Strongly Convergent Inertial Proximal Point Algorithm Without On-line Rule

Lateef O. Jolaoso[*], Yekini Shehu[†], Jen-Chih Yao[‡]

May 12, 2026

Abstract

We present a strongly convergent Halpern-type proximal point algorithm with double inertial effects to find a zero of a maximal monotone operator in Hilbert spaces. The strong convergence results are obtained without on-line rule of the inertial parameters and the iterates. This makes our proof arguments different from what is obtainable in the literature where on-line rule is imposed on a strongly convergent proximal point algorithm with inertial extrapolation. Numerical examples with applications to image restoration and compressed sensing show that our proposed algorithm is useful and has practical advantages over existing ones.
Keywords: Proximal point algorithm; Two-point inertia; Maximal monotone operators; Strong convergence; Hilbert spaces.

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

Communicated by Shoham Sabach.

1 Introduction

Let H be a real Hilbert space with inner product $\langle .,. \rangle$ and induced norm $\| \cdot \|$. Suppose a maximal monotone set-valued operator $A : H \to 2^H$ is given, let us study the inclusion problem:

$$\text{find } x \in H \ \text{ such that } \ \mathbf{0} \in A(x). \tag{1}$$

[*]Department of Mathematics and Applied Mathematics, Sefako Makgatho Health Sciences University, P.O. Box 94 Medunsa 0204, Pretoria, South Africa; School of Mathematical Sciences, University of Southampton, S017 1BJ, United Kingdom; e-mail: l.o.jolaoso@soton.ac.uk

[†](Corresponding Author) School of Mathematical Sciences, Zhejiang Normal University, Jinhua 321004, People's Republic of China; e-mail: yekini.shehu@zjnu.edu.cn

[‡]Research Center for Interneural Computing, China Medical University Hospital, China Medical University, Taichung, Taiwan; Academy of Romanian Scientists, 50044 Bucharest, Romania; and Department of Applied Mathematics, National Sun Yat-sen University, Kaohsiung, Taiwan; e-mail: yaojc@mail.cmu.edu.tw

We shall denote the set of zeros in (1) by $A^{-1}(\mathbf{0})$. It is known that fixed point problem, variational inequality problem, minimization of closed proper convex functions, and their variants are special cases of inclusion problem (1).

Previous Results. The proximal point algorithm (PPA) introduced by Martinet and developed further by Rockafellar and others (see, for example, [18,39,41,42,49]) has been a popular algorithm for solving inclusion problem (1): $x_0 \in H$,

$$x_{n+1} = J_\lambda^A(x_n), \tag{2}$$

(where the operator $J_\lambda^A := (I + \lambda A)^{-1}$ is the so-called resolvent operator, introduced by Moreau [41]) which is equivalent to

$$\mathbf{0} \in \lambda A(x_{n+1}) + x_{n+1} - x_n \tag{3}$$

where $\lambda > 0$ is called proximal parameter. The PPA (2) is a very powerful algorithmic tool and contains many well-known algorithms as special cases, including the classical augmented Lagrangian method [27,48], the Douglas-Rachford splitting method [21,34] and the alternating direction method of multipliers [23,24]. Interesting results on weak convergence and rate of convergence of PPA have been obtained in [22, 25, 40]. The equivalent representation of the PPA (3) can be written as

$$\mathbf{0} \in \frac{x_{n+1} - x_n}{\lambda} + A(x_{n+1}). \tag{4}$$

This can be viewed as an implicit discretization of the evolution differential inclusion problem

$$\mathbf{0} \in \frac{dx}{dt} + A(x(t)) \tag{5}$$

It has been shown that the solution trajectory of (5) converges to a solution of (1) provided that A satisfies certain conditions (see, for example, [11]).

One of the open problems proposed by Rockefellar [50]: Is the sequence generated by PPA (2) strongly convergent?

Güler provided a counter example in [26] to show that PPA (2) does not necessarily converge strongly even if A in PPA (2) is the subdifferential of a convex proper lower semicontinuous function. Since then several modifications of PPA (2) have been proposed and studied in the literature. Xu [56] and Kamimura and Takahashi [29] introduced a Halpern-type proximal point algorithm which guarantees the strong convergence. Their results were extended by He et al. [28], Xu [57], Marino and Xu [38], Yao and Noor [58], Boikanyo and Morosanu [7–10], Wang and Cui [55], Khatibzadeh and Ranjbar [30] and host of other authors.

Xu [56] and Kamimura and Takahashi [29] introduced the following regularization of PPA (2) (which is a Halpern-type PPA):

$$x_{n+1} = \alpha_n x_0 + (1 - \alpha_n) J_\lambda^A(x_n), \tag{6}$$

where $\alpha_n \in (0,1)$ satisfying some conditions. Boikanyo and Morosanu [7] showed that Halpern-type PPA (6) is equivalent to the following iterative method suggested by Xu [56]:

$$x_{n+1} = J_\lambda^A(\alpha_n x_0 + (1-\alpha_n)x_n). \tag{7}$$

Recent strong convergence results of Halpern-type PPA can be found in [32, 33, 44, 51].

To speed up convergence of PPA (2), the following second order evolution differential inclusion problem was introduced in the literature (see, e.g., [1, 5]):

$$\mathbf{0} \in \frac{d^2x}{dt^2} + \beta\frac{dx}{dt} + A(x(t)), \tag{8}$$

where $\beta > 0$ is a friction parameter. An implicit discretization of (8) was studied in [2, 3] as

$$\mathbf{0} \in \frac{x_{n+1} - 2x_n + x_{n-1}}{h^2} + \beta\frac{x_{n+1} - x_n}{h} + A(x_{n+1}), \tag{9}$$

which is consequently proximal point algorithm with one-step inertial extrapolation:

$$x_{n+1} = J_\lambda^A(x_n + \theta(x_n - x_{n-1})), \tag{10}$$

with $\lambda = \frac{h^2}{1+\beta h}$ and $\theta = \frac{1}{1+\beta h}$. Weak convergence properties of (10) have been studied in [2–4, 35, 36, 42] under some assumptions on the parameters θ and λ.

The strongly convergent proximal point algorithm with one-step inertial extrapolation proposed in the literature is of the form:

$$\begin{cases} y_n = x_n + \theta_n(x_n - x_{n-1}), \\ x_{n+1} = J_\lambda^A(\alpha_n x_0 + (1-\alpha_n)y_n), \end{cases} \tag{11}$$

where the inertial extrapolation step $y_n = x_n + \theta_n(x_n - x_{n-1}), \theta_n \geq 0$ (which we call hereafter, one-step inertial extrapolation) is such that $0 \leq \theta_n \leq \bar{\theta}_n$ and

$$\bar{\theta}_n = \begin{cases} \min\left\{\theta, \frac{\epsilon_n}{\|x_n - x_{n-1}\|}\right\}, & x_n \neq x_{n-1} \\ \theta, & \text{otherwise}, \end{cases} \tag{12}$$

where $\lim_{n\to\infty} \frac{\epsilon_n}{\alpha_n} = 0$ and $\theta > 0$. The condition (12) is called the on-line rule and (12) has been used extensively in the literature by several authors whenever strongly convergent inertial PPA (11) is considered (see, for example, [15, 16, 19, 53, 54, 59]).

Limitations of proximal point algorithm with one-step inertial extrapolation have been discussed in [46, 47], using example from feasibility problem with Douglas-Rachford splitting algorithm and ADMM both with one-step inertial extrapolation. Consequently, replacing one-step inertial extrapolation with the following two-step inertial extrapolation

$$y_n = x_n + \theta(x_n - x_{n-1}) + \delta(x_{n-1} - x_{n-2}), \tag{13}$$

where $\theta \geq 0$ and $\delta \leq 0$ could be more beneficial numerically and provide acceleration over the one-step inertial extrapolation. This alluded to the observation made by Polyak in [45] that multi-step inertial extrapolation can boost the speed of optimization algorithms even though neither the convergence results nor the rate of convergence of such is established by Polyak [45]. Recent results on algorithms with multi-step inertial extrapolation can be found in [17, 20]. Weak convergence results of PPA with two-step inertial extrapolation was recently obtained in [43].

Based on trend of results obtained so far in the literature for PPA with inertial extrapolation step, the following question arises:

Question: Can we obtain strongly convergent proximal point algorithm with two-step inertial extrapolation without assuming on-line rule (12)?

Our contribution. Our contribution in this paper is to answer the above question in the affirmative.

- We propose a Halpern-type proximal point algorithm with two-step inertial extrapolation and obtain strong convergence results without assuming on-line rule (12). Our method of proof is different from the approach used in the literature while employing the on-line rule (12), which is of independent interest.
- Numerical tests including applications to image restoration and compressed sensing are given to emphasize the usefulness and practical advantages of our algorithm over existing related ones in the literature.

Outline. In Section 2, we give some basic definitions and lemmas which will be needed in our convergence analysis. In Section 3, we introduce a Halpern-type proximal point algorithm with two-step inertial extrapolation and give strong convergence results without on-line rule assumption. Numerical tests including applications to image restoration and compressed sensing are given in Section 4 while final remarks are given in Section 5.

2 Preliminaries

In this section, we give some definitions and basic results that will be used in our subsequent analysis. The weak and the strong convergence of $\{x_n\} \subset H$ to $x \in H$ is denoted by $x_n \rightharpoonup x$ and $x_n \to x$ as $n \to \infty$ respectively.

Definition 2.1. *A mapping $T : H \to H$ is called*

(i) nonexpansive if $\|Tx - Ty\| \leq \|x - y\|$, for all $x, y \in H$;

(ii) firmly nonexpansive if $\|Tx - Ty\|^2 \leq \|x - y\|^2 - \|(I - T)x - (I - T)y\|^2$ for all $x, y \in H$. Equivalently, T is firmly nonexpansive if $\|Tx - Ty\|^2 \leq \langle x - y, Tx - Ty\rangle$ for all $x, y \in H$;

(iii) *averaged if* T *can be expressed as the averaged of the identity mapping* I *and a nonexpansive mapping* S, *i.e.*, $T = (1-\alpha)I + \alpha S$ *with* $\alpha \in (0,1)$. *Alternatively,* T *is* α*-averaged if*

$$\|Tx - Ty\|^2 \le \|x-y\|^2 - \frac{1-\alpha}{\alpha}\|(I-T)x - (I-T)y\|^2, \forall x, y \in H.$$

Definition 2.2. *A multivalued mapping* $A : H \to 2^H$ *is said to be monotone if for any* $x, y \in H$,

$$\langle x - y, f - g\rangle \ge 0,$$

where $f \in Ax$ *and* $g \in Ay$. *The Graph of* A *is defined by*

$$Gr(A) := \{(x, f) \in H \times H : f \in Ax\}.$$

If $Gr(A)$ *is not properly contained in the graph of any other monotone mapping, then we say that* A *is maximal. It is well-known that for each* $x \in H$, *and* $\lambda > 0$, *there is a unique* $z \in H$ *such that* $x \in (I + \lambda A)z$. *The single-valued operator* $J_\lambda^A(x)$ *is called the resolvent of* A *(see [22]).*

Lemma 2.3. *The following identities hold for all* $u, v \in H$*:*

$$2\langle u, v\rangle = \|u\|^2 + \|v\|^2 - \|u - v\|^2 = \|u + v\|^2 - \|u\|^2 - \|v\|^2.$$

Lemma 2.4. *Let* $x, y, z \in H$ *and* $a, b \in \mathbb{R}$. *Then*

$$\begin{aligned}\|(1+a)x - (a-b)y - bz\|^2 &= (1+a)\|x\|^2 - (a-b)\|y\|^2 - b\|z\|^2 \\ &\quad +(1+a)(a-b)\|x-y\|^2 + b(1+a)\|x-z\|^2 \\ &\quad -b(a-b)\|y-z\|^2.\end{aligned}$$

Lemma 2.5. *[52] Suppose that* $\{t_n\}$ *is a sequence of nonnegative real numbers,* $\{\sigma_n\}$ *is a sequence of real numbers in* $(0,1)$ *satisfying* $\sum_{n=1}^{\infty}\sigma_n = \infty$, *and* $\{h_n\}$ *is a sequence of real numbers such that*

$$t_{n+1} \le (1-\sigma_n)t_n + \sigma_n h_n, \; n \ge 0.$$

If $\limsup_{i\to\infty} h_{n_i} \le 0$ *for each subsequence* $\{t_{n_i}\}$ *of* $\{t_n\}$ *satisfying* $\liminf_{i\to\infty}(t_{n_i+1} - t_{n_i}) \ge 0$, *then* $\lim_{n\to\infty} t_n = 0$.

Lemma 2.6. *[37, Lem. 3.1] Suppose that* $\{t_n\}$ *and* $\{r_n\}$ *are sequences of nonnegative real numbers such that*

$$t_{n+1} \le (1-\sigma_n)t_n + s_n + r_n, \quad n \ge 0,$$

where $\{\sigma_n\}$ *is a sequence in* $(0,1)$ *and* $\{s_n\}$ *is a real sequence. Let* $\sum_{n=1}^{\infty} r_n < \infty$ *and* $s_n \le \sigma_n M$ *for some* $M \ge 0$. *Then* $\{t_n\}$ *is bounded.*

3 Main Results

In this section, we propose a Halpern-type proximal point algorithm with two-step inertial extrapolation and obtain strong convergence results.

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) The solution set $A^{-1}(\mathbf{0})$ of inclusion problem (1) is nonempty.

We furthermore assume the following conditions on iterative parameters are satisfied.

Assumption 3.2.

(i) Choose $\theta \in \left(0, \frac{1}{3}\right]$ and then choose δ such that $\max\left\{-\frac{\theta}{2}, \frac{3\theta-1}{3(2\theta+1)}\right\} < \delta \leq 0$;

(ii) With the choices of θ and δ in (i) above, compute $\frac{\delta(2\theta+1)}{2\theta-2\theta\delta-\delta-\frac{2}{3}}$. Now choose $\{\alpha_n\}$ in $\left(0, 1 - \frac{\delta(2\theta+1)}{2\theta-2\theta\delta-\delta-\frac{2}{3}}\right)$ such that $\lim_{n\to\infty} \alpha_n = 0$, $\sum_{n=0}^{\infty} \alpha_n = \infty$.

Remark 3.3. Observe that by the choices $\theta \in \left(0, \frac{1}{3}\right]$ and $\max\left\{-\frac{\theta}{2}, \frac{3\theta-1}{3(2\theta+1)}\right\} < \delta \leq 0$, we have

$$0 < \frac{\delta(2\theta+1)}{2\theta-2\theta\delta-\delta-\frac{2}{3}} < 1.$$

We now present our proposed method as follows:

Algorithm 1 (Strongly Convergent 2-Step Inertial PPA)

1: Choose parameters $\{\alpha_n\}, \delta$ and θ satisfying Assumption 3.2. Choose $x_{-1}, x_0, y_0 \in H$ arbitrarily, $\lambda > 0$ and set $n = 0$.

2: Given x_{n-1}, x_n and y_n, compute x_{n+1} as follows:

$$\begin{cases} x_{n+1} = J_\lambda^A(\alpha_n x_0 + (1-\alpha_n)y_n), \\ y_{n+1} = x_{n+1} + \theta(x_{n+1} - x_n) + \delta(x_n - x_{n-1}) \end{cases} \tag{14}$$

3: Set $n \leftarrow n+1$, and **go to Step 2**.

Remark 3.4. (i) Our proposed Algorithm 1 reduces to (11), which has been studied in [15, 16, 19, 53, 54, 59]), when $\delta = 0$ in Algorithm 1. Similarly, when $\theta = \delta = 0$ in Algorithm 1, we obtain (7) studied in [7].

(ii) Our Assumption 3.2 is different from on-line rule (12) which is used to obtain strong convergence results of (11) in [15, 16, 19, 53, 54, 59]).

3.1 Convergence Analysis

We present the weak convergence analysis of sequence of iterates generated by our proposed Algorithm 1 in this subsection.

Lemma 3.5. *Assume that Assumption 3.1 and Assumption 3.2 are fulfilled. Then* $\{x_n\}$ *generated by Algorithm 1 is bounded.*

Proof. By $x_{n+1} = J_\lambda^A(\alpha_n x_0 + (1-\alpha_n)y_n)$, we obtain

$$\alpha_n x_0 + (1-\alpha_n)y_n - x_{n+1} \in \lambda A x_{n+1}. \tag{15}$$

Let $x^* \in A^{-1}(\mathbf{0})$. Then by the monotonicity of A, we obtain

$$\langle \alpha_n x_0 + (1-\alpha_n)y_n - x_{n+1}, x_{n+1} - x^* \rangle \geq 0.$$

Thus,

$$\begin{aligned} 0 &\leq 2\langle x_{n+1} - (\alpha_n x_0 + (1-\alpha_n)y_n), x^* - x_{n+1} \rangle \\ &= 2\langle x_{n+1} - a_n, x^* - x_{n+1} \rangle \\ &= \|a_n - x^*\|^2 - \|x_{n+1} - x^*\|^2 - \|x_{n+1} - a_n\|^2, \end{aligned} \tag{16}$$

where $a_n := \alpha_n x_0 + (1-\alpha_n)y_n$. Therefore,

$$\|x_{n+1} - x^*\|^2 \leq \|a_n - x^*\|^2 - \|x_{n+1} - a_n\|^2. \tag{17}$$

By Lemma 2.3, we have

$$\begin{aligned} \|a_n - x^*\|^2 &= \|\alpha_n x_0 + (1-\alpha_n)y_n - x^*\|^2 \\ &= \|(y_n - x^*) - \alpha_n(y_n - x_0)\|^2 \\ &= \|y_n - x^*\|^2 + \alpha_n^2\|y_n - x_0\|^2 - 2\alpha_n\langle y_n - x^*, y_n - x_0\rangle \\ &= \|y_n - x^*\|^2 + \alpha_n^2\|y_n - x_0\|^2 - \alpha_n\|y_n - x_0\|^2 \\ &\quad -\alpha_n\|y_n - x^*\|^2 + \alpha_n\|x_0 - x^*\|^2. \end{aligned} \tag{18}$$

Similarly,

$$\begin{aligned} \|x_{n+1} - a_n\|^2 &= \|a_n - x_{n+1}\|^2 \\ &= \|y_n - x_{n+1}\|^2 + \alpha_n^2\|y_n - x_0\|^2 - \alpha_n\|y_n - x_0\|^2 \\ &\quad -\alpha_n\|y_n - x_{n+1}\|^2 + \alpha_n\|x_0 - x_{n+1}\|^2. \end{aligned} \tag{19}$$

Putting (19) and (18) in (17), we have

$$\begin{aligned} \|x_{n+1} - x^*\|^2 &\leq (1-\alpha_n)\|y_n - x^*\|^2 + \alpha_n\|x_0 - x^*\|^2 \\ &\quad -(1-\alpha_n)\|y_n - x_{n+1}\|^2 - \alpha_n\|x_{n+1} - x_0\|^2 \\ &\leq (1-\alpha_n)\|y_n - x^*\|^2 + \alpha_n\|x_0 - x^*\|^2 \\ &\quad -(1-\alpha_n)\|y_n - x_{n+1}\|^2. \end{aligned} \tag{20}$$

From Lemma 2.4, we have

$$\|y_n - x^*\|^2 = \|(1+\theta)(x_n - x^*) - (\theta - \delta)(x_{n-1} - x^*) - \delta(x_{n-2} - x^*)\|^2$$

$$
\begin{aligned}
&= (1+\theta)\|x_n - x^*\|^2 - (\theta-\delta)\|x_{n-1} - x^*\|^2 - \delta\|x_{n-2} - x^*\|^2 \\
&\quad +(1+\theta)(\theta-\delta)\|x_n - x_{n-1}\|^2 + \delta(1+\theta)\|x_n - x_{n-2}\|^2 \\
&\quad -\delta(\theta-\delta)\|x_{n-1} - x_{n-2}\|^2. \qquad (21)
\end{aligned}
$$

Note that

$$
-2\theta\langle x_{n+1} - x_n, x_n - x_{n-1}\rangle \geq -2\theta\|x_{n+1} - x_n\|\|x_n - x_{n-1}\|, \qquad (22)
$$

$$
-2\delta\langle x_{n+1} - x_n, x_{n-1} - x_{n-2}\rangle \geq -2|\delta|\|x_{n+1} - x_n\|\|x_{n-1} - x_{n-2}\|, \qquad (23)
$$

and

$$
\begin{aligned}
2\delta\theta\langle x_n - x_{n-1}, x_{n-1} - x_{n-2}\rangle &= -2\delta\theta\langle x_{n-1} - x_n, x_{n-1} - x_{n-2}\rangle \\
&\geq -2|\delta|\theta\|x_n - x_{n-1}\|\|x_{n-1} - x_{n-2}\|. \qquad (24)
\end{aligned}
$$

Using (22)- (24), we have

$$
\begin{aligned}
\|x_{n+1} - y_n\|^2 &= \|x_{n+1} - (x_n + \theta(x_n - x_{n-1}) + \delta(x_{n-1} - x_{n-2}))\|^2 \\
&= \|x_{n+1} - x_n - \theta(x_n - x_{n-1}) - \delta(x_{n-1} - x_{n-2})\|^2 \\
&= \|x_{n+1} - x_n\|^2 - 2\theta\langle x_{n+1} - x_n, x_n - x_{n-1}\rangle \\
&\quad -2\delta\langle x_{n+1} - x_n, x_{n-1} - x_{n-2}\rangle + \theta^2\|x_n - x_{n-1}\|^2 \\
&\quad +2\delta\theta\langle x_n - x_{n-1}, x_{n-1} - x_{n-2}\rangle + \delta^2\|x_{n-1} - x_{n-2}\|^2 \\
&\geq \|x_{n+1} - x_n\|^2 - 2\theta\|x_{n+1} - x_n\|\|x_n - x_{n-1}\| \\
&\quad -2|\delta|\|x_{n+1} - x_n\|\|x_{n-1} - x_{n-2}\| + \theta^2\|x_n - x_{n-1}\|^2 \\
&\quad -2|\delta|\theta\|x_n - x_{n-1}\|\|x_{n-1} - x_{n-2}\| + \delta^2\|x_{n-1} - x_{n-2}\|^2 \\
&\geq \|x_{n+1} - x_n\|^2 - \theta\|x_{n+1} - x_n\|^2 - \theta\|x_n - x_{n-1}\|^2 \\
&\quad -|\delta|\|x_{n+1} - x_n\|^2 - |\delta|\|x_{n-1} - x_{n-2}\|^2 + \theta^2\|x_n - x_{n-1}\|^2 \\
&\quad -|\delta|\theta\|x_n - x_{n-1}\|^2 - |\delta|\theta\|x_{n-1} - x_{n-2}\|^2 + \delta^2\|x_{n-1} - x_{n-2}\|^2 \\
&= (1-|\delta|-\theta)\|x_{n+1} - x_n\|^2 + (\theta^2 - \theta - |\delta|\theta)\|x_n - x_{n-1}\|^2 \\
&\quad +(\delta^2 - |\delta| - |\delta|\theta)\|x_{n-1} - x_{n-2}\|^2. \qquad (25)
\end{aligned}
$$

Using (21) and (25) in (20), we get

$$
\begin{aligned}
\|x_{n+1} - x^*\|^2 &\leq (1-\alpha_n)\Big[(1+\theta)\|x_n - x^*\|^2 - (\theta-\delta)\|x_{n-1} - x^*\|^2 \\
&\quad -\delta\|x_{n-2} - x^*\|^2 + (1+\theta)(\theta-\delta)\|x_n - x_{n-1}\|^2 + \delta(1+\theta)\|x_n - x_{n-2}\|^2 \\
&\quad -\delta(\theta-\delta)\|x_{n-1} - x_{n-2}\|^2\Big] + \alpha_n\|x_0 - x^*\|^2 \\
&\quad -(1-\alpha_n)\Big[(1-|\delta|-\theta)\|x_{n+1} - x_n\|^2 + (\theta^2 - \theta - |\delta|\theta)\|x_n - x_{n-1}\|^2 \\
&\quad +(\delta^2 - |\delta| - |\delta|\theta)\|x_{n-1} - x_{n-2}\|^2\Big] \\
&= (1-\alpha_n)(1+\theta)\|x_n - x^*\|^2 - (1-\alpha_n)(\theta-\delta)\|x_{n-1} - x^*\|^2 \\
&\quad -\delta(1-\alpha_n)\|x_{n-2} - x^*\|^2 + (1-\alpha_n)(1+\theta)(\theta-\delta)\|x_n - x_{n-1}\|^2 \\
&\quad +\delta(1+\theta)(1-\alpha_n)\|x_n - x_{n-2}\|^2 - \delta(\theta-\delta)(1-\alpha_n)\|x_{n-1} - x_{n-2}\|^2 \\
&\quad +\alpha_n\|x_0 - x^*\|^2 - (1-\alpha_n)(1-|\delta|-\theta)\|x_{n+1} - x_n\|^2
\end{aligned}
$$

$$
\begin{aligned}
& -(1-\alpha_n)(\theta^2-\theta-|\delta|\theta)\|x_n-x_{n-1}\|^2 \\
& -(1-\alpha_n)(\delta^2-|\delta|-|\delta|\theta)\|x_{n-1}-x_{n-2}\|^2\Big] \\
\leq\ & (1-\alpha_n)(1+\theta)\|x_n-x^*\|^2-(1-\alpha_n)(\theta-\delta)\|x_{n-1}-x^*\|^2 \\
& -\delta(1-\alpha_n)\|x_{n-2}-x^*\|^2+(1-\alpha_n)(1+\theta)(\theta-\delta)\|x_n-x_{n-1}\|^2 \\
& -\delta(\theta-\delta)(1-\alpha_n)\|x_{n-1}-x_{n-2}\|^2+\alpha_n\|x_0-x^*\|^2 \\
& -(1-\alpha_n)(1+\delta-\theta)\|x_{n+1}-x_n\|^2 \\
& -(1-\alpha_n)(\theta^2-\theta+\delta\theta)\|x_n-x_{n-1}\|^2 \\
& -(1-\alpha_n)(\delta^2+|\delta|+\delta\theta)\|x_{n-1}-x_{n-2}\|^2. \qquad (26)
\end{aligned}
$$

This implies that

$$
\begin{aligned}
& \|x_{n+1}-x^*\|^2-\theta\|x_n-x^*\|^2-\delta\|x_{n-1}-x^*\|^2 \\
\leq\ & (1-\alpha_n)\Big[\|x_n-x^*\|^2-\theta\|x_{n-1}-x^*\|^2-\delta\|x_{n-2}-x^*\|^2\Big] \\
& +(1-\alpha_n)(1+\theta)(\theta-\delta)\|x_n-x_{n-1}\|^2-\delta(\theta-\delta)(1-\alpha_n)\|x_{n-1}-x_{n-2}\|^2 \\
& +\alpha_n\|x_0-x^*\|^2-(1-\alpha_n)(1+\delta-\theta)\|x_{n+1}-x_n\|^2 \\
& -(1-\alpha_n)(\theta^2-\theta+\delta\theta)\|x_n-x_{n-1}\|^2 \\
& -(1-\alpha_n)(\delta^2+|\delta|+\delta\theta)\|x_{n-1}-x_{n-2}\|^2 \\
=\ & (1-\alpha_n)\Big[\|x_n-x^*\|^2-\theta\|x_{n-1}-x^*\|^2-\delta\|x_{n-2}-x^*\|^2\Big] \\
& +(1-\alpha_n)(2\theta-2\theta\delta-\delta)\|x_n-x_{n-1}\|^2+\alpha_n\|x_0-x^*\|^2 \\
& -(1-\alpha_n)(1+\delta-\theta)\|x_{n+1}-x_n\|^2 \\
& -(1-\alpha_n)(2\delta\theta+\delta)\|x_{n-1}-x_{n-2}\|^2. \qquad (27)
\end{aligned}
$$

Thus,

$$
\begin{aligned}
& \|x_{n+1}-x^*\|^2-\theta\|x_n-x^*\|^2-\delta\|x_{n-1}-x^*\|^2+\frac{2}{3}\|x_{n+1}-x_n\|^2 \\
\leq\ & (1-\alpha_n)\Big[\|x_n-x^*\|^2-\theta\|x_{n-1}-x^*\|^2-\delta\|x_{n-2}-x^*\|^2 \\
& +\frac{2}{3}\|x_n-x_{n-1}\|^2\Big]+(1-\alpha_n)\Big(2\theta-2\theta\delta-\delta-\frac{2}{3}\Big)\|x_n-x_{n-1}\|^2 \\
& +\alpha_n\|x_0-x^*\|^2-\Big((1-\alpha_n)(1+\delta-\theta)-\frac{2}{3}\Big)\|x_{n+1}-x_n\|^2 \\
& -(1-\alpha_n)(2\delta\theta+\delta)\|x_{n-1}-x_{n-2}\|^2. \qquad (28)
\end{aligned}
$$

Observe that

$$
\liminf_{n\to\infty}\Big((1-\alpha_n)(1+\delta-\theta)-\frac{2}{3}\Big)=1+\delta-\theta-\frac{2}{3}>0,
$$

since $\frac{3\theta-1}{3}<\delta$. Observe that $\frac{3\theta-1}{3}\leq\frac{3\theta-1}{3(2\theta+1)}<\delta$ when $0\leq\theta<\frac{1}{3}$. Therefore, there exists $n_1\in\mathbb{N}$ such that $\forall n\geq n_1$, we have

$$
(1-\alpha_n)(1+\delta-\theta)-\frac{2}{3}>0.
$$

Hence, (28) becomes

$$\begin{aligned}
& \|x_{n+1}-x^*\|^2-\theta\|x_n-x^*\|^2-\delta\|x_{n-1}-x^*\|^2+\frac{2}{3}\|x_{n+1}-x_n\|^2 \\
\leq\ & (1-\alpha_n)\Big[\|x_n-x^*\|^2-\theta\|x_{n-1}-x^*\|^2-\delta\|x_{n-2}-x^*\|^2 \\
& +\frac{2}{3}\|x_n-x_{n-1}\|^2\Big]+(1-\alpha_n)\Big(2\theta-2\theta\delta-\delta-\frac{2}{3}\Big)\|x_n-x_{n-1}\|^2 \\
& +\alpha_n\|x_0-x^*\|^2-(1-\alpha_n)(2\delta\theta+\delta)\|x_{n-1}-x_{n-2}\|^2, \qquad (29)
\end{aligned}$$

which then implies that

$$\begin{aligned}
& \|x_{n+1}-x^*\|^2-\theta\|x_n-x^*\|^2-\delta\|x_{n-1}-x^*\|^2+\frac{2}{3}\|x_{n+1}-x_n\|^2 \\
& -(1-\alpha_n)\Big(2\theta-2\theta\delta-\delta-\frac{2}{3}\Big)\|x_n-x_{n-1}\|^2 \\
\leq\ & (1-\alpha_n)\Big[\|x_n-x^*\|^2-\theta\|x_{n-1}-x^*\|^2-\delta\|x_{n-2}-x^*\|^2 \\
& +\frac{2}{3}\|x_n-x_{n-1}\|^2-(2\delta\theta+\delta)\|x_{n-1}-x_{n-2}\|^2\Big] \\
& +\alpha_n\|x_0-x^*\|^2. \qquad (30)
\end{aligned}$$

Since $\alpha_n < 1-\frac{\delta(2\theta+1)}{2\theta-2\theta\delta-\delta-\frac{2}{3}}$, we have that

$$-(2\delta\theta+\delta) < -(1-\alpha_n)\Big(2\theta-2\theta\delta-\delta-\frac{2}{3}\Big).$$

We then obtain from (30) that

$$\begin{aligned}
& \|x_{n+1}-x^*\|^2-\theta\|x_n-x^*\|^2-\delta\|x_{n-1}-x^*\|^2+\frac{2}{3}\|x_{n+1}-x_n\|^2 \\
& -(2\delta\theta+\delta)\|x_n-x_{n-1}\|^2 \\
\leq\ & (1-\alpha_n)\Big[\|x_n-x^*\|^2-\theta\|x_{n-1}-x^*\|^2-\delta\|x_{n-2}-x^*\|^2 \\
& +\frac{2}{3}\|x_n-x_{n-1}\|^2-(2\delta\theta+\delta)\|x_{n-1}-x_{n-2}\|^2\Big] \\
& +\alpha_n\|x_0-x^*\|^2. \qquad (31)
\end{aligned}$$

Define for each $n \geq n_1$,

$$t_n := \|x_n-x^*\|^2-\theta\|x_{n-1}-x^*\|^2-\delta\|x_{n-2}-x^*\|^2+\frac{2}{3}\|x_n-x_{n-1}\|^2-(2\delta\theta+\delta)\|x_{n-1}-x_{n-2}\|^2.$$

We show that $t_n \geq 0, \quad \forall n \geq n_1$. Observe that

$$\|x_{n-1}-x^*\|^2 \leq 2\|x_n-x_{n-1}\|^2+2\|x_n-x^*\|^2. \qquad (32)$$

So,

$$\begin{aligned}
t_n \ =\ & \|x_n-x^*\|^2-\theta\|x_{n-1}-x^*\|^2-\delta\|x_{n-2}-x^*\|^2 \\
& +\frac{2}{3}\|x_n-x_{n-1}\|^2-(2\delta\theta+\delta)\|x_{n-1}-x_{n-2}\|^2
\end{aligned}$$

$$
\begin{aligned}
&\geq \|x_n - x^*\|^2 - 2\theta\|x_n - x_{n-1}\|^2 - 2\theta\|x_n - x^*\|^2 \\
&\quad -\delta\|x_{n-2} - x^*\|^2 + \frac{2}{3}\|x_n - x_{n-1}\|^2 \\
&\quad -(2\delta\theta + \delta)\|x_{n-1} - x_{n-2}\|^2 \\
&= (1 - 2\theta)\|x_n - x^*\|^2 + (\frac{2}{3} - 2\theta)\|x_n - x_{n-1}\|^2 \\
&\quad -\delta\|x_{n-2} - x^*\|^2 - (2\delta\theta + \delta)\|x_{n-1} - x_{n-2}\|^2 \\
&\geq 0, \qquad (33)
\end{aligned}
$$

since $0 \leq \theta < \frac{1}{3}$ and $\delta \leq 0$. We then obtain from (31) that

$$
t_{n+1} \leq (1 - \alpha_n)t_n + \alpha_n\|x_0 - x^*\|^2. \qquad (34)
$$

By Lemma 2.6, we have that the sequence $\{t_n\}$ is bounded. Consequently, from (33) that $\{x_n\}$ is bounded. □

We give our strong convergence result in the next result.

Theorem 3.6. *Suppose $\{x_n\}$ is generated by Algorithm 1. Then, $\{x_n\}$ converges strongly to $P_{A^{-1}(\mathbf{0})}(x_0)$ when Assumption 3.1 and Assumption 3.2 are satisfied.*

Proof. Suppose $x^* = P_{A^{-1}(\mathbf{0})}(x_0)$. Then by Lemma 2.3, we have

$$
\begin{aligned}
\|a_n - x^*\|^2 &= \|\alpha_n x_0 + (1 - \alpha_n)y_n - x^*\|^2 \\
&= \|\alpha_n(x_0 - x^*) + (1 - \alpha_n)(y_n - x^*)\|^2 \\
&= \alpha_n^2\|x_0 - x^*\|^2 + (1 - \alpha_n)^2\|y_n - x^*\|^2 \\
&\quad +2\alpha_n(1 - \alpha_n)\langle x_0 - x^*, y_n - x^*\rangle. \qquad (35)
\end{aligned}
$$

Also,

$$
\begin{aligned}
\|x_{n+1} - a_n\|^2 &= \alpha_n^2\|x_0 - x_{n+1}\|^2 + (1 - \alpha_n)^2\|y_n - x_{n+1}\|^2 \\
&\quad +2\alpha_n(1 - \alpha_n)\langle x_0 - x_{n+1}, y_n - x_{n+1}\rangle \\
&\geq \alpha_n^2\|x_0 - x_{n+1}\|^2 + (1 - \alpha_n)^2\|y_n - x_{n+1}\|^2 \\
&\quad -2\alpha_n(1 - \alpha_n)\|x_0 - x_{n+1}\|\|y_n - x_{n+1}\| \\
&\geq \alpha_n^2\|x_{n+1} - x_0\|^2 + (1 - \alpha_n)^2\|x_{n+1} - y_n\|^2 \\
&\quad -2\alpha_n(1 - \alpha_n)M_1\|x_{n+1} - y_n\|, \qquad (36)
\end{aligned}
$$

where $M_1 := \sup_{n \geq n_1}\|x_{n+1} - x_0\| < \infty$, since $\{x_n\}$ is bounded. Using (35) and (36) in (17)

$$
\begin{aligned}
\|x_{n+1} - x^*\|^2 &\leq \alpha_n^2\|x_0 - x^*\|^2 + (1 - \alpha_n)^2\|y_n - x^*\|^2 \\
&\quad +2\alpha_n(1 - \alpha_n)\langle x_0 - x^*, y_n - x^*\rangle \\
&\quad -\alpha_n^2\|x_{n+1} - x_0\|^2 - (1 - \alpha_n)^2\|x_{n+1} - y_n\|^2 \\
&\quad +2\alpha_n(1 - \alpha_n)M_1\|x_{n+1} - y_n\| \\
&\leq (1 - \alpha_n)\|y_n - x^*\|^2 + \alpha_n\Big(\alpha_n\|x_0 - x^*\|^2 \\
&\quad +2(1 - \alpha_n)\langle x_0 - x^*, y_n - x^*\rangle
\end{aligned}
$$

$$
\begin{aligned}
&+2(1-\alpha_n)M_1\|x_{n+1}-y_n\|\Big) \\
&-(1-\alpha_n)^2\|x_{n+1}-y_n\|^2. \qquad (37)
\end{aligned}
$$

Using (21) and (25) in (37) gives us

$$
\begin{aligned}
\|x_{n+1}-x^*\|^2 \ \leq\ & (1-\alpha_n)\Big[(1+\theta)\|x_n-x^*\|^2-(\theta-\delta)\|x_{n-1}-x^*\|^2 \\
&+(1+\theta)(\theta-\delta)\|x_n-x_{n-1}\|^2+\delta(1+\theta)\|x_n-x_{n-2}\|^2 \\
&-\delta(\theta-\delta)\|x_{n-1}-x_{n-2}\|^2-\delta\|x_{n-2}-x^*\|^2\Big] \\
&+\alpha_n\Big(\alpha_n\|x_0-x^*\|^2+2(1-\alpha_n)\langle x_0-x^*, y_n-x^*\rangle \\
&+2(1-\alpha_n)M_1\|x_{n+1}-y_n\|\Big)-(1-\alpha_n)^2\Big[(1+\delta-\theta)\|x_{n+1}-x_n\|^2 \\
&+(\theta^2-\theta+\delta\theta)\|x_n-x_{n-1}\|^2 \\
&+(\delta^2+\delta+\delta\theta)\|x_{n-1}-x_{n-2}\|^2\Big]. \qquad (38)
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
& \|x_{n+1}-x^*\|^2-\theta\|x_n-x^*\|^2-\delta\|x_{n-1}-x^*\|^2+\frac{2}{3}\|x_{n+1}-x_n\|^2 \\
& -(2\delta\theta+\delta)\|x_n-x_{n-1}\|^2 \\
\leq\ & (1-\alpha_n)\Big[\|x_n-x^*\|^2-\theta\|x_{n-1}-x^*\|^2-\delta\|x_{n-2}-x^*\|^2 \\
& +\frac{2}{3}\|x_n-x_{n-1}\|^2-(2\delta\theta+\delta)\|x_{n-1}-x_{n-2}\|^2\Big] \\
& +\alpha_n\Big(\alpha_n\|x_0-x^*\|^2+2(1-\alpha_n)\langle x_0-x^*, y_n-x^*\rangle \\
& +2(1-\alpha_n)M_1\|x_{n+1}-y_n\|\Big)-\theta\alpha_n\|x_n-x^*\|^2 \\
& -\delta\alpha_n\|x_{n-1}-x^*\|^2+(1-\alpha_n)\Big((1+\theta)(\theta-\delta)-\frac{2}{3}\Big)\|x_n-x_{n-1}\|^2 \\
& -(2\delta\theta+\delta)\|x_n-x_{n-1}\|^2-(1-\alpha_n)^2(\theta^2-\theta+\delta\theta)\|x_n-x_{n-1}\|^2 \\
& +(1-\alpha_n)\Big((2\delta\theta+\delta)-\delta(\theta-\delta)\Big)\|x_{n-1}-x_{n-2}\|^2 \\
& -(1-\alpha_n)^2(\delta^2+\delta+\delta\theta)\|x_{n-1}-x_{n-2}\|^2 \\
& +\Big(\frac{2}{3}-(1-\alpha_n)^2(1+\delta-\theta)\Big)\|x_{n+1}-x_n\|^2 \\
\leq\ & (1-\alpha_n)t_n+\alpha_n h_n \\
& +\Big[(1-\alpha_n)\Big((1+\theta)(\theta-\delta)-\frac{2}{3}\Big) \\
& -(2\delta\theta+\delta)-(1-\alpha_n)^2(\theta^2-\theta+\delta\theta)-2\delta\alpha_n\Big]\|x_n-x_{n-1}\|^2 \\
& +(1-\alpha_n)\Big[2\delta\theta+\delta-\delta(\theta-\delta)-(1-\alpha_n)(\delta^2+\delta+\delta\theta)\Big]\|x_{n-1}-x_{n-2}\|^2 \\
& +\Big(\frac{2}{3}-(1-\alpha_n)^2(1+\delta-\theta)\Big)\|x_{n+1}-x_n\|^2 \\
& +(-2\delta\alpha_n-\theta\alpha_n)\|x_n-x^*\|^2, \qquad (39)
\end{aligned}
$$

where

$$
\begin{aligned}
h_n : &= \alpha_n\|x_0 - x^*\|^2 + 2(1-\alpha_n)\langle x_0 - x^*, y_n - x^*\rangle \\
& +2(1-\alpha_n)M_1\|x_{n+1} - y_n\|.
\end{aligned}
$$

Since $-\frac{\theta}{2} \leq \delta$, we have

$$
-2\delta\alpha_n - \theta\alpha_n \leq 0, \quad \forall n \geq n_1. \tag{40}
$$

Observe that

$$
\lim_{n\to\infty}\left[\frac{2}{3} - (1-\alpha_n)^2(1+\delta-\theta)\right] = \frac{2}{3} - (1+\delta-\theta) \leq 0,
$$

since $\frac{3\theta-1}{3} \leq \delta$. Also,

$$
\begin{aligned}
& \lim_{n\to\infty}\left[2\delta\theta + \delta - \delta(\theta-\delta) - (1-\alpha_n)(\delta^2+\delta+\delta\theta)\right] \\
= \; & 2\delta\theta + \delta - \delta(\theta-\delta) - (\delta^2+\delta+\delta\theta) = 0.
\end{aligned} \tag{41}
$$

Furthermore,

$$
\begin{aligned}
& \lim_{n\to\infty}\Big[(1-\alpha_n)\Big((1+\theta)(\theta-\delta) - \frac{2}{3}\Big) \\
& -(2\delta\theta+\delta) - (1-\alpha_n)^2(\theta^2-\theta+\delta\theta) - 2\delta\alpha_n\Big] \\
= \; & (1+\theta)(\theta-\delta) - \frac{2}{3} - (2\delta\theta+\delta) \\
& -(\theta^2-\theta+\delta\theta) < 0,
\end{aligned} \tag{42}
$$

since $\frac{3\theta-2}{18(1-2\theta)} < \delta$ because $\frac{3\theta-2}{18(1-2\theta)} < -\frac{\theta}{2} < \delta$ when $0 \leq \theta < \frac{1}{3}$. Using (40)- (42) in (39), there exists $n_2 \geq n_1 \in \mathbb{N}$ such that for all $n \geq n_2 \geq n_1$,

$$
\begin{aligned}
t_{n+1} \leq \; & (1-\alpha_n)t_n + \alpha_n h_n \\
& +\Big[(1-\alpha_n)\Big((1+\theta)(\theta-\delta) - \frac{2}{3}\Big) \\
& -(2\delta\theta+\delta) - (1-\alpha_n)^2(\theta^2-\theta+\delta\theta)\Big]\|x_n - x_{n-1}\|^2.
\end{aligned} \tag{43}
$$

To conclude the proof, it suffices to show, in view of Lemma 2.5 that $\limsup_{i\to\infty} h_{n_i} \leq 0$ for each subsequence $\{t_{n_i}\} \subset \{t_n\}$ such that $\liminf_{i\to\infty}(t_{n_i+1} - t_{n_i}) \geq 0$. To this end, let $\{t_{n_i}\}$ be a subsequence of $\{t_n\}$ such that $\liminf_{i\to\infty}(t_{n_i+1} - t_{n_i}) \geq 0$. From (43), we obtain

$$
\begin{aligned}
& \limsup_{i\to\infty}\Big[-(1-\alpha_{n_i})\Big((1+\theta)(\theta-\delta) - \frac{2}{3}\Big) \\
& +(2\delta\theta+\delta) + (1-\alpha_{n_i})^2(\theta^2-\theta+\delta\theta)\Big]\|x_{n_i} - x_{n_i-1}\|^2 \\
\leq \; & \limsup_{i\to\infty}[(t_{n_i} - t_{n_i+1}) + \alpha_{n_i}(h_{n_i} - t_{n_i})] \\
\leq \; & -\liminf_{i\to\infty}(t_{n_i+1} - t_{n_i}) \leq 0.
\end{aligned}
$$

Since

$$\begin{aligned}
&\lim_{i\to\infty}\Big[-(1-\alpha_{n_i})\Big((1+\theta)(\theta-\delta)-\frac{2}{3}\Big)\\
&+(2\delta\theta+\delta)+(1-\alpha_{n_i})^2(\theta^2-\theta+\delta\theta)\Big]\\
>\ & 0,
\end{aligned}$$

we obtain

$$\lim_{i\to\infty}\|x_{n_i}-x_{n_i-1}\|=0. \tag{44}$$

Consequently,

$$\|y_{n_i}-x_{n_i}\|\le\theta\|x_{n_i}-x_{n_i-1}\|+|\delta|\|x_{n_i-1}-x_{n_i-2}\|\to 0, i\to\infty. \tag{45}$$

Also, we obtain

$$\lim_{i\to\infty}\|x_{n_i+1}-y_{n_i}\|=0. \tag{46}$$

By Lemma 3.5, $\{x_{n_i}\}$ is bounded. Therefore, we can choose a subsequence $\{x_{n_{i_j}}\}\subset\{x_{n_i}\}$ which converges weakly to some $z\in H$ such that

$$\begin{aligned}
\limsup_{i\to\infty}\langle x_0-x^*,x_{n_i}-x^*\rangle &= \lim_{j\to\infty}\langle x_0-x^*,x_{n_{i_j}}-x^*\rangle\\
&= \langle x_0-x^*,z-x^*\rangle.
\end{aligned} \tag{47}$$

Let $(x,y)\in G(A)$. Then $y\in Ax$. So, we have from (15) that

$$\langle\lambda y-(\alpha_{n_{i_j}}x_0+(1-\alpha_{n_{i_j}})y_{n_{i_j}}-x_{n_{i_j}+1},x-x_{n_{i_j}+1}\rangle\ge 0$$

which implies that

$$\begin{aligned}
\langle y,x-x_{n_{i_j}+1}\rangle &\ge \frac{1}{\lambda}\langle\alpha_{n_{i_j}}x_0+(1-\alpha_{n_{i_j}}y_{n_{i_j}}-x_{n_{i_j}+1},x-x_{n_{i_j}+1}\rangle\\
&= \frac{1}{\lambda}\langle\alpha_{n_{i_j}}(x_0-y_{n_{i_j}})+y_{n_{i_j}}-x_{n_{i_j}+1},x-x_{n_{i_j}+1}\rangle.
\end{aligned} \tag{48}$$

As $j\to\infty$ in (48), we get from (48) (using (46)) that $\langle y,x-z\rangle\ge 0$. Since A is maximal monotone, we have that $z\in A^{-1}(\mathbf{0})$.

Since $x^*=P_{A^{-1}(\mathbf{0})(x_0)}$, we have from (47) that

$$\limsup_{i\to\infty}\langle x_0-x^*,x_{n_i}-x^*\rangle=\langle x_0-x^*,z-x^*\rangle\le 0. \tag{49}$$

Therefore,

$$\limsup_{i\to\infty}\langle x_0-x^*,y_{n_i}-x^*\rangle\le 0$$

by (45) and (49). Hence,

$$\limsup_{i\to\infty}h_{n_i}\le 0.$$

Since $\sum_{n=1}^{\infty}\alpha_n=\infty$, we obtain by Lemma 2.5 in (39) that $\lim_{n\to\infty}t_n=0$. Using (33), we have that $\{x_n\}$ converges strongly to $P_{A^{-1}(\mathbf{0})}(x_0)$. This completes the proof. □

Remark 3.7. The on-line rule (12) assumed in [15,16,19,53,54,59] in order to obtain strong convergence of (11) was imposed so that one can have $\lim_{n\to\infty} \theta_n\|x_n - x_{n-1}\| = 0$, which is necessary for the convergence of (11). However, in our convergence analysis, we obtain a much weaker (44) without assuming on-line rule (12). Therefore, the new step-size rule given in Assumption 3.2 is useful. This is one of the state-of-the art contributions of our results.

4 Numerical Tests

The focus of this section is to provide some computational experiments to demonstrate the effectiveness, accuracy and easy-to-implement nature of our proposed algorithm. We further compare our proposed algorithm with some existing related methods in the literature. All codes were written in MATLAB R2020b and performed on a PC Desktop Intel(R) Core(TM) i7-6600U CPU @ 3.00GHz, RAM 32.00 GB.

Numerical comparisons are made with the algorithms proposed in [7, 19, 31].

Example 4.1. *Let us consider the following problem [6, 14, 31]:*

$$\begin{cases} \min_{x\in\mathbb{R}^N} \max_{z\in\mathbb{R}^M} \frac{1}{2}\|Fx-b\|^2 + \gamma\|z\|_1 \\ s.t. Dx - z = 0 \end{cases} \tag{50}$$

where $\gamma > 0$. This problem is associated with the total-variation-regularized least squares problem

$$\min_{x\in\mathbb{R}^N} \frac{1}{2}\|Fx-b\|^2 + \gamma\|Dx\|_1, \tag{51}$$

where $F \in \mathbb{R}^{p\times N}, b \in \mathbb{R}^p$, and a matrix $D \in \mathbb{R}^{M\times N}$ is given as

$$D = \begin{pmatrix} -1 & 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & -1 & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & 0 & 1 & -1 & 0 \\ 0 & \dots & \dots & 0 & 1 & -1 \end{pmatrix}$$

Let us take $f(x) := \frac{1}{2}\|Fx-b\|^2, g(z) := \gamma\|z\|_1$. Then using similar ideas as in [31, 43], we can convert Algorithm 1 to

Algorithm 2 (Modified Strongly Convergent 2-Step Inertial PPA)

1: Choose parameters δ and θ satisfying Assumption 3.2. Choose $x_0 \in \mathbb{R}^N, z_0 \in \mathbb{R}^M, \hat{v}_0 \in \mathbb{R}^M, \lambda > 0$. Set $n = 0$.

2: Compute as follows:

$$\begin{cases} x_{n+1} = \arg\min_{x \in \mathbb{R}^N} \left\{ \frac{1}{2}\|Fx - b\|^2 + \langle \alpha_n \hat{v}_0 + (1-\alpha_n)\hat{v}_n, Dx - z_n \rangle + \frac{\lambda}{2}\|Dx - z_n\|^2 \right\} \\ = (F^T F + \lambda D^T D)^{-1}(D^T(\lambda z_n - (\alpha_n \hat{v}_0 + (1-\alpha_n)\hat{v}_n)) + F^T b), \\ \hat{\eta}_n = \hat{v}_n, n = 0, 1, \\ \hat{\eta}_n = \hat{v}_n + \theta(\hat{v}_n - \hat{v}_{n-1} + \lambda D(x_{n+1} - x_n)) + \delta(\hat{v}_{n-1} - \hat{v}_{n-2} + \lambda D(x_n - x_{n-1})) \\ n = 2, 3, \dots \\ z_{n+1} = \arg\min_{z \in \mathbb{R}^M} \left\{ \gamma\|z\|_1 + \langle \hat{\eta}_n, Dx_{n+1} - z \rangle + \frac{\lambda}{2}\|Dx_{n+1} - z\|^2 \right\} \\ = S_{\frac{\gamma}{\lambda}}\left(Dx_{n+1} + \frac{1}{\lambda}\hat{\eta}_n \right) \\ \hat{v}_{n+1} = \hat{\eta}_n + \lambda(Dx_{n+1} - z_{n+1}), \end{cases} \tag{52}$$

3: Set $n \leftarrow n+1$, and **go to 2**.

where $S_\tau(z) := \max\{|z| - \tau, 0\} \odot \operatorname{sign}(z)$ *is the soft-thresholding operator, with the element-wise absolute value, maximum and multiplication operators,* $|.|, \max\{.,.\}$ *and* $\odot$*, respectively.*

4.1 Effects of parameters $(\theta, \delta, \alpha_n)$

In this subsection, we investigate the effects of the parameters θ, δ, α_n *on the performance of the proposed algorithm. The choice of the parameter are carefully chosen such that Condition 3.2 is satisfied. Moreover, we fixed some of the parameters and vary the others in order to determine the optimal choice for the performance of the algorithm. The numerical results of the algorithm is presented in Table 1. We used the size* $N = M = 5, p = 3, \lambda = 0.1, \gamma = 0.9$ *and* $\|x_{n+1} - x_n\| < 10^{-6}$ *as stopping criterion. In the numerical table, "Iter" denotes number of iteration, "CPU" denotes time taken for running the computation, "Err" denotes the residual value, i.e.,* $Err = \|x_{n+1} - x_n\|$*, "IRE" denotes the relative error, i.e.,* $IRE = \frac{\|x_n - y_n\|}{\max\{\|x_n\|, \|y_n\|\}}$ *and "Obj" is the optimal objective function vale, i.e.,* $Obj = f(x^*) + g(x^*)$*, where* x^* *is the value of* x_n *at the last iteration. The maximum iteration is set as 2000 and the initial points* x_0, x_{-1}, y_0 *are generated randomly. The algorithm is tested run with the following choices of parameters and the numerical results are shown in Table 1 and Figures 1-3:*

(i) We choose $\delta = -0.0001, \alpha_n = \frac{1}{10n+1}$ *and varies* θ *between 0.01 and 0.3. Note that* $-0.0208 < lb < -0.005$ *where* $lb = \max\left\{ -\frac{\theta}{2}, \frac{3\theta - 1}{3(2\theta+1)} \right\}$ *and* $0.9976 < ub < 0.9998$ *where* $ub = 1 - \frac{\delta(2\theta+1)}{2\theta - 2\theta\delta - \delta - \frac{2}{3}}$ *which ensured that Assumption 3.2 (i) and (ii) are satisfied.*

(ii) Also, we considered $\theta = 0.1 + \delta, \alpha_n = \frac{1}{10n+1}$ *and varies* δ *between -0.05 and*

-0.001. Note that in this case, $-0.0495 < lb < -0.0250$ *and* $0.1298 < ub < 0.9974$*. Hence Assumption 3.2 (1), (ii) and (iii) are satisfied.*

(iii) We choose $\delta = -0.01$, $\theta = 0.01$ *and* $\alpha_n = \frac{1}{(10n+1)^{1/q}}$*, where* $q = 1, \dots, 5$.

From the numerical results in Table 1 and Figures 1-3, we observed the following

- *The proposed algorithm performs better with fixed and small values of* θ*,* δ *and* α_n*.*
- *In optimization, optimal points which provide minimal objective function values are preferred. Though varying the values of* θ *and* δ *yield smaller value of objective function evaluation, the number of iteration and CPU time are higher which makes such choice not preferable. Also, we observed that the relative error value for varying choice of* θ *and* δ *is very high.*

Parameters	Iter	CPU	Err	IRE	Obj
Case 1: $(\delta = -0.01, \alpha_n = \frac{1}{10n+1})$					
$\theta = 0.01$	35	0.0123	7.8885E-7	6.9822E-5	0.5132
$\theta = 0.03$	97	0.0374	9.7502E-7	6.4824E-5	0.5363
$\theta = 0.08$	49	0.0170	8.8155E-7	3.8903E-4	0.4291
$\theta = 0.1$	109	0.0325	9.3124E-7	9.2642E-4	0.5811
$\theta = 0.3$	66	0.0205	7.4914E-7	5.3055E-6	0.6588
Case 2: $(\theta = 0.1 + \delta, \alpha_n = \frac{1}{10n+1})$					
$\delta = -0.001$	83	0.0762	9.7316E-7	1.4850E-6	0.9394
$\delta = -0.005$	248	0.0385	9.2249E-7	2.6475E-6	0.6327
$\delta = -0.009$	336	0.0447	9.9406E-7	1.4947E-6	0.4450
$\delta = -0.01$	64	0.0271	9.8790E-7	2.8061E-6	0.4927
$\delta = -0.05$	127	0.0356	9.5202E-7	3.2598E-5	0.5521
Case 3: $(\theta = 0.01, \delta = -0.01, \alpha_n = \frac{1}{(n+1)^{1/q}})$					
$q = 1$	83	0.0220	8.0047E-7	2.3593E-6	0.6125
$q = 2$	135	0.0517	9.7278E-7	4.5206E-5	0.6574
$q = 3$	51	0.0106	9.8897E-7	8.6457E-6	0.9525
$q = 4$	274	0.1538	9.7814E-7	9.1106E-6	0.5435
$q = 5$	590	0.8869	9.9865E-7	1.00	0.2543

Table 1: Numerical results of the proposed algorithm for various choices of θ, δ and α_n.

4.2 Comparing with other methods

Now, we would compare the performance of the proposed algorithm with algorithm proposed in [31], denoted by Kim Algorithm and the algorithm proposed in [7], denoted by BM algorithm for different cases for the choices of N*,* M*, and* p*. A true*

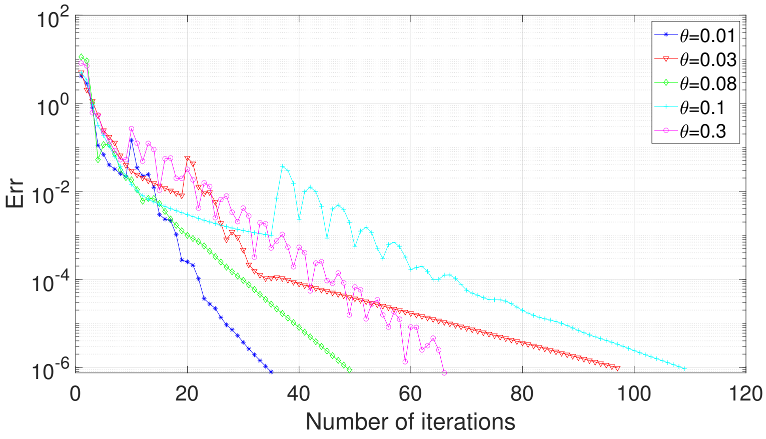

Figure 1: Comparison of parameters for the proposed algorithm; Case 1

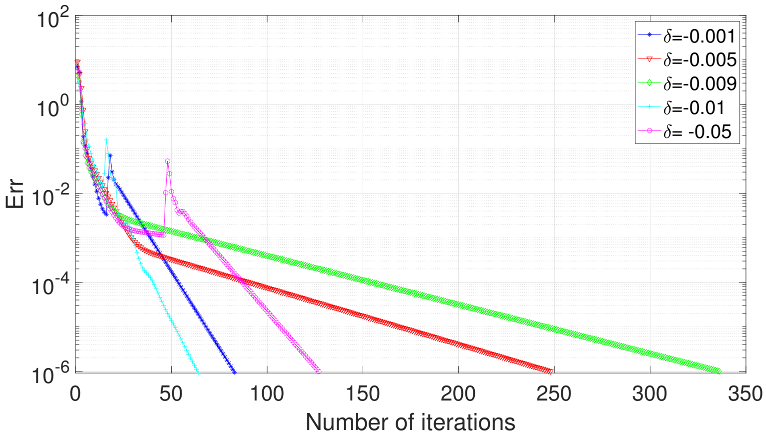

Figure 2: Comparison of parameters for the proposed algorithm; Case 2.

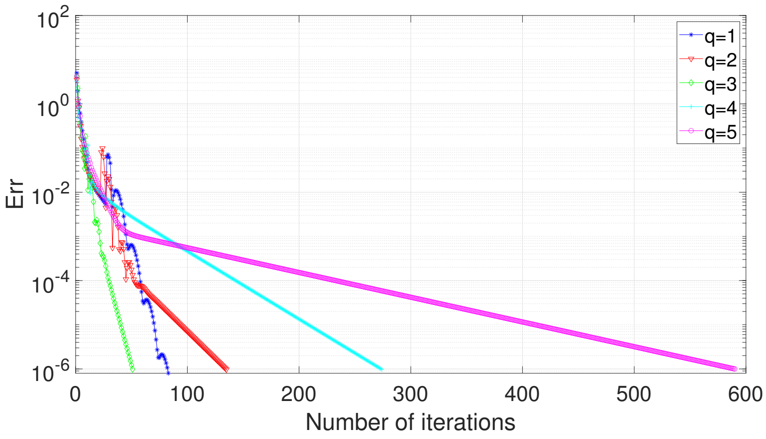

Figure 3: Comparison of parameters for the proposed algorithm; Case 3.

vector x^* is constructed such that a vector Dx^* has few nonzero elements. A matrix F is randomly generated and a noisy vector b is generated by adding randomly generated (noise) vector to Fx^*.

Case 1: $N = 50,\ M = 30,$ *and* $p = 5$
Case 2: $N = 100,\ M = 150,$ *and* $p = 10$
Case 3: $N = 300,\ M = 300,$ *and* $p = 20$
Case 4: $N = 400,\ M = 399,$ *and* $p = 40$

In order to compare the performance of the algorithms, we choose $\lambda = 0.4, \gamma = 0.8, \delta = -0.01, \theta = 0.01, \alpha_n = \frac{1}{5n+1}$ for our proposed algorithm, $\lambda = 0.1, \gamma = 0.4$ for Kim Algorithm and α_n for BM algorithm. The initial points are generated randomly and we use $E_n = \|x_{n+1} - x_n\| < 10^{-6}$ as stopping criterion. The numerical results are shown in Table 2 and Figure 4.

Methods	Case 1		Case 2		Case 3		Case 4	
	Iter	CPU	Iter	CPU	Iter	CPU	Iter	CPU
Pro. Alg.	620	0.0349	202	0.1789	1945	12.0832	2189	9.2076
Kim Alg.	720	0.0485	281	0.2810	2673	15.4102	2710	16.1221
BM Alg.	1179	0.9912	325	0.2910	3203	18.4379	3650	26.1219

Table 2: Numerical results of comparison for Example 4.1.

Remark 4.2.

Noting from Figure 4, Table 1 and Table 2, it is clear that our proposed algorithm is easy to implement for various choices of N, M and p.

Moreover, the proposed algorithm outperforms other algorithms considered in the experiment in each case.

Example 4.3. *Let $H = L^2([0,1])$. Let $A := \partial\|.\|$ in (1). Note that $A^{-1}(\mathbf{0})$ since $0 \in A^{-1}(\mathbf{0})$. Furthermore, the resolvent J_λ^A is given by the Moreau decomposition*

$$\begin{aligned} J_\lambda^A(x) &= (I + \lambda\partial\|.\|)^{-1}(x) \\ &= Prox_{\lambda\|.\|}(x) = x - \lambda P_{B_{\|.\|_*}}\Big(\frac{x}{\lambda}\Big), \end{aligned}$$

where $Prox_{\lambda\|.\|}(x) := argmin_y\{\|y\| + \frac{1}{2\lambda}\|y-x\|^2\}$, $P_{B_{\|.\|_*}}$ is the projection operator and $B_{\|.\|_*}$ is the norm unit ball(of the dual norm). Note that in this case, $L^2([0,1])$ is self dual. Moreover, the projection $P_{B_{\|.\|_*}}$ (see [13, Chapter 4]) is given by:

$$P_{B_{\|.\|_*}}(x) = \begin{cases} \frac{x}{\|x\|}, & \|x\| > 1 \\ x, & \|x\| \le 1. \end{cases}$$

Therefore,

$$J_\lambda^A(x) = x - \lambda P_{B_{\|.\|}}\Big(\frac{x}{\lambda}\Big) = \begin{cases} x - \lambda\frac{x}{\|x\|}, & \|\frac{x}{\lambda}\| > 1 \\ 0, & \|\frac{x}{\lambda}\| \le 1. \end{cases}$$

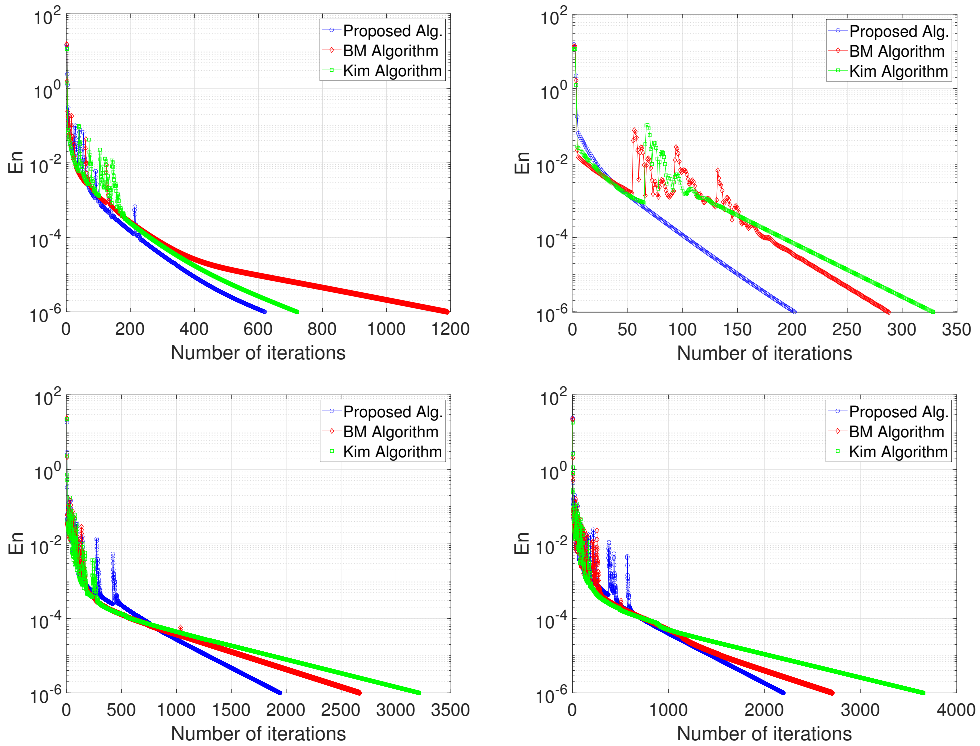

Figure 4: Numerical result for Example 4.1, Top Left: Case 1, Top Right: Case 2, Bottom Left: Case 3, Bottom Right: Case 4.

In order to compare the methods, we choose $\lambda = 0.4, \gamma = 0.8, \delta = -0.01, \theta = 0.01, \alpha_n = \frac{1}{5n+1}$ *for our proposed algorithm,* $\lambda = 0.8/L, \alpha = 0.01, \gamma = 1.2$, *for [19] method, denoted as Dey algorithm and* $\lambda = 0.04, \alpha_n = \frac{1}{10n+1}$ *for BM algorithm. We used* $\|x_{n+1} - x_n\| < 10^{-6}$ *as stopping criterion and test the algorithms using the following initial parameters:*
Case 1: $x_0 = t^3 + 2t - 1$, $x_1 = t^4 - 17$, *and* $y_0 = t^3/7$
Case 2: $x_0 = (t^2 - 1)/4$, $x_1 = \exp(2t)$, *and* $y_0 = 2t\exp(3t)$
Case 3: $x_0 = \exp(4t)/8$, $x_1 = t^3 - 2t$, *and* $y_0 = exp(2t)/5$
Case 4: $x_0 = 5(t^3 + 7t)/7$, $x_1 = exp(2t)$, *and* $y_0 = t^3 - 9t + 1$.

The numerical results are shown in Figures 5.

4.3 Application to Image Restoration

In this subsection, we demonstrate the application of the proposed algorithm to image restoration problem and compare its performance with the algorithms in [10, 19, 31]. The problem of image restoration can be modelled as the following minimization problem:

$$\min_x \frac{1}{2}\|\mathbb{E}x - b\|_2^2 + \mu\|x\|_1, \tag{53}$$

where x represents the (vectorized) observed image, $\mathbb{E} \in \mathbb{R}^{M\times N}$ is the blur operator, $b \in \mathbb{R}^M$ is an additive noise, μ is a positive constant, $\|x\|_2$ is the Euclidean norm of

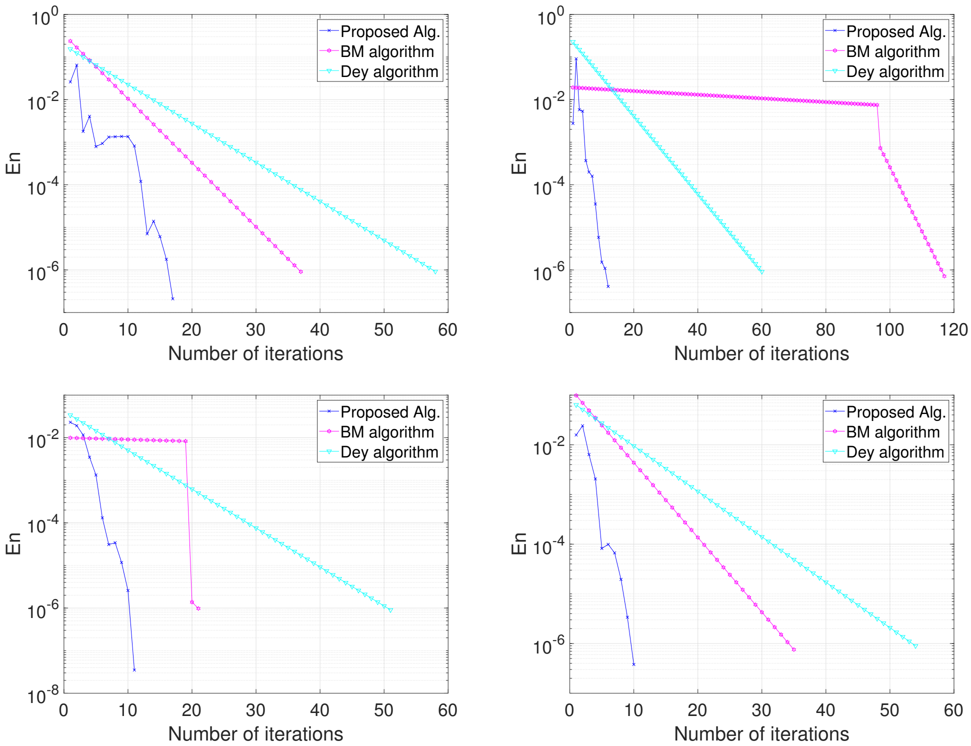

Figure 5: Numerical result for Example 4.1, Top Left: Case 1, Top Right: Case 2, Bottom Left: Case 3, Bottom Right: Case 4.

x and $\|x\|_1 = \sum_{i=1}^{N} |x_i|$ is the l_1-norm of x. Note that problem (53) is the same as problem (51) and thus Algorithm 2, which is a special case of Algorithm 1 can be adapted for this problem (53). In this experiment, we used the cameraman, tree, pout and kids images which are inbuilt in MATLAB consisting of M pixels wide and N pixel height given in Figure 6. Each image went through a Gaussian blur of size 9×9 and standard deviation 4 (applied by the MATLAB functions `imfilter` and `fspecial`), see Figure 7. In our implementation, we used the following parameters: for the proposed algorithm (Algorithm 2), we take $\alpha_n = \frac{1}{100n+5}$, $\theta = 0.04$, $\delta = -0.07$ and $\lambda = 0.02$; for Kim alg [31], $\lambda = 0.02$; for BM alg [10], we take $\alpha_n = \frac{1}{100n+5}, \beta_n = 0.02, e_n = \frac{1}{(5n+7)^2}$ and Dey alg [19], we take $\gamma = 1.46$, $\lambda_n = \frac{1}{2L}$, $\alpha = 0.08$. The initial values in each algorithm is taken as $x_0, y_0 = \mathbf{0} \in \mathbb{R}^D, x_{-1}, x_1 = \mathbf{1} \in \mathbb{R}^D$ where $D = M \times N$. We evaluate the quality of the restored image using the signal-to-noise ratio (SNR) defined by

$$SNR = 20 \log \frac{\|x\|}{\|x - x_r\|_2},$$

where x denotes the original image and x_r is the debluring image. Typically, higher SNR value indicate better restored image quality. We also compared the performance of the algorithms with respect to the relative error given by

$$\text{Rel. Err} = \frac{\|x_{n+1} - x_n\|}{\|x_n\|}.$$

The restored images are shown in Figures 8, 9, 10, 11. From the figures, we see that the proposed algorithm has the best CPU time for restoring the blurred images for the four test images. Furthermore, the quality of the restored images is better than that of Kim and Dey algorithms. Our proposed algorithm also has better Rel. Err value than Kim and BM algorithms.

(a) Cameramen. (b) Tree . (c) Pout. (d) Kids.

Figure 6: Original images of Cameraman, Tree, Pout and Kids.

(a) Cameramen. (b) Tree . (c) Pout. (d) Kids.

Figure 7: Blurred images of Cameraman, Tree, Pout and Kids.

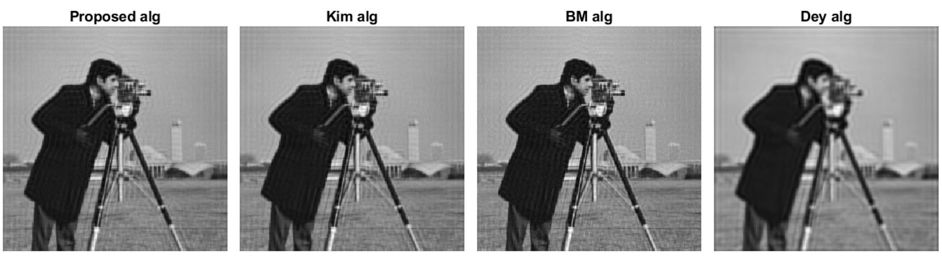

Figure 8: Reconstructed Cameraman image by Proposed algorithm, Kim alg, BM alg and Dey alg.

4.4 Application to LASSO Problem in Compressed Sensing

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 which can be modeled as the minimization problem in (53). In this case, where the number of unknowns is greater

Figure 9: Reconstructed Tree image by Proposed algorithm, Kim alg, BM alg and Dey alg.

Figure 10: Reconstructed Pout image by Proposed algorithm, Kim alg, BM alg and Dey alg.

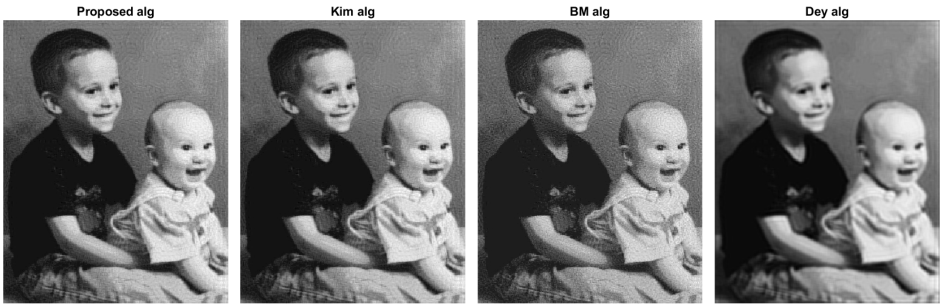

Figure 11: Reconstructed Kids image by Proposed algorithm, Kim alg, BM alg and Dey alg.

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 model (53) can be computed to recover x when x is sparse which is the case in most applications. The model given in (53) is most often referred to as LASSO. Standard general algorithms such as an Interior Point Method (IPM), [2], 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 a convex minimization which can be rewritten as an inclusion problem. In our implementation, we consider 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 $N = 2^{12}$

Image		Proposed alg	Kim alg	BM alg	Dey alg
Cameraman	Time (s)	48.4666	65.0012	52.2155	72.2947
	SNR	32.6153	32.5463	34.4465	24.6752
	Rel Err	7.5208E-7	1.0791E-6	7.9288E-6	2.0662E-7
Trees	Time (s)	61.2082	68.7534	62.6111	85.4045
	SNR	33.2801	33.2525	34.2088	25.8542
	Rel Err	7.3595E-7	1.03118E-6	7.0801E-6	1.7672E-7
Pout	Time (s)	45.2742	54.2742	48.0209	77.5505
	SNR	37.8125	37.3652	38.8312	26.0029
	Rel Err	3.5218E-7	5.2772E-7	3.6858E-6	1.0576E-7
Kids	Time (s)	88.3103	90.0161	112.2526	116.3457
	SNR	35.8383	35.6683	36.1332	25.9380
	Rel Err	8.5024E-7	1.1119E-6	7.9963E-6	1.2218E-7

Table 3: Time, SNR and Relative Error values for each images.

and $M = 2^6$ and the original signal contains 160 randomly nonzero elements for the experiment. The vector b is randomly generated by the normal distribution. 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

$$MSE = \frac{1}{N}\|x_n - x\|,$$

where x_n is an estimated signal of x and the signal-to-noise ratio given by

$$SNR = 20 \log \frac{\|x\|}{\|x - x_n\|}.$$

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 use the following parameters for the algorithms: for Proposed Algorithm 1, we choose $\alpha_n = 1/(n+1), \theta = 0.06, \delta = -0.05$ and $\lambda = 0.01$, for Kim alg, we take $\lambda = 0.04$, for BM alg, we take $\alpha_n = \frac{1}{n+1}, \beta_n = 0.003, e_n = 0$ and for Dey alg, we take $\gamma = 1, \lambda_n = \frac{1}{2L}$, and $\alpha = 0.64$. The starting points of the algorithms are generated as the zero vector in each case by setting $x_0 = y_0 = x_{-1}, x_1 = \mathbf{0} \in \mathbb{R}^N$. The original and restored signal are shown in Figure 12. Also the time taken by each algorithm, the SNR values and the MSE values for each algorithm are recorded in Table 4. From the numerical results in Figure 12 and Table 4, it is clear that the proposed method performs better than the other methods used in the experiments. The values of the SNR of the proposed Algorithm are higher than that of others while the MSE values are the lowest signifying a better quality of restored signal than other methods. Furthermore, the time taken by the proposed method is lower than the time taken by any other method. This highlights the importance of the new algorithm in denoising problems.

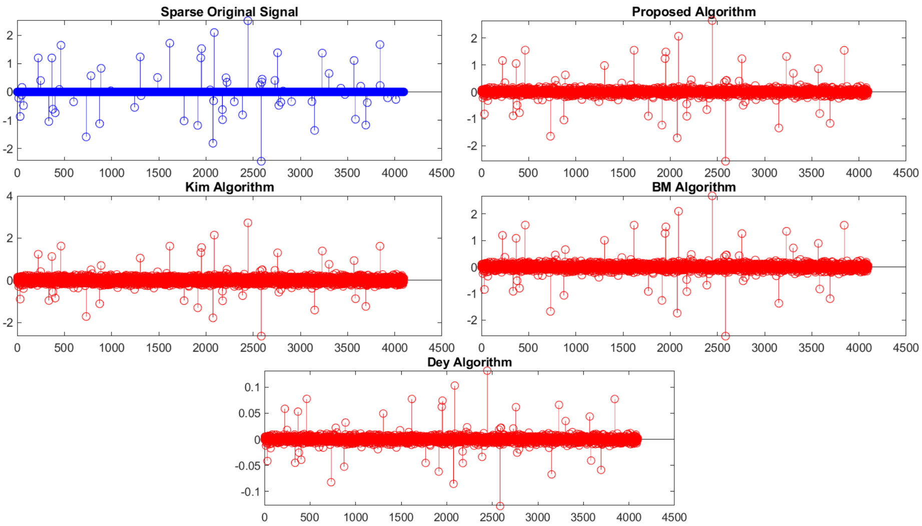

Figure 12: Algorithm performance for recovery sparse signals.

	Time (sec)	SNR values	MSE values
Prop. Alg.	0.0117	7.49 dB	7.7146E-4
BM Alg.	0.0575	5.53 dB	9.7408E-4
Kim Alg.	0.0756	2.19 dB	1.5821E-3
Dey Alg.	0.1704	0.42 dB	1.4378E-3

Table 4: Numerical performance showing the Time of execution, SNR values and MSE values for each algorithm.

Remark 4.4. A major novelty of our paper is the introduction of the new step-size rule given in Assumption 3.2 and as can be seen from the numerical examples given in Section 4, our proposed Algorithm 1 with Assumption 3.2 is more useful and has more practical advantages in image restoration and compressed sensing than related methods in [7, 19, 31].

5 Conclusion

We have introduced in this paper, a Halpern-type proximal point algorithm with two-step inertial extrapolation to solve the monotone inclusion problem in Hilbert spaces. We proved that the sequence of iterates generated by our proposed algorithm converges strongly to a zero of the monotone inclusion problem under some conditions on the iterative parameters without on-line rule assumption (12). Our main novelty is the new step-size rule in Assumption 3.2 which is useful and has practical advantages from the image restoration and compressed sensing problems over existing ones with on-line rule assumption (12). As part of our future project, we consider our proposed algorithm with errors and also give its rate of metastability.

Disclosure statement

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.

Availability of supporting data

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

Competing interests

The authors declare no competing interests.

Funding

Not Applicable.

References

[1] F. Aluffi-Pentini, V. Parisi and F. Zirilli; Algorithm 617. DAFNE: a differential-equations algorithm for nonlinear equations, *ACM Trans. Math. Software* **10** (1984), 317-324.

[2] F. Alvarez; On the minimizing property of a second order dissipative system in Hilbert spaces, *SIAM J. Control Optim.* **38** (2000), 1102–1119.

[3] F. Alvarez and H. Attouch; An inertial proximal method for maximal monotone operators via discretization of a nonlinear oscillator with damping, *Set-Valued Anal.* **9** (2001), 3–11.

[4] F. Alvarez; Weak convergence of a relaxed and inertial hybrid projection-proximal point algorithm for maximal monotone operators in Hilbert space, *SIAM J. Optim.* **14** (2004), 773–782.

[5] A. S. Antipin; Minimization of convex functions on convex sets by means of differential equations, *Differentsial'nye Uravneniya* **30** (1994), 1475-1486, 1652; translation in Differential Equations 30 (1994), 1365–1375.

[6] J. Bai, H. Zhang and J. Li; A parameterized proximal point algorithm for separable convex optimization, *Optim. Lett.* **12** (2018), 1589-1608.

[7] O. A. Boikanyo and G. Morosanu; A proximal point algorithm converging strongly for general errors, *Optim. Lett.* **4** (2010),635–641.

[8] O. A. Boikanyo and G. Morosanu; Modified Rockafellar's algorithms, *Math. Sci. Res. J.* **13** (2009), 101–122.

[9] O. A. Boikanyo and G. Morosanu; Inexact Halpern-type proximal point algorithm, *J. Global Optim.* **51** (2011), 11–26.

[10] O. A. Boikanyo and G. Morosanu; Four parameter proximal point algorithms, *Nonlinear Anal.* **74** (2011), 544–555.

[11] R. E. Bruck Jr.; Asymptotic convergence of nonlinear contraction semigroups in Hilbert space, *J. Functional Analysis* **18** (1975), 15–26.

[12] R. I. Boţ and E. R. Csetnek; An inertial alternating direction method of multipliers, *Minimax Theory Appl.* **1** (2016), 29-49.

[13] A. Cegielski; Iterative Methods for Fixed Point Problems in Hilbert Spaces, Lecture Notes in Mathematics 2057, Springer, Berlin, 2012.

[14] C. Chen, S. Ma and J. Yang; A general inertial proximal point algorithm for mixed variational inequality problem, *SIAM J. Optim.* **25** (2015), 2120–2142.

[15] W. Cholamjiak, P. Cholamjiak and S. Suantai; An inertial forward-backward splitting method for solving inclusion problems in Hilbert spaces, *J. Fixed Point Theory Appl.* **20**(42) (2018).

[16] W. Cholamjiak and H. Dutta; Viscosity modification with parallel inertial two steps forward-backward splitting methods for inclusion problems applied to signal recovery, *Chaos Solitons Fractals* **157** (2022), Paper No. 111858, 14 pp.

[17] P. L. Combettes and L. E. Glaudin; Quasi-nonexpansive iterations on the affine hull of orbits: from Mann's mean value algorithm to inertial methods, *SIAM J. Optim.* **27** (2017), 2356-2380.

[18] E. Corman and X. M. Yuan; A generalized proximal point algorithm and its convergence rate, *SIAM J. Optim.* **24** (2014), 1614-1638.

[19] S. Dey; A hybrid inertial and contraction proximal point algorithm for monotone variational inclusions, *Numer. Algor.* **93** (2023), 1-25.

[20] Q.L. Dong, J. Z. Huang, X. H. Li, Y. J. Cho and Th. M. Rassias;; MiKM: multi-step inertial Krasnosel'skii–Mann algorithm and its applications, *J. Global Optim.* **73** (2019), 801–824.

[21] J. Douglas and 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.

[22] J. Eckstein and D. P. Bertsekas; On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators, *Math. Program.* **55** (1992), 293-318.

[23] D. Gabay and B. Mercier; A dual algorithm for the solution of nonlinear variational problems via finite element approximation, *Comput. Math. Appl.* **2** (1976), 17-40.

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

[25] O. Güler; New proximal point algorithms for convex minimization, *SIAM J. Optim.* **2** (1992), 649-664.

[26] O. Güler; On the convergence of the proximal point algorithm for convex minimization, *SIAM J. Control Optim.* **29** (1991), 403–419.

[27] M. R. Hestenes; Multiplier and gradient methods, *J. Optim. Theory Appl.* **4** (1969), 303-320.

[28] Z. He, D. Zhang and F. Gu; Viscosity approximation method for m-accretive mapping and variational inequality in Banach space, *An. Stiint. Univ. Ovidius Constanta Ser. Mat.* **17** (2009), 91–104.

[29] S. Kamimura and W. Takahashi; Approximating solutions of maximal monotone operators in Hilbert spaces, *J. Approx. Theory* **106** (2000), 226–240.

[30] H. Khatibzadeh and S. Ranjbar; On the Strong Convergence of Halpern Type Proximal Point Algorithm, *J. Optim. Theory Appl.* **158** (2013), 385–396.

[31] D. Kim; Accelerated proximal point method for maximally monotone operators, *Math. Program.* **190** (2021), 57–87.

[32] U. Kohlenbach; On the proximal point algorithm and its Halpern-type variant for generalized monotone operators in Hilbert space, *Optim. Lett.* **16** (2022), 611-621.

[33] L. Leuştean and P. Pinto; Quantitative results on a Halpern-type proximal point algorithm, *Comput. Optim. Appl.* **79** (2021), 101–125.

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

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

[36] P. E. Maingé and N. Merabet; A new inertial-type hybrid projection-proximal algorithm for monotone inclusions, *Appl. Math. Comput.* **215** (2010), 3149-3162.

[37] P. E. Maingé; Approximation methods for common fixed points of nonexpansive mappings in Hilbert spaces, *J. Math. Anal. Appl.* **325** (2007), 469-479.

[38] G. Marino and H. K. Xu; Convergence of generalized proximal point algorithm, *Commun. Pure Appl. Anal.* **3** (2004), 791–808.

[39] B. Martinet; Régularisation d'inéquations variationnelles par approximations successives, (French) *Rev. Française Informat. Recherche Opérationnelle* **4** (1970), Sér. R-3, 154–158.

[40] M. Tao and X. Yuan; On the optimal linear convergence rate of a generalized proximal point algorithm, *J. Sci. Comput.* **74** (2018), 826-850.

[41] J. J. Moreau; Proximité et dualité dans un espace Hilbertien, (French) *Bull. Soc. Math. France* **93** (1965), 273-299.

[42] A. Moudafi and E. Elisabeth; An approximate inertial proximal method using the enlargement of a maximal monotone operator, *Int. J. Pure Appl. Math.* **5** (2003), 283-299.

[43] O. S. Iyiola and Y. Shehu; Convergence results of two-step inertial proximal point algorithm, *Appl. Numer. Math.* **182** (2022), 57-75.

[44] P. Pinto; A rate of metastability for the Halpern type proximal point algorithm, *Numer. Funct. Anal. Optim.* **42** (2021), 320-343.

[45] B. T. Polyak; Introduction to optimization, New York, Optimization Software, Publications Division, 1987.

[46] C. Poon, J. Liang; Trajectory of Alternating Direction Method of Multipliers and Adaptive Acceleration, *arXiv:1906.10114* (2019).

[47] C. Poon, J. Liang; Geometry of First-Order Methods and Adaptive Acceleration, *arXiv:2003.03910.* (2020).

[48] M. J. D. Powell; A method for nonlinear constraints in minimization problems, In: Fletcher, R., Ed., Optimization, Academic Press, New York, NY, (1969) 283-298.

[49] R. T. Rockafellar; Monotone operators and the proximal point algorithm, *SIAM J. Control Optim.* **14** (1976), 877–898.

[50] R. T. Rockafellar; Monotone operators and the proximal point algorithm, *SIAM J. Control. Optim.* **14** (1976) 877-898.

[51] B. Djafari Rouhani and V. Mohebbi; Strong Convergence of an Inexact Proximal Point Algorithm in a Banach Space, *J. Optim. Theory Appl.* **186** (2020) 134-147.

[52] S. Saejung and P. Yotkaew; Approximation of zeros of inverse strongly monotone operators in Banach spaces, *Nonlinear Anal.* **75** (2012), 742-750.

[53] B. Tan and S. Y. Cho; Strong convergence of inertial forward–backward methods for solving monotone inclusions, *Appl. Anal.* **101** (2022), 5386-5414.

[54] D.V. Thong, N. T. Vinh and Y. J. Cho; New strong convergence theorem of the inertial projection and contraction method for variational inequality problems, *Numer. Algor.* **84** (2020), 285-305.

[55] F. Wang and H. Cui; On the contraction-proximal point algorithms with multiparameters, *J. Global Optim.* **54** (2012), 485–491.

[56] H. K. Xu; Iterative algorithms for nonlinear operators, *J. Lond. Math. Soc.* **66** (2002), 240–256.

[57] H. K. Xu; A regularization method for the proximal point algorithm, *J. Global Optim.* **36** (2006), 115–125.

[58] Y. Yao and M. A. Noor; On convergence criteria of generalized proximal point algorithms, *J. Comput. Appl. Math.* **217** (2008), 46–55.

[59] C. Zhang, Q.L. Dong and J. Chen; Multi-step inertial proximal contraction algorithms for monotone variational inclusion problems, *Carpathian J. Math.* **36** (2020), 159-177.