial manuscript.

Data availability

Data will be made available on request.

References

[1] Combettes PL, Wajs V. Signal recovery by proximal forward–backward splitting. Multiscale Model Simul 2005;4:1168–200.
[2] Lorenz DA, Pock T. An inertial forward–backward algorithm for monotone inclusions. J Math Imaging Vision 2015;51:311–25.
[3] Boţ RI, Sedlmayer M, Vuong PT. A relaxed inertial forward–backward-forward algorithm for solving monotone inclusions with application to GANs. J Mach Learn Res 2023;24:1–37.
[4] Lions PL, Mercier B. Splitting algorithms for the sum of two nonlinear operators. SIAM J Numer Anal 1979;16:964–79.
[5] Rockafellar RT. Monotone operators and the proximal point algorithm. SIAM J Control Optim 1976;14:877–98.
[6] Chen GH, Rockafellar RT. Convergence rates in forward–backward splitting. SIAM J Optim 1997;7:421–44.
[7] Tseng P. A modified forward–backward splitting method for maximal monotone mappings. SIAM J Control Optim 2000;38:431–46.
[8] Gibali A, Thong DV. Tseng type methods for solving inclusion problems and its applications. Calcolo 2018;55:49.
[9] Polyak B. Some methods of speeding up the convergence of iteration methods. USSR Comput Math Math Phys 1964;4:1–17.
[10] Alakoya TO, Ogunsola OJ, Mewomo OT. An inertial viscosity algorithm for solving monotone variational inclusion and common fixed point problems of strict pseudocontractions. Bol Soc Mat Mex 2023;29:31.
[11] Gibali A, Thong DV, Vinh NT. Three new iterative methods for solving inclusion problems and related problems. Comp Appl Math 2020;39:187.
[12] Cholamjiak P, Hieu DV, Cho YJ. Relaxed forward–backward splitting methods for solving variational inclusions and applications. J Sci Comput 2021;88:85.
[13] Wang Z, Lei Z, Long X, Chen Z. A modified Tseng splitting method with double inertial steps for solving monotone inclusion problems. J Sci Comput 2023;96:92.
[14] Malitsky Y, Tam MK. A forward–backward splitting method for monotone inclusions without cocoercivity. SIAM J Optim 2020;30:1451–72.

[15] Cevher V, Vũ BC. A reflected forward–backward splitting method for monotone inclusions involving Lipschitzian operators. Set-Valued Var Anal 2021;29:163–74.
[16] Malitsky Y. Projected reflected gradient methods for monotone variational inequalities. SIAM J Optim 2015;25:502–20.
[17] Brézis H. Operateurs maximaux monotones, Chapitre II. North-Holland. Math Stud 1973;5:19–51.
[18] Opial Z. Weak convergence of the sequence of successive approximations for nonexpansive mappings. Bull Amer Math Soc 1967;73:591–7.
[19] Lou Y, Zeng T, Osher S, Xin J. A weighted difference of anisotropic and isotropic total variation model for image processing. SIAM J Imaging Sci 2015;8:1798–823.
[20] Dong B, Li J, Shen Z. X-ray CT image reconstruction via wavelet frame-based regularization and radon domain inpainting. J Sci Comput 2013;54:333–49.
[21] Lustig M, Donoho D, Pauly JM. Sparse MRI: the application of compressed sensing for rapid MR imaging. Magn Reson Med 2007;58:1182–95.