Outer Reflected Forward-Backward Splitting Algorithm with Inertial Extrapolation Step

Yekini Shehu[*], Lateef O. Jolaoso[†], C. C. Okeke[‡], Renqi Xu[§]

May 12, 2026

Abstract

This paper studies an outer reflected forward-backward splitting algorithm with an inertial step to find a zero of the sum of three monotone operators composing the maximal monotone operator, Lipschitz monotone operator, and a cocoercive operator in real Hilbert spaces. One of the interesting features of the proposed method is that both the Lipschitz monotone operator and the cocoercive operator are computed explicitly each with one evaluation per iteration. We obtain weak and strong convergence results under some easy-to-verify assumptions. We also obtain a non-asymptotic $O(1/n)$ convergence rate of our proposed algorithm in a non-ergodic sense. We finally give some numerical illustrations arising from compressed sensing and image processing and show that our proposed method is effective and competitive with other related methods in the literature.

Keywords: three operator splitting; inertial extrapolation step; weak and strong convergence; non-asymptotic convergence rate.

2010 MSC classification: 90C25, 90C30, 90C60, 68Q25, 49M25, 90C22

1 Introduction

Suppose H is a real Hilbert space and $A : H \rightrightarrows H$ is a set-valued operator, $B, C : H \to H$ are single-valued operators. Let us consider the following inclusion problem:

$$\text{Find } x^* \in H \text{ such that } 0 \in (A + B + C)(x^*). \tag{1}$$

[*]College of Mathematics and Computer Science, Zhejiang Normal University, Jinhua 321004, People's Republic of China; e-mail: yekini.shehu@zjnu.edu.cn

[†]School of Mathematical Sciences, University of Southampton, SO17 1BJ, United Kingdom; Department of Mathematics and Applied Mathematics, Sefako Makgatho Health Sciences University, P.O. Box 94 Medunsa 0204, Pretoria, South Africa; e-mail: l.o.jolaoso@soton.ac.uk

[‡]School of Mathematics, University of the Witwatersrand, Johannesburg, South Africa; e-mail:chibueze.okeke87@yahoo.com

[§]College of Mathematics and Computer Science, Zhejiang Normal University, Jinhua, 321004, People's Republic of China; e-mail: xurenqi@zjnu.edu.cn

We assume that the set of solutions to inclusion problem (1) is denoted by $(A+B+C)^{-1}(0)$. Inclusion problem (1) can be considered as a central problem in continuous optimization around several known mathematical models such as optimization problems, variational inequalities [25, 26, 27, 28], split feasibility problems with applications signal and image processing, machine learning [15, 19, 23, 38] and so on. Some models which motivate the studying of the problem can be found in [20, 32] and the references cited therein.

In the case when the inclusion problem (1) involves two operators, some notable methods are the Forward-Backward Splitting Algorithm [3], the Douglas-Rachford Splitting Algorithm [30] and the Forward-Backward-Forward Splitting Algorithm [7, 16, 22, 30, 36, 44]. Some variants of these three splitting algorithms have also been proposed in the literature (see, for example, [1, 8, 9, 10, 12, 14, 17, 21, 38, 40, 45]).

Recently, three-operator splitting algorithms have been developed to solve inclusion problem (1) [2, 20, 29, 39, 41, 42]. Davis and Yin [20] studied the inclusion problem (1) and introduced the following three-operator splitting algorithm:

$$x_{n+1} = (1-\lambda_n)x_n + \lambda_n T x_n, \quad x_0 \in H, \tag{2}$$

where $T := J_{\gamma A}(2J_\gamma - I - \gamma C J_{\gamma B}) + I - J_{\gamma B}$, and $\gamma > 0$ and $\{\lambda_n\}$ are suitable parameters. Under the hypotheses of maximal monotonicity of A, B, and the $\frac{1}{L_2}$-cocoercivity of C, the authors in [20] proved that the sequence $\{x_n\}$ generated by (2) converges weakly to a solution of inclusion problem (1), provided by some suitable choices of parameters γ and $\{\lambda_n\}$ depending on the cocoercive constant of C. An inexact version of the scheme (2) has also been recently investigated by Zong et al. in [48].

Malitsky and Tam [32] studied the problem (1) when A is maximal monotone, B is monotone and L_1-Lipschitz continuous, C is $\frac{1}{L_2}$-cocoercive, and proposed the following Forward-Reflected-Backward Splitting Algorithm with a fixed stepsize:

$$x_{n+1} = J_{\lambda A}(x_n - 2\lambda B(x_n) + \lambda B(x_{n-1}) - \lambda C(x_n)), \quad x_0, x_{-1} \in H. \tag{3}$$

Malitsky and Tam [32] proved that if $\lambda \in \left(0, \frac{2}{4L_1+L_2}\right)$, then sequence $\{x_n\}$ generated by (3) converges weakly to a point in $(A+B+C)^{-1}(0)$. Csetnek et al. [18] proposed the following shadow Douglas-Rachford Splitting Algorithm to solve inclusion problem (1) when $C = 0$:

$$x_{n+1} = J_{\lambda A}(x_n - \lambda B(x_n)) - \lambda(B(x_n) - B(x_{n-1})). \tag{4}$$

The algorithm (4) is shown in [18] to arise naturally from nonstandard discretization of a continuous dynamical system associated with Douglas-Rachford Splitting Algorithm and converges weakly to a solution of (1) when $\lambda \in \left(0, \frac{1}{3L}\right)$ with L being the Lipschitz constant of Lipschitz monotone operator B.

Motivated by [18, 32], Yu et al. [47] proposed an outer reflected forward-backward splitting algorithm for solving inclusion problem (1) in which A is a maximal monotone operator, B is a Lipschitz monotone operator, and C a cocoercive operator:

$$x_{n+1} = J_{\lambda A}(x_n - \lambda B(x_n) - \lambda C(x_n)) - \lambda(B(x_n) - B(x_{n-1})). \tag{5}$$

Weak convergence analysis of (5) was obtained under some mild conditions on the iterative parameters and applications to composite monotone inclusions involving monotone Lipschitzian operator, cocoercive operator, and the parallel sum of operators are given.

The inertial extrapolation step which was introduced by Polyak [37] in his so-called heavy ball method, has been recently fused into Forward-Backward Splitting Algorithm, thus helping to speedily increase the rate of convergence. Moudafi and Oliny [33] added a single valued $\frac{1}{L_2}$-cocoercive operator C to the inertial proximal point algorithm:

$$\begin{cases} y_n = x_n + \beta_n(x_n - x_{n-1}) \\ x_{n+1} = J_{\lambda_n A}(y_n - \lambda_n C x_n). \end{cases} \tag{6}$$

It was proved in [33, Theorem 2.1] that (6) converges weakly to solution of (1), provided $\lambda_n < \frac{2}{L_2}$, where $\sum_{n=1}^{\infty} \beta_n \|x_n - x_{n-1}\|^2 < +\infty$ and $\beta_n \in [0, 1)$. Kindly see also [31] for another version of inertial Forward-Backward Splitting Algorithm. In several other inertial Forward-Backward Splitting Algorithms, the convergence analysis is obtained when $\beta_n \in [0, 1)$ (see, for example, [4, 6, 11, 24, 31, 35, 46]).

Our Contributions: Motivated by the results in [18, 32, 47], our contributions in this paper are given as:

- we propose an outer reflected forward-backward splitting algorithm with an inertial step to finding a zero of the sum of three monotone operators composing of the maximal monotone operator, Lipschitz monotone operator, and a cocoercive operator in real Hilbert spaces. One of the interesting features of the proposed method is that both the Lipschitz monotone operator and the cocoercive operator are computed explicitly with one evaluation per iteration;
- we obtain weak and strong convergence results under some easy-to-verify assumptions. We also obtain a non-asymptotic $O(1/n)$ convergence rate of our proposed algorithm in a non-ergodic sense;
- we finally give some numerical illustrations arising from compressed sensing and image processing, and show that our proposed method is effective and competitive with other related methods in the literature.

2 Preliminaries

Let $A : H \rightrightarrows H$ be a multi-valued mapping in a real Hilbert space H. The graph of A is a subset of $H \times H$ defined by

$$\text{Graph}(A) = \{(x, u) \in H \times H : x \in H,\ u \in A(x)\}.$$

The operator A is called:

(i) monotone if

$$\langle u - v, x - y\rangle \geq 0, \quad \forall x, y \in H, \ u \in A(x), \quad v \in A(y);$$

(ii) maximal monotone if it is monotone and its graph is not properly contained in the graph of any other monotone operator;

(iii) L_1-Lipschitz continuous if there exists a number $L_1 > 0$, such that

$$\|u - v\| \leq L_1 \|x - y\|, \quad \forall x, y \in H, \quad u \in A(x), \quad v \in A(y).$$

(iv) $\frac{1}{L_2}$-cocoercive (or $\frac{1}{L_2}$-inverse strongly monotone) if there exists $L_2 > 0$ such that

$$\langle u - v, x - y\rangle \geq \frac{1}{L_2}\|u - v\|^2, \quad \forall x, y \in H, \quad u \in A(x), \ v \in A(y);$$

(v) η-strongly monotone if there exists $\eta > 0$ such that

$$\langle u - v, x - y\rangle \geq \eta\|x - y\|^2, \quad \forall x, y \in H, \quad u \in A(x), \ v \in A(y);$$

Remark here that the maximal monotone of operator $A : H \rightrightarrows H$ is equivalent to the following characteristic property: if for any $(x, u) \in H \times H$, $\langle u - v, x - y\rangle \geq 0$ for all $(y, v) \in \text{Graph}(A)$ then $u \in A(x)$. Beside, if A is $\frac{1}{L_2}$-cocoercive, then A is L_2-Lipschitz continuous. Recall the resolvent of a maximal monotone operator $A : H \rightrightarrows H$ is a single-valued mapping $J_A : H \to H$, defined by $J_{\lambda A}(x) = (I + \lambda A)^{-1}(x), \quad x \in H, \lambda > 0$ where I stands for the identity operator on H.

We need the following lemmas in our convergence analysis.

Lemma 2.1. *([3]) Let $\{x_n\}$ be a sequence in H. Then $\{x_n\}$ converges weakly if and only if it is bounded and possesses at most one weak cluster point*

Lemma 2.2. *([34] Let D be a nonempty subset of H and $\{x_n\}$ be a sequence in H such that the following two conditions hold:*

(i) For every $x^* \in D$, $\lim_{n\to\infty} \|x_n - x^*\|$ exists;

(ii) Every sequential weak cluster point of $\{x_n\}$ in D.

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

Lemma 2.3. *The following identities hold for all $\forall a, b, c \in H$ and $\alpha \in \mathbb{R}$:*
(i) $2\langle a - b, c - a\rangle = \|c - b\|^2 - \|a - b\|^2 - \|c - a\|^2$;
(ii) $\|(1 + \alpha)a - \alpha b\|^2 = (1 + \alpha)\|a\|^2 - \alpha\|b\|^2 + \alpha(1 + \alpha)\|a - b\|^2$.

3 Main Results

In this section, we present our iterative method and prove its weak and strong convergence results alongside the non-asymptotic convergence rate. In the sequel, we assume that the following conditions are satisfied:

Assumption 3.1. (i) $A : H \rightrightarrows H$ is set-valued maximal monotone;

(ii) $B : H \to H$ is a single-valued monotone and L_1-Lipschitz continuous;

(iii) $C : H \to H$ is $\frac{1}{L_2}$-cocoercive.

(iv) The solution set $(A + B + C)^{-1}(0)$ of inclusion problem (1) is nonempty.

We furthermore assume the following conditions on parameter $\lambda > 0$ and the inertial factor β are satisfied.

Assumption 3.2.

(i) $0 < \lambda < \min\left\{\frac{1}{\sqrt{L_1 L_2}}, \frac{2}{9(L_1 + L_2(1 + 2\beta - \beta^2))}\right\}$;

(ii) $0 \leq \beta \leq 1$.

We now present our proposed method to solve the inclusion problem (1) below.

Algorithm 3.3.

Suppose Assumption 3.1 and Assumption 3.2 are fulfilled. Given $x_{-1}, x_0 \in H$, compute

$$\begin{cases} z_n = x_n + \beta(x_n - x_{n-1}) \\ x_{n+1} = J_{\lambda A}\left(x_n - \lambda(Bx_n + Cz_n)\right) - \lambda(Bx_n - Bx_{n-1}). \end{cases} \tag{7}$$

Remark 3.4. (a) In our proposed Algorithm 3.3, the choice $\beta = 1$ is possible and this is not possible in the method proposed in [24] where the inertial factor is bounded away from 1. Furthermore, Algorithm 3.3 reduces to the method proposed in [47, Theorem 3.1] when $\beta = 0$.

(b) Our Algorithm 3.3 reduces to the shadow Douglas-Rachford splitting method proposed in [18] when $\beta = 0$ and $C = 0$.

We present the convergence analysis of our proposed Algorithm 3.3 below.

Lemma 3.5. *Suppose Assumption 3.1 and Assumption 3.2 are fulfilled and let $\{x_n\}$ be generated by Algorithm 3.3. Then we have $\sum_{n=1}^{\infty} \|Cz_n - Cx^*\|^2 < \infty$ for any $x^* \in (A + B + C)^{-1}(0)$.*

Proof. Using the definition of resolvent and Algorithm 3.3, we have

$$x_{n+1} + \lambda(Bx_n - Bx_{n-1}) - x_n + \lambda(Bx_n + Cz_n) \in -\lambda A(x_{n+1} + \lambda(Bx_n - Bx_{n-1})). \quad (8)$$

Since $0 \in (A + B + C)x^*$, we have that

$$\lambda(B + C)x^* \in -\lambda Ax^*. \quad (9)$$

Using the monotonicity of A, we have that (noting (8) and (9))

$$\begin{aligned} &\langle x_{n+1} + \lambda(Bx_n - Bx_{n-1}) - x_n + \lambda(Bx_n + Cz_n) \\ &-\lambda(B + C)x^*, x^* - x_{n+1} - \lambda(Bx_n - Bx_{n-1})\rangle \geq 0. \end{aligned} \quad (10)$$

Thus,

$$\begin{aligned} &\langle x_{n+1} - x_n + \lambda(Bx_n - Bx_{n-1}) + \lambda(Bx_n - Bx^*) \\ &+\lambda(Cz_n - Cx^*), x^* - x_{n+1} - \lambda(Bx_n - Bx_{n-1})\rangle \geq 0. \end{aligned}$$

Re-arranging, we have

$$\begin{aligned} 0 \leq\ &\langle x_{n+1} - x_n, x^* - x_{n+1}\rangle + \langle x_{n+1} - x_n, \lambda(Bx_{n-1} - Bx_n)\rangle \\ &+\langle \lambda Bx_n - \lambda Bx_{n-1}, x^* - x_{n+1}\rangle \\ &+\langle \lambda(Bx_n - Bx_{n-1}), \lambda(Bx_{n-1} - Bx_n)\rangle + \langle \lambda(Bx_n - Bx^*), x^* - x_{n+1}\rangle \\ &+\langle \lambda(Bx_n - Bx^*), \lambda(Bx_{n-1} - Bx_n)\rangle + \langle \lambda(Cz_n - Cx^*), x^* - z_n\rangle \\ &+\langle \lambda(Cz_n - Cx^*), z_n - x_{n+1}\rangle + \langle \lambda(Cz_n - Cx^*), \lambda(Bx_{n-1} - Bx_n)\rangle. \end{aligned} \quad (11)$$

By Lemma 2.3 (i), we have

$$2\langle x_{n+1} - x_n, x^* - x_{n+1}\rangle = \|x_n - x^*\|^2 - \|x_{n+1} - x_n\|^2 - \|x_{n+1} - x^*\|^2. \quad (12)$$

Observe also that

$$2\langle x_{n+1} - x_n, \lambda(Bx_{n-1} - Bx_n)\rangle \leq \lambda L_1 \left(\|x_{n+1} - x_n\|^2 + \|x_{n-1} - x_n\|^2\right),$$

$$\begin{aligned} \langle \lambda Bx_n - \lambda Bx_{n-1}, x^* - x_{n+1}\rangle &= \langle \lambda Bx_n - \lambda Bx_{n-1}, x_n - x_{n+1}\rangle \\ &\quad +\langle \lambda Bx_{n-1} - \lambda Bx^*, x_n - x^*\rangle \\ &\quad +\langle \lambda Bx^* - \lambda Bx_n, x_n - x^*\rangle \end{aligned} \quad (13)$$

and

$$\begin{aligned} 2\langle \lambda(Bx_n - Bx^*), \lambda(Bx_{n-1} - Bx_n)\rangle &= 2\langle \lambda(Bx_n - Bx_{n-1}), \lambda(Bx^* - Bx_n)\rangle \\ &= \|\lambda(Bx_{n-1} - Bx^*)\|^2 - \|\lambda(Bx_n - Bx_{n-1})\|^2 \\ &\quad -\|\lambda(Bx_n - Bx^*)\|^2. \end{aligned} \quad (14)$$

Substituting (12), (13) and (14) into (11), we obtain

$$\begin{aligned} 0 \leq\ &\|x_n - x^*\|^2 - \|x_{n+1} - x_n\|^2 - \|x_{n+1} - x^*\|^2 + 2\langle x_{n+1} - x_n, \lambda(Bx_{n-1} - Bx_n)\rangle \\ &+2\langle \lambda(Bx_n - Bx_{n-1}), x_n - x_{n+1}\rangle + 2\langle \lambda(Bx_{n-1} - Bx^*), x_n - x^*\rangle \end{aligned}$$

$$
\begin{aligned}
&+2\langle\lambda(Bx^*-Bx_n),x_n-x^*\rangle+2\langle\lambda(Bx_n-Bx_{n-1}),\lambda(Bx_{n-1}-Bx_n)\rangle\\
&+2\langle\lambda(Bx_n-Bx^*),x^*-x_{n+1}\rangle+\|\lambda(Bx_{n-1}-Bx^*)\|^2\\
&-\|\lambda(Bx_n-Bx_{n-1})\|^2-\|\lambda(Bx_n-Bx^*)\|^2+2\langle\lambda(Cz_n-Cx^*),x^*-z_n\rangle\\
&+2\langle\lambda(Cz_n-Cx^*),z_n-x_{n+1}\rangle+2\langle\lambda(Cz_n-Cx^*),\lambda(Bx_{n-1}-Bx_n)\rangle. \qquad (15)
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
&\|x_{n+1}-x^*\|^2+2\langle\lambda(Bx_n-Bx^*),x_{n+1}-x^*\rangle+\|\lambda(Bx_n-Bx^*)\|^2\\
&\leq\|x_n-x^*\|^2+2\langle\lambda(Bx_{n-1}-Bx^*),x_n-x^*\rangle+\|\lambda(Bx_{n-1}-Bx^*)\|^2\\
&\ -3\|\lambda(Bx_n-Bx_{n-1})\|^2+4\langle\lambda(Bx_n-Bx_{n-1}),x_n-x_{n+1}\rangle\\
&\ -\|x_{n+1}-x_n\|^2-2\langle\lambda(Bx^*-Bx_n),x^*-x_n\rangle\\
&\ +2\langle\lambda(Cz_n-Cx^*),x^*-z_n\rangle+2\langle\lambda(Cz_n-Cx^*),z_n-x_{n+1}\rangle\\
&\ +2\langle\lambda(Cz_n-Cx^*),\lambda(Bx_{n-1}-Bx_n)\rangle. \qquad (16)
\end{aligned}
$$

Since B is Lipschitz continuous

$$
2\langle\lambda(Bx_{n-1}-Bx^*),x_n-x^*\rangle\leq\lambda L_1\left(\|x_{n-1}-x^*\|^2+\|x_n-x^*\|^2\right), \qquad (17)
$$

and

$$
2\langle x_{n+1}-x_n,\lambda(Bx_{n-1}-Bx_n)\rangle\leq\lambda L_1\left(\|x_{n+1}-x_n\|^2+\|x_{n-1}-x_n\|^2\right). \qquad (18)
$$

By Young's inequality, we have

$$
2\langle\lambda(Bx_n-Bx_{n-1}),x_n-x_{n+1}\rangle\ \leq\ 3\|\lambda(Bx_n-Bx_{n-1})\|^2+\frac{1}{3}\|x_{n+1}-x_n\|^2. \qquad (19)
$$

If we apply (17)-(19) in (16) and using the fact that $-2\langle\lambda(Bx^*-Bx_n),x^*-x_n\rangle\leq 0$, we obtain

$$
\begin{aligned}
\|(x_{n+1}+\lambda Bx_n)-(x^*+\lambda Bx^*)\|^2\ \leq\ &\|(x_n+\lambda Bx_{n-1})-(x^*+\lambda Bx^*)\|^2\\
&-3\|\lambda(Bx_n-Bx_{n-1})\|^2-\|x_{n+1}-x_n\|^2\\
&+\lambda L_1\left(\|x_{n+1}-x_n\|^2+\|x_{n-1}-x_n\|^2\right)\\
&+3\|\lambda(Bx_n-Bx_{n-1})\|^2+\frac{1}{3}\|x_{n+1}-x_n\|^2\\
&+2\langle\lambda(Cz_n-Cx^*),x^*-z_n\rangle\\
&+2\langle\lambda(Cz_n-Cx^*),z_n-x_{n+1}\rangle\\
&+2\langle\lambda(Cz_n-Cx^*),\lambda(Bx_{n-1}-Bx_n)\rangle.
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
&\|(x_{n+1}+\lambda Bx_n)-(x^*+\lambda Bx^*)\|^2+\left(\frac{2}{3}-\lambda L_1\right)\|x_{n+1}-x_n\|^2\\
&\leq\|(x_n+\lambda Bx_{n-1})-(x^*+\lambda Bx^*)\|^2+\lambda L_1\|x_n-x_{n-1}\|^2\\
&\ +2\langle\lambda(Cz_n-Cx^*),x^*-z_n\rangle+2\langle\lambda(Cz_n-Cx^*),z_n-x_{n+1}\rangle\\
&\ +2\langle\lambda(Cz_n-Cx^*),\lambda(Bx_{n-1}-Bx_n)\rangle. \qquad (20)
\end{aligned}
$$

Using $\frac{1}{L_2}-$ cocoercivity of C, we have

$$\langle Cz_n - Cx^*, x^* - z_n\rangle \leq \frac{-1}{L_2}\|Cz_n - Cx^*\|^2 \tag{21}$$

and by Cauchy-Schwarz inequality

$$\langle Cz_n - Cx^*, z_n - x_{n+1}\rangle \leq \frac{1}{6L_2}\|Cz_n - Cx^*\|^2 + \frac{3L_2}{2}\|x_{n+1} - z_n\|^2 \tag{22}$$

Since B is L_2-Lipschitz continuous, we have

$$\begin{aligned}
&2\langle\lambda(Cz_n - Cx^*), \lambda(Bx_{n-1} - Bx_n)\rangle \\
&\leq 2\lambda L_1\|\lambda(Cz_n - Cx^*)\|\|x_{n-1} - x_n\| \\
&\leq \lambda L_1\|x_{n-1} - x_n\|^2 + \lambda^3 L_1\|Cz_n - Cx^*\|^2 \\
&\leq \lambda L_1\|x_{n-1} - x_n\|^2 + \frac{\lambda}{L_2}\|Cz_n - Cx^*\|^2,
\end{aligned} \tag{23}$$

since $\lambda^2 L_1 L_1 < 1$ by Assumption 3.2 (i). Using (21), (22) and (23) in (20), we have

$$\begin{aligned}
&\|(x_{n+1} + \lambda Bx_n) - (x^* + \lambda Bx^*)\|^2 + \left(\frac{2}{3} - \lambda L_1\right)\|x_{n+1} - x_n\|^2 \\
&\leq \|(x_n + \lambda Bx_{n-1}) - (x^* + \lambda Bx^*)\|^2 + \lambda L_1\|x_n - x_{n-1}\|^2 \\
&-\frac{2\lambda}{L_2}\|Cz_n - Cx^*\|^2 + \frac{\lambda}{3L_2}\|Cz_n - Cx^*\|^2 + 3\lambda L_2\|x_{n+1} - z_n\|^2 \\
&+\lambda L_1\|x_{n-1} - x_n\|^2 + \frac{\lambda}{L_2}\|Cz_n - Cx^*\|^2.
\end{aligned} \tag{24}$$

By Lemma 2.3 (ii), we have

$$\begin{aligned}
\lambda L_2\|x_{n+1} - z_n\|^2 &= \lambda L_2\|x_n + \beta(x_n - x_{n-1}) - x_{n+1}\|^2 \\
&= \lambda L_2\|(1+\beta)(x_n - x_{n+1}) - \beta(x_{n-1} - x_{n+1})\|^2 \\
&= \lambda L_2(1+\beta)\|x_{n+1} - x_n\|^2 - \lambda L_2\beta\|x_{n+1} - x_{n-1}\|^2 \\
&\quad +\lambda L_2\beta(1-\beta)\|x_n - x_{n-1}\|^2.
\end{aligned} \tag{25}$$

Putting (25) into (24) gives

$$\begin{aligned}
&\|(x_{n+1} + \lambda Bx_n) - (x^* + \lambda Bx^*)\|^2 + \left(\frac{2}{3} - \lambda L_1\right)\|x_{n+1} - x_n\|^2 \\
&\leq \|(x_n + \lambda Bx_{n-1}) - (x^* + \lambda Bx^*)\|^2 + 2\lambda L_1\|x_n - x_{n-1}\|^2 \\
&+3\lambda L_2(1+\beta)\|x_{n+1} - x_n\|^2 - 3\lambda L_2\beta\|x_{n+1} - x_{n-1}\|^2 \\
&+3\lambda L_2\beta(1-\beta)\|x_n - x_{n-1}\|^2 - \frac{2\lambda}{3L_2}\|Cz_n - Cx^*\|^2.
\end{aligned} \tag{26}$$

Hence,

$$\|(x_{n+1} + \lambda Bx_n) - (x^* + \lambda Bx^*)\|^2 + \left(\frac{2}{3} - \lambda L_1 - 3\lambda L_2(1+\beta)\right)\|x_{n+1} - x_n\|^2$$

$$\leq \|(x_n + \lambda B x_{n-1}) - (x^* + \lambda B x^*)\|^2 + \Big[2\lambda L_1 + 3\lambda L_2\beta(1-\beta)\Big]\|x_n - x_{n-1}\|^2$$
$$-\frac{2\lambda}{3L_2}\|Cz_n - Cx^*\|^2. \tag{27}$$

Noting that

$$\epsilon := \frac{2}{3} - 3\lambda L_1 - 3\lambda L_2(1 + 2\beta - \beta^2) > 0$$

and

$$\mu := 2\lambda L_1 + 3\lambda L_2\beta(1-\beta) > 0,$$

we obtain from (27) that

$$\begin{aligned}&\|(x_{n+1} + \lambda B x_n) - (x^* + \lambda B x^*)\|^2 + (\mu + \epsilon)\|x_{n+1} - x_n\|^2\\ &\leq \|(x_n + \lambda B x_{n-1}) - (x^* + \lambda B x^*)\|^2 + \mu\|x_n - x_{n-1}\|^2\\ &\quad -\frac{2\lambda}{3L_2}\|Cz_n - Cx^*\|^2.\end{aligned} \tag{28}$$

Let $u_n := x_n + \lambda B x_{n-1}$ and $u^* := x^* + \lambda B x^*$. Then we have

$$\begin{aligned}\|u_{n+1} - u^*\|^2 + (\mu + \epsilon)\|x_{n+1} - x_n\|^2 \;\leq\; &\|u_n - u^*\|^2 + \mu\|x_n - x_{n-1}\|^2\\ &-\frac{2\lambda}{3L_2}\|Cz_n - Cx^*\|^2.\end{aligned} \tag{29}$$

Telescoping, we have

$$\sum_{n=1}^{\infty}\|Cz_n - Cx^*\|^2 < \infty$$

and this completes the proof. □

We now show that the sequence of iterates in Algorithm 3.3 converges weakly to a zero of inclusion problem (1) below.

Theorem 3.6. *Assume that Assumption 3.1 and Assumption 3.2 are satisfied and $\{x_n\}$ is generated by Algorithm 3.3. Then*

(i) $\{x_n\}$ converges weakly to $x^* \in (A + B + C)^{-1}(0)$;

(ii) $\{Bx_n\}$ converges weakly to Bx^*.

Proof. (i) We obtain from (29) that

$$\begin{aligned}&\|u_{n+1} - u^*\|^2 + \mu\|x_{n+1} - x_n\|^2 + \epsilon\sum_{j=0}^{n}\|x_{j+1} - x_j\|^2\\ &\leq \|u_0 - u^*\|^2 + \mu\|x_0 - x_{-1}\|^2.\end{aligned} \tag{30}$$

Thus, we have from (30) that $\{u_n\}$ is bounded and

$$\lim_{n\to\infty}\|x_n - x_{n-1}\| = 0.$$

Using the Lipschitz continuity of B, we obtain

$$\lim_{n\to\infty}\|Bx_n - Bx_{n-1}\| = 0,$$

and thus

$$\lim_{n\to\infty} \|u_n - u_{n-1}\| = 0.$$

Furthermore, we also obtain

$$\lim_{n\to\infty} \|Cx_n - Cx_{n-1}\| = 0$$

since C is Lipschitz continuous. Noting that

$$\begin{aligned} u_n &= x_n + \lambda B x_{n-1} + \lambda B x_n - \lambda B x_n \\ &= (I + \lambda B)x_n + \lambda B x_{n-1} - \lambda B x_n, \end{aligned}$$

we have $x_n = J_{\lambda B}(u_n - (\lambda B x_{n-1} - \lambda B x_n))$. Since $\{u_n\}$ is bounded, $\lim_{n\to\infty} \|Bx_n - Bx_{n-1}\| = 0$ and $J_{\lambda B}$ is nonexpansive, we have that $\{x_n\}$ is also bounded. From (28), we have

$$\lim_{n\to\infty} \left(\|u_n - u^*\|^2 + \mu\|x_n - x_{n-1}\|^2\right) = \lim_{n\to\infty} \|u_n - u^*\|^2. \tag{31}$$

Let (z^*, u^*) be a weak cluster point of the sequence $\{(x_n, u_n)\}$. Then there exists subsequence $\{x_{n_k}\}$ and $\{z_{n_k}\}$ such that $x_{n_k} \rightharpoonup z^*$ and $u_{n_k} \rightharpoonup u^*$. From (i), we have

$$\lim_{n\to\infty} \|Cz_n - Cx^*\| = 0.$$

Also,

$$\|z_n - x_n\| = \beta\|x_n - x_{n-1}\| \to 0, \quad n \to \infty.$$

Since $x_{n_k} \rightharpoonup z^*$, we then have that $z_{n_k} \rightharpoonup z^*$ also. By the fact that C is co-coercive and its graph is sequentially closed in $H^{\text{weak}} \times H^{\text{strong}}$ then we have $Cx^* = Cz^*$. Hence, $Cz_{n_k} \to Cz^*$, $k \to \infty$. We can re-write Algorithm 3.3 as

$$\begin{aligned} \begin{pmatrix} \lambda C z_n \\ 0 \end{pmatrix} - \begin{pmatrix} u_{n+1} - u_n \\ u_{n+1} - u_n \end{pmatrix} &\in \left(\begin{pmatrix} \lambda A & 0 \\ 0 & (\lambda B)^{-1} \end{pmatrix} + \begin{pmatrix} 0 & I \\ -I & 0 \end{pmatrix}\right) \\ &\times \begin{pmatrix} u_{n+1} - u_n + x_n \\ u_{n+1} - x_{n+1} \end{pmatrix} \end{aligned} \tag{32}$$

Since the operator

$$T := \begin{pmatrix} \lambda A & 0 \\ 0 & (\lambda B)^{-1} \end{pmatrix} + \begin{pmatrix} 0 & I \\ -I & 0 \end{pmatrix}$$

is maximally monotone and its graph is sequentially closed in $H^{\text{weak}} \times H^{\text{strong}}$ Passing to the limit of subsequence $\{n_k\}$ in (32), we have

$$\begin{cases} -\lambda C z^* \in \lambda A z^* + u^* - z^* \\ z^* \in (\lambda B)^{-1}(u^* - z^*) \end{cases}$$

and this implies that

$$\begin{cases} 0 \in A z^* + B z^* + C z^* \\ u^* = z^* + \lambda B z^*. \end{cases} \tag{33}$$

By Lemma 2.2, we have that $\{u_n\}$ converges weakly to $u^* = z^* + \lambda B z^*$, where z^* is the unique weak cluster point of $\{x_n\}$. By Lemma 2.1, we have the $\{x_n\}$ converges weakly to a point in $(A+B+C)^{-1}(0)$. This completes (i).

(ii) Observe that $u_n = x_n + \lambda B x_{n-1}$ implies that

$$\lambda B x_{n-1} = u_n - x_n \rightharpoonup u^* - z^* = \lambda B z^*, \quad n \to \infty.$$

Since

$$\lim_{n\to\infty} \|Bx_n - Bx_{n-1}\| = 0,$$

we have

$$Bx_n \rightharpoonup Bz^*, \quad n \to \infty.$$

□

We next give the following strong convergence result.

Theorem 3.7. *Suppose Assumption 3.1 and Assumption 3.2 are satisfied and A is η-strongly monotone on H. Let $\{x_n\}$ be generated by Algorithm 3.3. Then $\{x_n\}$ converges strongly to a point $x^* \in (A+B+C)^{-1}(0)$.*

Proof. Following the same line of arguments given in (8)–(29), we obtain

$$\begin{aligned} & 2\eta\|x^* - x_{n+1} - \lambda(Bx_n - Bx_{n-1})\|^2 + \|u_{n+1} - u^*\|^2 + (\mu+\epsilon)\|x_{n+1} - x_n\|^2 \\ \leq\ & \|u_n - u^*\|^2 + \mu\|x_n - x_{n-1}\|^2 - \frac{2}{3}\frac{\lambda}{L_2}\|Cz_n - Cx^*\|^2. \end{aligned}$$

Thus,

$$\begin{aligned} 2\eta\|x^* - x_{n+1} - \lambda(Bx_n - Bx_{n-1})\|^2 \ \leq\ & \Big(\|u_n - u^*\|^2 + \mu\|x_n - x_{n-1}\|^2\Big) \\ & -\Big(\|u_{n+1} - u^*\|^2 + \mu\|x_{n+1} - x_n\|^2\Big) - \epsilon\|x_{n+1} - x_n\|^2 \end{aligned}$$

which furthermore implies that

$$\begin{aligned} 2\eta\sum_{j=0}^{n}\|x^* - x_{j+1} - \lambda(Bx_j - Bx_{j-1})\|^2 \ \leq\ & \Big(\|u_0 - u^*\|^2 + \mu\|x_0 - x_{-1}\|^2\Big) \\ & -\Big(\|u_{n+1} - u^*\|^2 + \mu\|x_{n+1} - x_n\|^2\Big). \end{aligned}$$

Therefore,

$$\sum_{n=0}^{\infty}\|x^* - x_{n+1} - \lambda(Bx_n - Bx_{n-1})\|^2 < \infty$$

and

$$\lim_{n\to\infty}\|x^* - x_{n+1} - \lambda(Bx_n - Bx_{n-1})\| = 0.$$

Now,

$$\begin{aligned} \|x^* - x_{n+1}\| &= \|x^* - x_{n+1} - \lambda(Bx_n - Bx_{n-1}) + \lambda(Bx_n - Bx_{n-1})\| \\ &\leq \|x^* - x_{n+1} - \lambda(Bx_n - Bx_{n-1})\| + \lambda\|Bx_n - Bx_{n-1}\| \to 0, n \to \infty. \end{aligned}$$

This completes the proof. □

In our next theorem, we give a non-asymptotic $O(1/n)$ convergence rate of our proposed Algorithm 3.3.

Theorem 3.8. *If Assumption 3.1 and Assumption 3.2 are fulfilled and* $\{x_n\}$ *is generated by Algorithm 3.3, then for any* $x^* \in (A+B+C)^{-1}(0)$ *and* $n > 0$,

$$\min_{0\leq j\leq n} \|x_{j+1} - x_j\| \leq \frac{c_1}{\sqrt{n+1}}$$

where $c_1 := \sqrt{\frac{\|(x_0-x^*)+\lambda(Bx_{-1}-Bx^*)\|^2+\mu\|x_0-x_{-1}\|^2}{\epsilon}}$, $\epsilon = \frac{2}{3} - 3\lambda L_1 - 3\lambda L_2(1+2\beta-\beta^2)$ *and* $\mu = 2\lambda L_1 + 3\lambda L_2\beta(1-\beta)$.

Proof. We obtain from (30) that

$$\epsilon \sum_{j=0}^{n} \|x_{j+1} - x_j\|^2 \leq \|u_0 - u^*\|^2 + \mu\|x_0 - x_{-1}\|^2$$

and this implies that

$$\begin{aligned} \min_{0\leq j\leq n} \|x_{j+1} - x_j\|^2 &\leq \frac{\|u_0 - u^*\|^2 + \mu\|x_0 - x_{-1}\|^2}{(n+1)\epsilon} \\ &= \frac{\|(x_0 - x^*) + \lambda(Bx_{-1} - Bx^*)\|^2 + \mu\|x_0 - x_{-1}\|^2}{(n+1)\epsilon}. \end{aligned}$$

Therefore,

$$\min_{0\leq j\leq n} \|x_{j+1} - x_j\| \leq \frac{c_1}{\sqrt{n+1}},$$

where $c_1 := \sqrt{\frac{\|(x_0-x^*)+\lambda(Bx_{-1}-Bx^*)\|^2+\mu\|x_0-x_{-1}\|^2}{\epsilon}}$.

□

4 Numerical Simulations

We give numerical implementations of our proposed Algorithm 3.3 and compare the performance of Algorithm 3.3 with some other related schemes in [4, 18, 11, 31, 24, 46] using examples from compressed sensing and image processing. All codes were written in MATLAB R2020b and performed on a PC Desktop Intel(R) Core(TM) i7-6600U CPU @ 3.00GHz 3.00 GHz, RAM 32.00 GB.

4.1 Application to LASSO Problem in Compressed Sensing

Example 4.1. Given matrix $E \in \mathbb{R}^{M\times N}$, a vector $b \in \mathbb{R}^M$, λ a positive scalar, the l_1-norm regularized least squares model can be expressed as

$$\min_{x\in\mathbb{R}^N} \left\{ \frac{1}{2}||Ex - b||_2^2 + \lambda||x||_1 \right\}. \tag{34}$$

The l_1 regularization is a popular concept and has gained a lot of popularization in different areas. For example, when the least squares problem is posed with l_1 penalty, this is called Least Absolute Selection and Shrinkage Operator (LASSO)[43], and Basis Pursuit Denoising [13].

Compressed sensing is very important when it comes to the problem of efficiently acquiring and reconstructing a signal. This signal processing technique has to do with solving underdetermined linear systems $Ex = b$ (linear equations where $N \gg M$). In this case, where the number of unknowns is greater than the number of equations, the linear system generates many solutions or could result in no solution. The approach to solving such a system is known as the linear least squares method (finding the minimum l_2-norm solution). The above l_1-norm regularized least squares model (34) can be computed to recover x when x is sparse which is the case in most applications. The model given in (34) is most often referred to as LASSO. Standard general algorithms such as an Interior Point Method (IPM), [5], can be used to solve the LASSO problem by reformulating the problem as a second order cone programming. However, the computational complexity of such traditional methods is too high to handle large-scale data encountered in many real-life applications.

The LASSO problem is a special case of convex minimization problem where

$$f(x) = \frac{1}{2}||Ex - b||^2, \quad g(x) = \lambda||x||_1.$$

Its gradient $\nabla f = E^T Ex - E^T b$ is Lipschitz continuous with Lipschitz constant $L(f) = \|E^T E\|$. The proximal map with $g(x) = \lambda||x||_1$ is given as

$$\text{prox}_g(x) = arg\min_u \lambda||x||_1 + \frac{1}{2}||u - x||_2^2,$$

which is separable in indices. Thus, for $x \in \mathbb{R}^N$,

$$\begin{aligned} \text{prox}_g(x) &= \text{prox}_{\lambda||.||_1}(x) \\ &= \Big(\text{prox}_{\lambda|.|_1}(x_1), \cdots, \text{prox}_{\lambda|.|_1}(x_N)\Big) \\ &= (\alpha_1, \cdots, \alpha_N), \end{aligned}$$

where

$$\alpha_k = \text{sgn}(x_k)\max\{|x_k| - \lambda, 0\} \quad \text{for} \quad k = 1, 2, \ldots, N.$$

First, we implementing our algorithm to solve the minimization problem (34). The matrix $E \in \mathbb{R}^{M\times N}$, vector $b \in \mathbb{R}^M$ and starting points $x^0, x^{-1} \in \mathbb{R}^N$ are randomly generated such that:
Case I: $M = 20$ and $N = 100$,
Case II: $M = 50$ and $N = 200$,
Case III: $M = 100$ and $N = 500$
Case IV: $M = 200$ and $N = 1000$.
The initial parameter β is chosen such that $\beta \in [0, 1]$ and λ satisfies Assumption 3.2. From Figure 1, we notice that the performance of the algorithm worsen as the value of β increased. In particular, the algorithm performs worse when $\beta = 1$.

Table 1: Computation result comparing the performance of Algorithm 3.3 with different values of β.

		$\beta = 0.1$	$\beta = 0.3$	$\beta = 0.5$	$\beta = 0.7$	$\beta = 1$
Case I	No of Iter.	10	10	10	13	43
	CPU time (sec)	0.0036	0.0072	0.0091	0.0101	0.0349
Case II	No of Iter.	11	11	11	15	69
	CPU time (sec)	0.0130	0.0127	0.0120	0.0163	
Case III	No of Iter.	11	11	12	15	87
	CPU time (sec)	0.0793	0.0643	0.0614	0.0741	0.4052
Case IV	No of Iter.	11	11	12	15	72
	CPU time (sec)	0.5293	0.5326	0.5376	0.6740	4.5041

In this experiment, we begin by considering a typical compressing sensing problem with the ultimate goal of reconstructing a length-N sparse signal from M observation, with $M \ll N$. A matrix E (partial DWT) whose M rows are randomly selected from the $N \times N$ DWT matrix. This type of matrix E requires no storage and helps in speeding up the matrix-vector multiplications involving E and E^T. Our aim is to recover x from the nosiy experiment b. We choose $N = 2^{12}$ and $M = 2^{10}$ and the original signal contains 160 randomly nonzero elements for the experiment. The vector b is randomly generated by the normal distribution with $SNR = 40$, E is generated via the normal distribution with mean zero and variance one, and $x \in \mathbb{R}^N$ is generated by a uniform distribution in [-2,2]. The quality of recovery is assessed by the mean squared error to the original signal x where

$$MSE = \frac{1}{N}\|x_n - x\| < 10^{-3},$$

where x_n is an estimated signal of x. E is the Gaussian matrix generated by the command `rand(m,n)` in MATLAB. Typically speaking, the lesser the MSE value, the better the quality of the signal recovered. In our numerical experiments, we set $L = \|E^T E\|$, and use the following parameters for the algorithms: we chose $\beta = 0.5$ for our Algorithm 3.3 (namely, ORFBSIM), we take $\lambda = 1/\mathrm{L}$ for FISTA [4, 11]; SDRA [18], iSDRA [24] and IF-BA [31] and $\lambda = 1/L, \alpha_k = \beta_k = \frac{k-1}{k+\alpha-1}$ with $\alpha > 3$ for vGIGPM [46]. The numerical result is presented in Fig. 2. We note that while the new Algorithm 3.3 (ORFBSIM) and IF-BA introduced in [31] converged at 50 and 52 iterations respectively, other methods, FISTA [4, 11], SDRA [18], iSDRA [24] and vGIGPM [46], used in the numerical comparison do not converge even at the maximum iteration set at 1000. Furthermore, Algorithm 3.3 (ORFBSIM) uses the least time of computation compared to any other method. However, Algorithm 3.3 (ORFBSIM) ranked third in term of the MSE valued. The numerical results showing the graphs of objective value function against number of iteration and CPU time are shown in Fig. 3.

4.2 Application to Image Processing

Example 4.2. The image restoration problem is formulated by the following model

$$z = Ex + b, \tag{35}$$

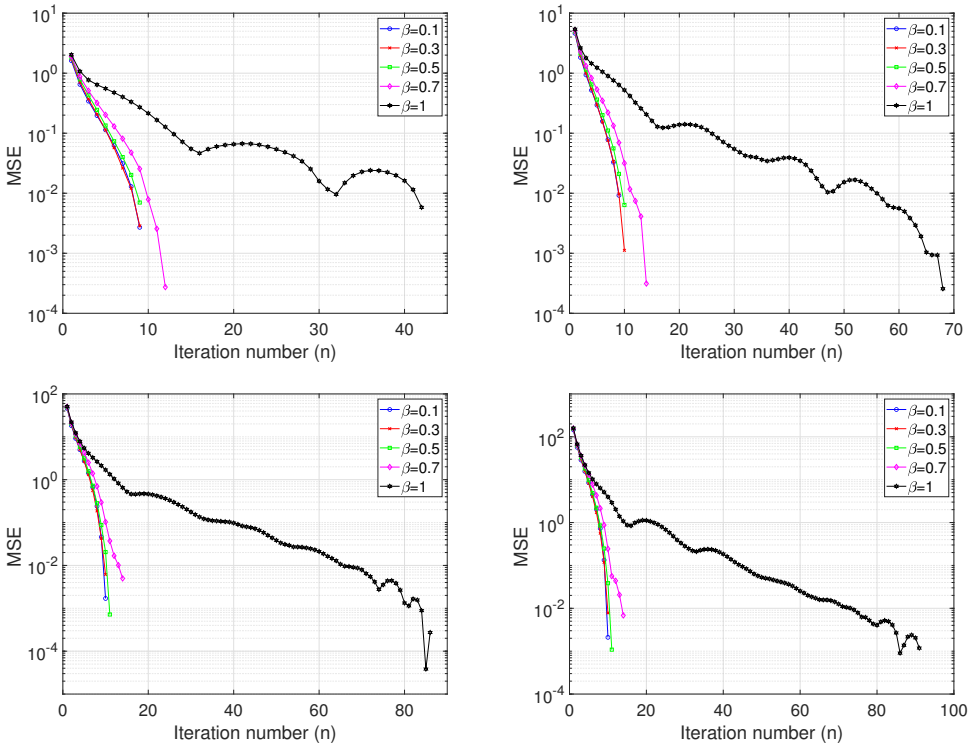

Figure 1: Comparison of the performance of Algorithm 3.3 with different β. Top Left: Case I; Top Right: Case II; Bottom Left: Case III; Bottom Right: Case IV.

where x is the original image, z is the degraded image, E is a blurring matrix and b is the noise. One of the efficient methods for recovering the original image is the l_1-norm regularized least square model (34). In this case, E represents the blurring operator, x is the original image and b is the observed image. Our aim here is to recover the original image x given the data of the blurred image z. We consider the grey scale image of M pixels wide and N pixel height, each value is known to be in the range $[0, 255]$. Let $D = M \times N$. The quality of the restored image is measured by the signal-to-noise ratio defined as

$$SNR = 20 \times \log_{10}\left(\frac{\|x\|_2}{\|x - x^*\|_2}\right),$$

where x is the original image and x^* is the restored image. Typically, the larger the SNR, the better the quality of the restored image. In this experiment, we aim to restore a two-dimensional image from its limited measurement. We used the Cameraman, Pout and MRI images which are degraded by Gaussian 7×7 blur kernel with standard deviation 4. We also used similar parameters as in Example 4.1 for the test algorithms with the initial values taken as $x_0 = \mathbf{0} \in \mathbb{R}^D$ and $x_1 = \mathbf{1} \in \mathbb{R}^D$. Classical test image were obtained from MATLAB toolbox. The performance of the algorithms are shown in Figure 4 – Figure 6. In Figure 7, we show the numerical results using the graph of SNR value against number of iteration for each test image. More so, in Table 1, we recorded the SNR value and the CPU time used by the algorithms for the numerical experiment. In all our simulations, we observed that

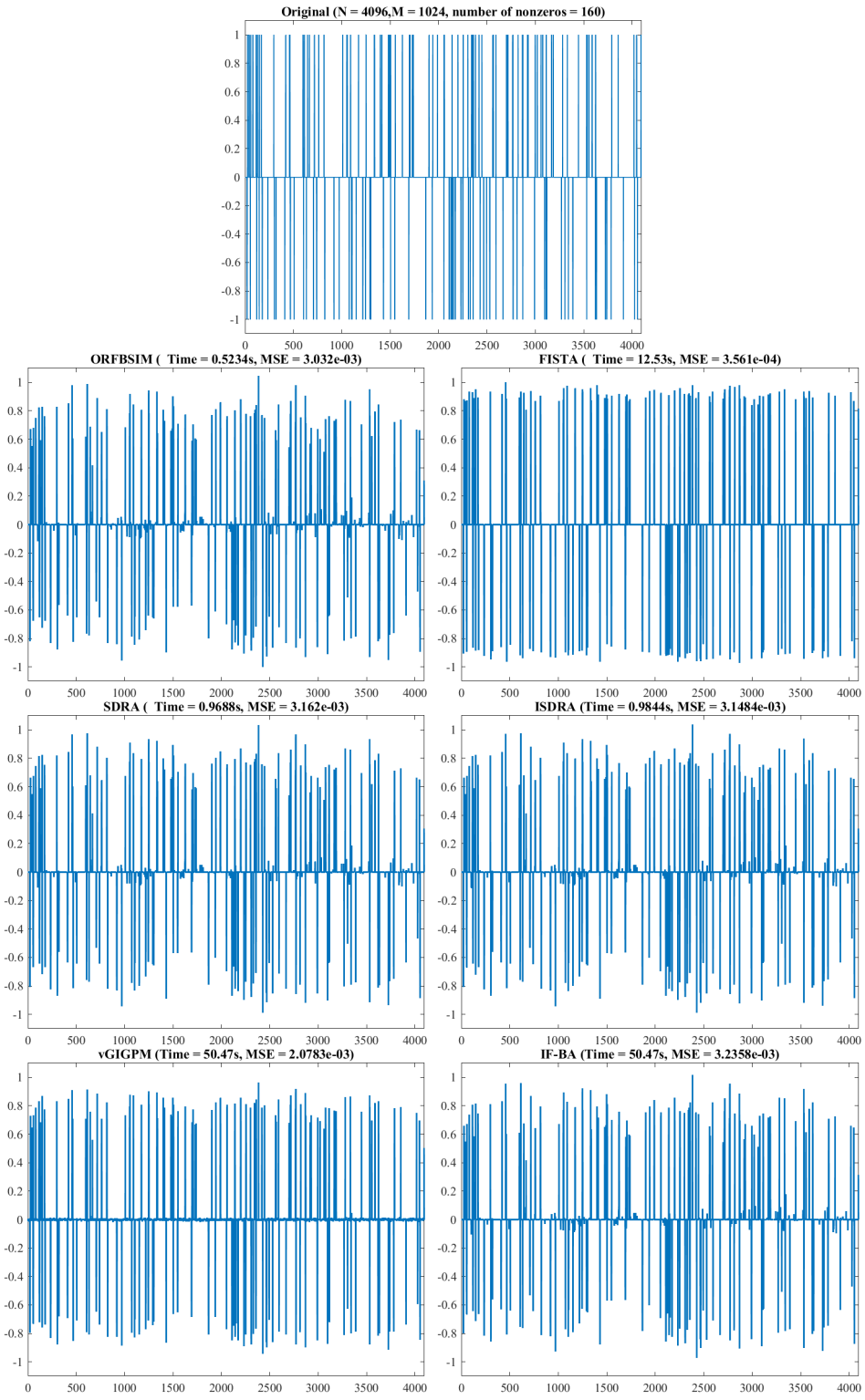

Figure 2: Algorithms performance for recovery the sparse signal.

our Algorithm 3.3 (ORFBSIM) performs better than FISTA [4, 11], SDRA [18], iSDRA [24], IF-BA [31] and vGIGPM [46] in terms of SNR value and CPU time.

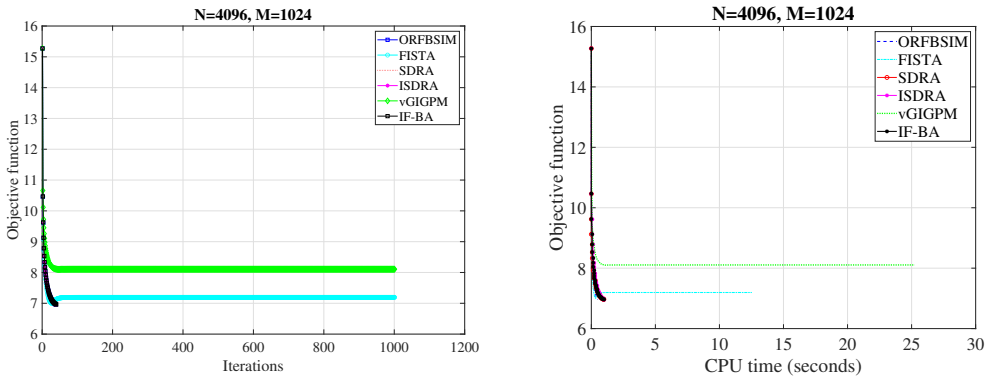

Figure 3: Numerical results showing the objective value against number of iteration (right) and CPU time (right).

Table 2: Numerical results showing the SNR value and CPU time for each algorithm

Methods	Cameraman		Pout		MRI	
	SNR	Time (sec)	SNR	Time (sec)	SNR	Time (sec)
ORFBSIM	35.4464	13.9949	40.3108	13.9465	27.1674	3.4257
FISTA	34.4353	14.3480	36.3893	14.3618	26.4704	3.5447
SDRA	34.4954	18.1721	40.0287	17.6751	26.5755	5.2949
ISDRA	34.5813	20.9432	38.3175	21.0578	26.9213	5.3720
vGIGPM	34.7088	17.6924	39.6908	18.3660	26.7975	3.5548
IF-BA	34.8683	18.1798	40.2591	18.4484	26.7757	3.6252

Declarations

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 upon request to the authors.

Availability of supporting data

Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

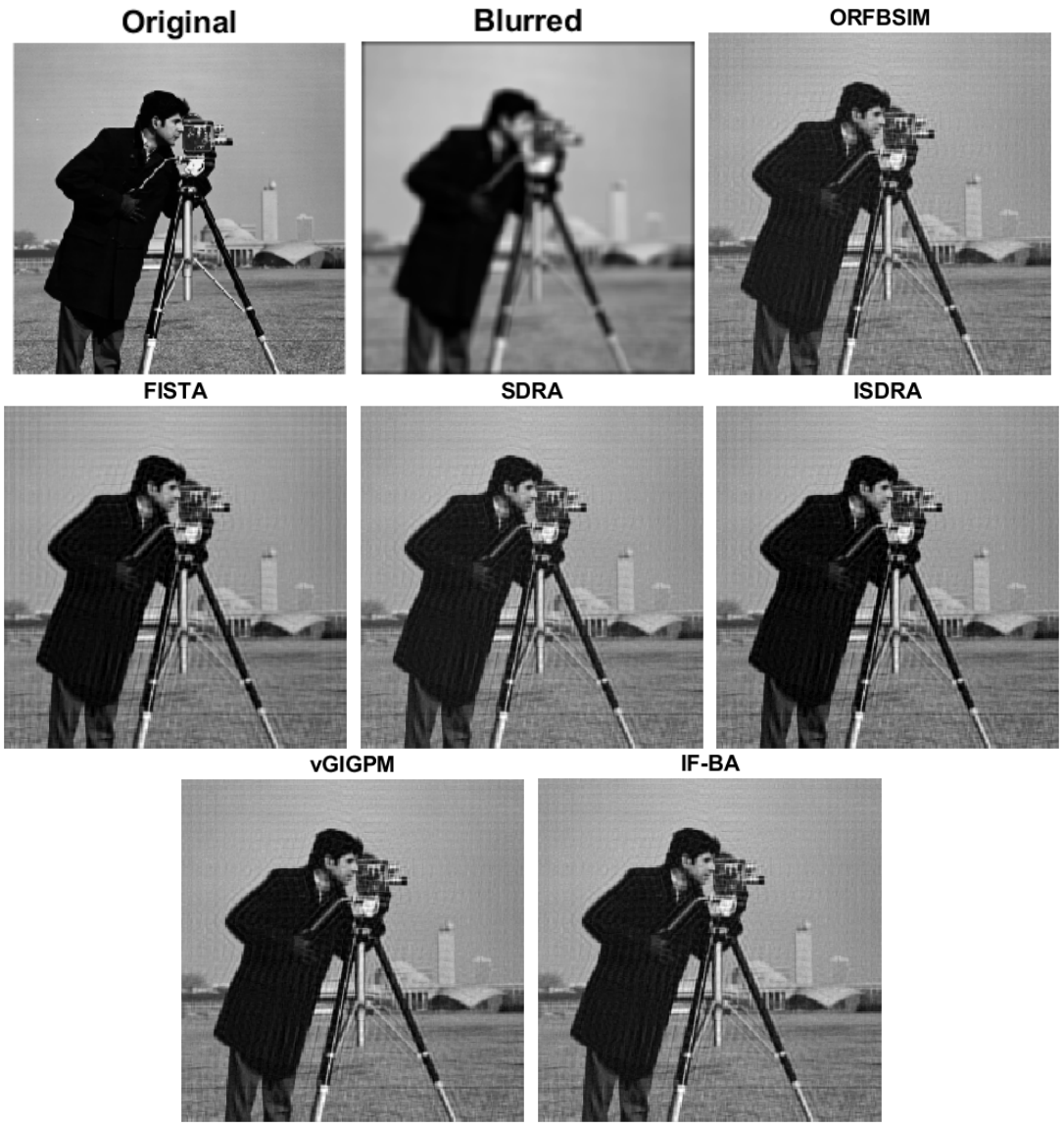

Figure 4: Performance of the algorithms using `Cameraman` (256×256) test image.

Competing interests

The authors declare no competing interests.

Funding

Not Applicable.

References

[1] H. Attouch, J. Peypouquet, P. Redont; Backward-forward algorithms for structured monotone inclusions in Hilbert spaces, *J. Math. Anal. Appl.* **457** (2018), 1095-1117.

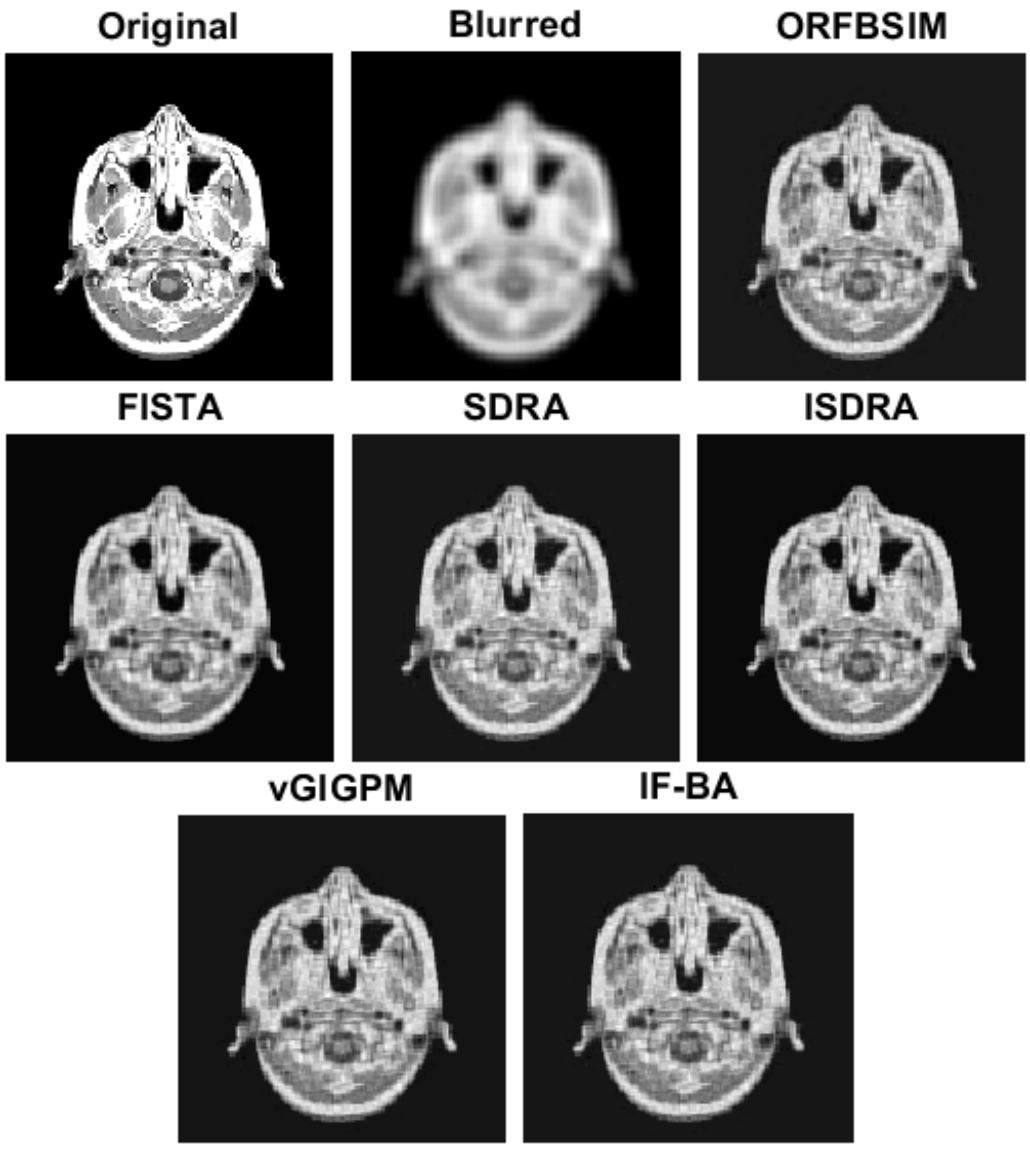

Figure 5: Performance of the algorithm using `MRI` (128×128) test image.

[2] F. J. Aragón-Artacho, D. Torregrosa-Belén; A Direct Proof of Convergence of Davis–Yin Splitting Algorithm Allowing Larger Stepsizes, *Set-Valued Var. Anal.* (2022), https://doi.org/10.1007/s11228-022-00631-6.

[3] H. H. Bauschke and P. L. Combettes; Convex Analysis and Monotone Operator Theory in Hilbert Spaces, *CMS Books in Mathematics, Springer, New York* (2011).

[4] A. Beck, M. Teboulle; A fast iterative shrinkage-thresholding algorithm for linear inverse problems, *SIAM J. Imaging Sci.* **2** (2009), 183–202.

[5] A. Ben-Tal, A. Nemirovski; Lectures on modern convex optimization: Analysis, Algorithms and Engineering Applications, *MPS-SIAM series on optimization, SIAM,Philadelphia* (2001).

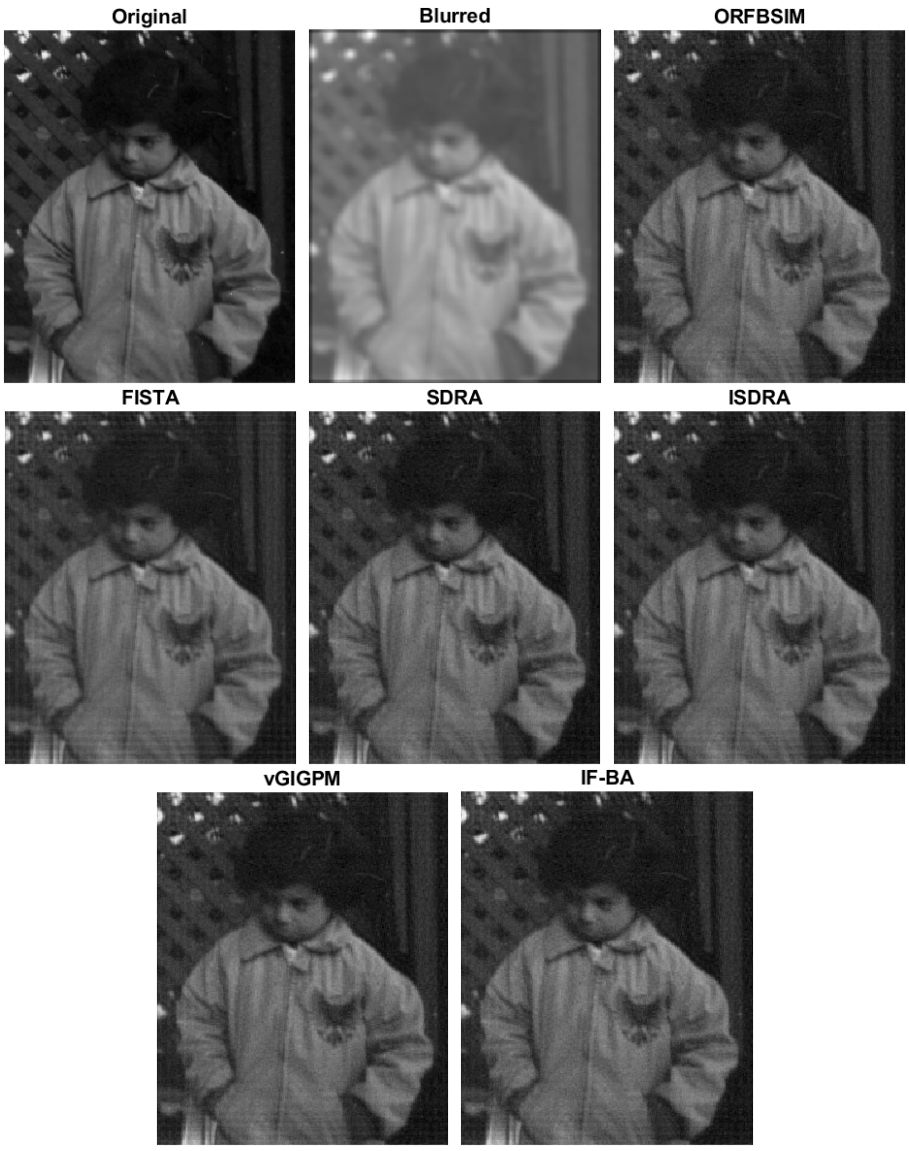

Figure 6: Performance of the algorithm using `Pout` (291×240) test image.

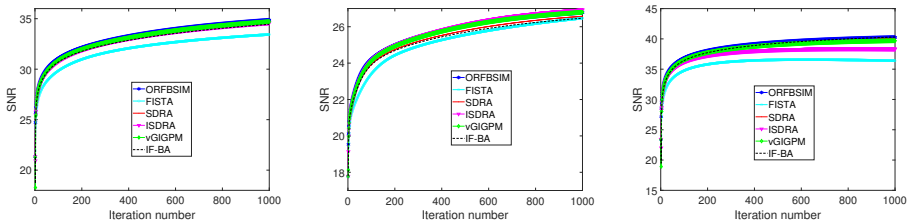

Figure 7: Comparison of the performance of the Algorithms: Cameraman (left), MRI (middle), Pout (right)

[6] R. I. Bot, E. R. Csetnek; An inertial forward-backward-forward primal-dual splitting algorithm for solving monotone inclusion problems, *Numer. Algorithms* **71** (2016), 519-540.

[7] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein; Distributed optimization and statistical learning via the alternating direction method of multipliers, *Foundations and Trends in Machine Learning* **3**(1) (2011), 1-122.

[8] L. M. Briceño-Arias; Forward-Douglas-Rachford splitting and forward-partial inverse method for solving monotone inclusions, *Optimization* **64**(5) (2015), 1239-1261.

[9] L. M. Briceño-Arias, D. Davis; Forward-backward-half forward algorithm for solving monotone inclusions, *SIAM J. Optim.* **28** (2018), 2839-2871.

[10] V. Cevher, B.C. Vũ; A reflected Forward-Backward splitting method for monotone inclusions involving Lipschitzian operators, *Set-Valued Var. Anal.* **29** (2021), 163-174.

[11] A. Chambolle, C. Dossal; On the convergence of the iterates of the "fast iterative shrinkage/ thresholding algorithm", *J. Optim. Theory Appl.* **66**(3) (2015), 968–982.

[12] A. Chambolle, T. Pock; A first-order primal-dual algorithm for convex problems with applications to imaging, *J. Math. Imaging Vision* **40**(1) (2011), 120-145.

[13] S. Chen, D. L. Donoho, M. Saunders; Atomic decomposition by basis pursuit, *SIAM J. Sci. Comput.* **20** (1998), 33-61.

[14] P. Cholamjiak, D. Van Hieu, Y.J. Cho; Relaxed Forward–Backward Splitting Methods for Solving Variational Inclusions and Applications, *J. Sci. Comput.* **88** (2021), 85.

[15] P. L. Combettes, V. Wajs; Signal recovery by proximal forward-backward splitting, *SIAM Multiscale Model. Simul.* **4** (2005), 1168-1200.

[16] P. L. Combettes, J. C. Pesquet; Proximal splitting methods in signal processing. In: Fixed point algorithms for inverse problems in science and engineering, *Springer, New York* (2011) 185-212.

[17] L. Condat; A primal-dual splitting method for convex optimization involving Lipschitzian, proximable and linear composite terms, *J. Optim. Theory Appl.* **158**(2) (2013), 460–479.

[18] E. R. Csetnek, Y. Malitsky, M. K. Tam; Shadow Douglas-Rachford splitting for monotone inclusions, *Appl. Math. Optim.* **80** (2019), 665–678.

[19] I. Daubechies, M. Defrise, C. De Mol; An iterative thresholding algorithm for linear inverse problems with a sparsity constraint, *Commun. Pure Appl. Math.* **57** (2004), 1413-1457.

[20] D. Davis, W. T. Yin; A three-operator splitting scheme and its optimization applications, Set Valued Var. Anal. **25** (2017), 829-858.

[21] Y. D. Dong, A. Fischer; A family of operator splitting methods revisited, *Nonlinear Anal.* **72** (2010), 4307-4315.

[22] J. Douglas, H. H. Rachford; On the numerical solution of heat conduction problems in two or three space variables. *Trans. Amer. Math. Soc.* **82** (1956), 421-439.

[23] J. Duchi, Y. Singer; Efficient online and batch learning using forward–backward splitting, *J. Mach. Learn. Res.* **10** (2009), 2899-2934.

[24] J. Fan, X. Qin, B. Tan; Convergence of an inertial shadow Douglas-Rashford splitting algorithm for monotone inclusion, *Numer. Funct. Anals. Optim.* **42** (2021), 1627-1644.

[25] A. Gibali, D. V. Hieu; A new inertial double-projection method for solving variational inequalities, *J. Fixed Point Theory Appl.* **21** 97 (2019)

[26] D. V. Hieu, D. V. Thong; A new projection method for a class of variational inequalities, *Appl. Anal.* **98** (13) (2019), 2423-2439.

[27] D. V. Hieu; Convergence analysis of a new algorithm for strongly pseudomontone equilibrium problems, *Numer. Algorithms* **77** (2018), 983-1001.

[28] D. V. Hieu; New extragradient method for a class of equilibrium problems in Hilbert spaces, *Appl. Anal.* **97** (2018), 811-824.

[29] P. Latafat, P. Patrinos; Asymmetric forward-backward-adjoint splitting for solving monotone inclusions involving three operators, *Comput. Optim. Appl.* **68** (2017), 57-93.

[30] P. L. Lions, B. Mercier; Splitting algorithms for the sum of two nonlinear operators, *SIAM J. Numer. Anal.* **16** (1979), 964-979.

[31] D. A. Lorenz, T. Pock; An inertial Forward-Backward algorithm for monotone inclusions, *J. Math. Imaging Vision* **51** (2015), 311-325.

[32] Y. Malitsky, M. K. Tam; A forward–backward splitting method for monotone inclusions without cocoercivity, *SIAM J. Optim.* **30** (2) (2020), 1451-1472.

[33] A. Moudafi, M. Oliny; Convergence of a splitting inertial proximal method for monotone operators, *J. Comput. Appl. Math.* **155** (2003), 447-454.

[34] Z. Opial; Weak convergence of the sequence of successive approximations for nonexpansive mappings, *Bull. Am. Math. Soc.* **73** (1967), 591–597.

[35] A. Padcharoen, D. Kitkuan, W. Kumam; Tseng methods with inertial for solving inclusion problems and application to image deblurring and image recovery problems, *Comput. Math. Methods.* **3**(3), Paper No. e1088, 14pp. (2020).

[36] G. B. Passty; Ergodic convergence to a zero of the sum of monotone operators in Hilbert spaces, *J. Math. Anal. Apppl.* **72**(2) (1979), 383-390.

[37] B. T. Polyak; Some methods of speeding up the convergence of iteration methods, *U.S.S.R. Comput. Math. Math. Physc.* **4**(5) (1964), 1-17.

[38] H. Raguet, J. Fadili, G. Peyre; Generalized forward-backward splitting, *SIAM J. Imaging Sci.* **6** (2013), 1199-1226.

[39] J. Rieger, M. K. Tam; Backward-forward-reflected-backward splitting for three operator monotone inclusions, *Appl. Math. Comput.* **381** (2020), 125248.

[40] E. K. Ryu, S. Boyd, A primer on monotone operator methods, *Appl. Comput. Math.* **15** (2016), 3-43.

[41] E. K. Ryu; Uniqueness of DRS as the 2 operator resolvent-splitting and impossibility of 3 operator resolvent-splitting, *Math. Program.* **182** (2020), 233-273.

[42] E. K. Ryu, B. C. Vũ; Finding the forward-Douglas–Rachford-forward method, *J. Optim. Theory Appl.* **184** (2020), 858-876.

[43] R. Tibshirami; Regression shrinkage and selection via lasso, *J. Roy. Statist. Soc. Ser. B* **58** (1996), 267-288.

[44] P. Tseng; A modified forward-backward splitting method for maximal monotone mappings, *SIAM J. Control Optim.* **38**(2) (2000), 431-446.

[45] B. C. Vũ; A splitting algorithm for dual monotone inclusions involving cocoercive operators. *Adv. Comput. Math.* **38**(3) (2013), 667-681.

[46] Z. Wu, M. Li; General inertial proximal gradient method for a class of nonconvex nonsmooth optimization problems, *Comp. Optim. Appl.* **73** (2019), 129-158.

[47] H. Yu, C. Zong, Y. Tang; An outer reflected forward-backward splitting algorithm for solving monotone inclusions, *arXiv:2009.12493v1* (2020).

[48] C. Zong, Y. Tang, Y. J. Cho; Convergence analysis of an inexact three-operator splitting algorithm, *Symmetry* 2018, 10, 563.