A New Forward–Backward Algorithm with Line Search and Inertial Techniques for Convex Minimization Problems with Applications

Dawan Chumpungam 1, Panitarn Sarnmeta 1 and Suthep Suantai 1,2,*

1 Data Science Research Center, Department of Mathematics, Faculty of Science, Chiang Mai University, Chiang Mai 50200, Thailand; dawan_c@cmu.ac.th (D.C.); panitarn_s@cmu.ac.th (P.S.)
2 Research Center in Mathematics and Applied Mathematics, Department of Mathematics, Faculty of Science, Chiang Mai University, Chiang Mai 50200, Thailand
* Correspondence: suthep.s@cmu.ac.th

Abstract: For the past few decades, various algorithms have been proposed to solve convex minimization problems in the form of the sum of two lower semicontinuous and convex functions. The convergence of these algorithms was guaranteed under the L-Lipschitz condition on the gradient of the objective function. In recent years, an inertial technique has been widely used to accelerate the convergence behavior of an algorithm. In this work, we introduce a new forward–backward splitting algorithm using a new line search and inertial technique to solve convex minimization problems in the form of the sum of two lower semicontinuous and convex functions. A weak convergence of our proposed method is established without assuming the L-Lipschitz continuity of the gradient of the objective function. Moreover, a complexity theorem is also given. As applications, we employed our algorithm to solve data classification and image restoration by conducting some experiments on these problems. The performance of our algorithm was evaluated using various evaluation tools. Furthermore, we compared its performance with other algorithms. Based on the experiments, we found that the proposed algorithm performed better than other algorithms mentioned in the literature.

Keywords: convex minimization problems; machine learning; forward–backward algorithm; line search; accelerated algorithm; data classification; image restoration

1. Introduction

The convex minimization problem has been studied intensively for the past few decades due to its wide range of applications in various real-world problems. Some major problems in physics, economics, data science, engineering, and medical science can be viewed as convex minimization problems, for instance a reaction–diffusion equation, which is a mathematical model describing physical phenomena, such as chemical reactions, heat flow models, and population dynamics. The problem of finding an unknown reaction term of such an equation can be formulated as a coefficient inverse problem (CIP) for a partial differential equation (PDE). Numerical methods for solving CIPs for PDEs, as well as their applications to various subjects have been widely studied and analyzed; for more comprehensive information on this topic, see [1–5]. To obtain a globally convergent method for solving CIPs for PDEs, many authors have employed the convexification technique, which reformulates CIPs as convex minimization problems; for a more in-depth development and discussion, we refer to [6–8]. Therefore, a numerical method for convex minimization problems can be applied. More examples of convex minimization problems are signal and image processing, compressed sensing, and machine learning tasks such as regression and classification; see [9–16] and the references therein for more information.
The problem is formulated, in the form of the summation of two convex functions, as follows:

$$ \min_{x \in H} \{ f(x) + g(x) \}, \quad (1) $$

where \( f, g : H \to \mathbb{R} \cup \{+\infty\} \) are proper, lower semicontinuous convex functions and \( H \) is a Hilbert space.

If \( f \) is differentiable, then \( x \) solves (1) if and only if:

$$ x = \text{prox}_{\alpha g}(I - \alpha \nabla f)(x), \quad (2) $$

where \( \alpha > 0 \), \( \text{prox}_{\alpha g}(x) = \{ y \in H \mid \| y - x \| = \inf_{z \in H} \| z - x \| + \alpha \partial g(z) \} \), \( I \) is an identity mapping and \( \partial g \) is a subdifferential of \( g \). In other words, \( x \) is a fixed point of \( \text{prox}_{\alpha g}(I - \alpha \nabla f) \). Under some conditions, the Picard iteration converges to the solution. As a result, the forward–backward algorithm \([17]\), which is defined as follows:

$$ x_{n+1} = \text{prox}_{\alpha_n g}(I - \alpha_n \nabla f)(x_n), \quad \text{for all } n \in \mathbb{N}, \quad (3) $$

where \( \alpha_n \) is a suitable step size, converges to a solution of (1). Due to its simplicity, this method has been intensively studied and improved by many researchers; see \([10,16,18]\) and the references therein for more information. One well-known method, which notably improves the convergence rate of (3), is the fast iterative shrinkage-thresholding algorithm (FISTA) introduced by Beck and Teboulle \([19]\). To the best of our knowledge, most of these works assumed that \( \nabla f \) is Lipschitz continuous. However, in general, such an assumption might be too strong, and finding a Lipschitz constant of \( \nabla f \) is sometimes difficult. To relax this strong assumption, Cruz and Nghia \([20]\) proposed a line search technique and replaced the Lipschitz continuity assumption of \( \nabla f \) with weaker assumptions, as seen in the following conditions:

**Assumption 1.** \( f, g \) are proper lower semicontinuous convex functions with \( \text{dom} \ g \subseteq \text{dom} \ f \);

**Assumption 2.** \( f \) is differentiable on an open set containing \( \text{dom} \ g \), and \( \nabla f \) is uniformly continuous on any bounded subset of \( \text{dom} \ g \) and maps any bounded subset of \( \text{dom} \ g \) to a bounded set in \( H \).

Note that if \( \nabla f \) is \( L \)-Lipschitz continuous, then A2 holds. They also established an algorithm using a line search technique and obtained a weak convergence result under Assumptions 1 and 2. In the same work, they also proposed an accelerated algorithm based on FISTA and a line search technique. They showed that this accelerated algorithm performed better than the other introduced algorithm.

Recently, inspired by \([20]\), Kankam et al. \([9]\) proposed a new line search technique and a new algorithm to solve (1). They conducted some experiments on signal processing and found that their method performed better than that of Cruz and Nghia \([20]\).

In recent years, many authors have utilized the inertial technique in order to accelerate their algorithms. This was first introduced by Polyak \([21]\) to solve smooth convex minimization problems. After that, many inertial-type algorithms were proposed and studied by many authors; see \([22–25]\) and the references therein.

The algorithms introduced in \([9,20]\) are easy to employ, since a Lipschitz constant of the gradient of an objective function \( f \) is not required to exist. Although the convergence of these algorithms is guaranteed, some improvements are still welcome, specifically utilizing an inertial technique in order to improve the performance. Hence, in this work, motivated by the works mentioned above, our main objective was to propose a new algorithm that utilizes both line search and inertial techniques to not only guarantee its convergence to a solution of (1) without assuming an \( L \)-Lipschitz continuity on \( \nabla f \), but also to improve its performance by mean of an inertial technique. We established a weak convergence theorem under Assumptions 1 and 2. Moreover, a complexity theorem of this new algorithm was
studied. Furthermore, we employed our proposed algorithm to solve an image restoration problem, as well as data classification problems. Then, we evaluated its performance and compared it with various other algorithms. The proposed method is also of great interest in solving nonlinear coefficient inverse problems for partial differential equations, along with other possible applications of convex minimization problems.

This work is organized as follows: In Section 2, we recall important definitions, lemmas that will be used in later sections, as well as various methods introduced in [9, 19, 20, 22]. In Section 3, we introduce a new algorithm using new line search and inertial techniques and establish its weak convergence to a solution of (1). Moreover, the complexity of this method is also analyzed. In Section 4, in order to compare the performance of the studied algorithms, some numerical experiments on data classification problems and image restoration problem are conducted and discussed. In the last section, the brief conclusion of this paper is presented.

2. Preliminaries

Throughout this work, we denote \( x_n \to x \) and \( x_n \rightharpoonup x \) as \( \{x_n\} \) converges strongly and weakly to \( x \), respectively. For \( h : H \to \mathbb{R} \cup \{+\infty\} \), we also denote \( \text{dom } h := \{x \in H : h(x) < +\infty\} \).

First, we recall some methods for solving (1) introduced by many authors mentioned in Section 1.

The fast iterative shrinkage-thresholding algorithm (FISTA) was introduced by Beck and Teboulle[19] as follows (Algorithm 1).

**Algorithm 1.** FISTA.

1: **Input** Given \( y_1 = x_0 \in \mathbb{R}^n \), and \( t_1 = 1 \), for \( n \in \mathbb{N} \),

\[
    x_n = \text{prox}_{\frac{1}{L} \nabla f}(y_n - \frac{1}{L} \nabla f(y_n)),
\]

\[
    t_{n+1} = \frac{1 + \sqrt{1 + 4t_n^2}}{2}, \quad \theta_n = \frac{t_n - 1}{t_{n+1}},
\]

\[
    y_{n+1} = x_n + \theta_n(x_n - x_{n-1}), \quad n \in \mathbb{N},
\]

where \( L \) is a Lipschitz constant of \( \nabla f \).

In 2016, Cruz and Nghia [20] introduced a line search step as follows (Algorithm 2).

**Algorithm 2.** Line Search 1\((x, \sigma, \theta, \delta)\).

1: **Input** Given \( x \in \text{dom } g, \sigma > 0, \theta \in (0, 1) \), and \( \delta \in (0, \frac{1}{2}) \).

2: Set \( \alpha = \sigma \).

3: **while** \( \alpha \|\nabla f(\text{prox}_{\alpha g}(x - \alpha \nabla f(x))) - \nabla f(x)\| > \delta \|\text{prox}_{\alpha g}(x - \alpha \nabla f(x)) - x\| \) **do**

4: **Set** \( \alpha = \theta \alpha \)

5: **end while**

6: **Output** \( \alpha \).
They asserted that Line Search 1 stops after finitely many steps and proposed the following algorithm (Algorithm 3).

**Algorithm 3.** Line Search 1 stops after finitely many steps.

1. **Input** Given $x_0 \in \text{dom } g$, $\sigma > 0$, $\theta \in (0, 1)$, and $\delta \in (0, \frac{1}{2})$ for $n \in \mathbb{N}$,
   \[ x_{n+1} = \text{prox}_{\alpha_n g}(I - \alpha_n \nabla f)(x_n), \]

where $\alpha_n := \text{Line Search 1}(x_n, \sigma, \theta, \delta)$.

They showed that a sequence generated by Algorithm 3 converges weakly to a solution of (1) under Assumptions 1 and 2.

Recently, Kankam et al. [9] proposed a new line search technique as follows (Algorithm 4).

**Algorithm 4.** Line Search 2($x, \sigma, \theta, \delta$).

1. **Input** Given $x \in \text{dom } g$, $\sigma > 0$, $\theta \in (0, 1)$, and $\delta > 0$. Set
   \[ L(x, \alpha) = \text{prox}_{\alpha g}(x - \alpha \nabla f(x)), \]
   \[ S(x, \alpha) = \text{prox}_{\alpha g}(L(x, \alpha) - \alpha \nabla f(L(x, \alpha))). \]

2. Set $\alpha = \sigma$.
3. **while** \[ \alpha \max\left\{\|\nabla f(S(x, \alpha)) - \nabla f(L(x, \alpha))\|, \|\nabla f(L(x, \alpha)) - \nabla f(x)\|\right\} > \delta(\|S(x, \alpha) - L(x, \alpha)\| + \|L(x, \alpha) - x\|) \]
   **do**
4. Set $\alpha = \theta \alpha$, $L(x, \alpha) = L(x, \theta \alpha)$, $S(x, \alpha) = S(x, \theta \alpha)$
5. **end while**
6. **Output** $\alpha$.

They proved that Line Search 2 stops at finitely many steps and introduced the following algorithm (Algorithm 5).

**Algorithm 5.** Line Search 2 stops at finitely many steps.

1. **Input** Given $x_0 \in \text{dom } g$, $\sigma > 0$, $\theta \in (0, 1)$, and $\delta \in (1, \frac{1}{2})$ for $n \in \mathbb{N}$,
   \[ y_n = \text{prox}_{\gamma_n g}(x_n - \gamma_n \nabla f(x_n)), \]
   \[ x_{n+1} = \text{prox}_{\gamma_n g}(y_n - \gamma_n \nabla f(y_n)), \]

where $\gamma_n := \text{Line Search 2}(x_n, \sigma, \theta, \delta)$.

They established a weak convergence theorem for Algorithm 5, under Assumptions 1 and 2. As we can see, Algorithms 3 and 5 do not utilize an inertial step.

Inspired by Algorithm 1 (FISTA), Cruz and Nghia [20] also proposed the following accelerated algorithm (Algorithm 6).
Algorithm 6. Accelerated algorithm.

1: Input Given $x_0, x_1 \in \text{dom } g, \alpha > 0, \theta \in (0, 1), t_0 = 1$, and $\delta \in (0, \frac{1}{2})$, for $n \in \mathbb{N}$,

$$
t_{n+1} = \frac{1 + \sqrt{1 + 4\mu_n^2}}{2},$$

$$y_n = x_n + \left(\frac{t_n - 1}{t_{n+1}}\right)(x_n - x_{n-1}),$$

$$\tilde{g}_n = P_{\text{dom } g}(y_n),$$

$$x_{n+1} = \text{prox}_{\alpha g}(\tilde{g}_n - \alpha_n \nabla f(\tilde{g}_n)),$$

where $\alpha_n := \text{Line Search } 1(\tilde{g}_n, \alpha_{n-1}, \theta, \delta)$, and $P$ is a metric projection.

They showed that Algorithm 6 has better complexity than Algorithm 3. However, the convergence to a solution of (1) is not guaranteed under the inertial parameter $\beta_n = \frac{t_n - 1}{t_{n+1}}$.

In 2019, Attouch and Cabot [22] analyzed the convergence rate of a method called the relaxed inertial algorithm (RIPA) for solving monotone inclusion problems, which is closely related to convex minimization problems. This method utilizes an inertial step $x_n + \beta_n(x_n - x_{n-1})$ to improve its performance. It was defined as follows (Algorithm 7).

Algorithm 7. RIPA.

1: Input Given $x_0, x_1 \in H, \beta_n \geq 0, \rho_n \in (0, 1], \mu_n > 0$, for $n \in \mathbb{N}$,

$$y_n = x_n + \beta_n(x_n - x_{n-1}),$$

$$x_{n+1} = (1 - \rho_n)y_n + \rho_n f_{\mu_n}(y_n),$$

where $A$ is a maximal monotone operator and $f_{\mu_n}(y_n) = (1 + \mu_n A)^{-1}$.

Under mild restrictions of the control parameters, they showed that Algorithm 7 (RIPA) gives a fast convergence rate.

Next, we recall some important tools that will be used in the later sections.

Definition 1. For $x \in H$, a subdifferential of $h$ at $x$ is defined as follows:

$$\partial h(x) := \{u \in H : \langle u, y - x \rangle + h(x) \leq h(y), y \in H\}.$$ 

The following can be found in [26].

Lemma 1 ([26]). A subdifferential $\partial h$ is maximal monotone. Moreover, a graph of $\partial h, \text{Gph}(\partial h) := \{(u, v) \in H \times H : v \in \partial h(u)\}$ is demiclosed, i.e., for any sequence $(u_n, v_n) \in \text{Gph}(\partial h)$ such that $(u_n)$ converges weakly to $u$ and $(v_n)$ converges strongly to $v$, we have $(u, v) \in \text{Gph}(\partial h)$.

The proximal operator, $\text{prox}_{g} : H \to \text{dom } g$ with $\text{prox}_{g}(x) = (1 + g)^{-1}(x)$, is single-valued with the full domain, and the following holds:

$$\frac{x - \text{prox}_{\alpha g}x}{\alpha} \in \partial g(\text{prox}_{\alpha g}x), \text{ for all } x \in H \text{ and } \alpha > 0. \quad (4)$$

The following lemmas are crucial for the main results.

Lemma 2 ([27]). Let $H$ be a real Hilbert space. Then, the following hold, for all $x, y \in H$ and $\alpha \in [0, 1]$:

(i) $\|x \pm y\|^2 = \|x\|^2 \pm 2 \langle x, y \rangle + \|y\|^2$;

(ii) $\|\alpha x + (1 - \alpha)y\|^2 = \alpha \|x\|^2 + (1 - \alpha)\|y\|^2 - \alpha(1 - \alpha)\|x - y\|^2$;
Mathematics 2021, 9, 1562

(iii) $\|x + y\|^2 \leq \|x\|^2 + 2(y, x + y)$.

Lemma 3 ([16]). Let $\{a_n\}$ and $\{\beta_n\}$ be sequences of non-negative real numbers such that:

$$a_{n+1} \leq (1 + \beta_n)a_n + \beta_na_{n-1}, \text{ for all } n \in \mathbb{N}.$$ 

Then, the following holds:

$$a_{n+1} \leq K \prod_{j=1}^{n}(1 + 2\beta_j), \text{ where } K = \max\{a_1, a_2\}.$$ 

Moreover, if $\sum_{n=1}^{\infty} \beta_n < +\infty$, then $\{a_n\}$ is bounded.

Lemma 4 ([27]). Let $\{a_n\}, \{b_n\}$ and $\{\delta_n\}$ be sequences of non-negative real numbers such that:

$$a_{n+1} \leq (1 + \delta_n)a_n + b_n, \text{ for all } n \in \mathbb{N}.$$ 

If $\sum_{n=1}^{\infty} \delta_n < +\infty$ and $\sum_{n=1}^{\infty} b_n < +\infty$, then $\lim_{n \to +\infty} a_n$ exists.

Lemma 5 ([28]). Let $H$ be a Hilbert space and $\{x_n\}$ a sequence in $H$ such that there exists a nonempty subset $\Omega$ of $H$ satisfying the following:

(i) for any $x^* \in \Omega$, $\lim_{n \to} \|x_n - x^*\|$ exists;

(ii) every weak-cluster point of $\{x_n\}$ belongs to $\Omega$.

Then, $\{x_n\}$ converges weakly to an element in $\Omega$.

3. Main Results

In this section, we assume the existence of a solution of (1) and denote $S_*$ the set of all such solutions. We modify Line Search 2 as follows (Algorithm 8).

Algorithm 8. Line Search 3($x, \sigma, \theta, \delta$).
1: Input Given $x \in \text{dom } g$, $\sigma > 0$, $\theta \in (0, 1)$, and $\delta > 0$. Set 

$L(x, \alpha) = \text{prox}_{\alpha g}(x - \alpha \nabla f(x))$, and 

$S(x, \alpha) = \text{prox}_{\alpha g}(L(x, \alpha) - \alpha \nabla f(L(x, \alpha)))$.

2: Set $\alpha = \sigma$.
3: while

$$\frac{\alpha}{2}(\|\nabla f(S(x, \alpha)) - \nabla f(L(x, \alpha))\| + \|\nabla f(L(x, \alpha)) - \nabla f(x)\|)$$

$$> \delta(\|S(x, \alpha) - L(x, \alpha)\| + \|L(x, \alpha) - x\|),$$

do

4: Set $\alpha = \theta \alpha$, $L(x, \alpha) = L(x, \theta \alpha)$, $S(x, \alpha) = S(x, \theta \alpha)$
5: end while
6: Output $\alpha$.

We know that Line Search 3 terminates at a lower step than Line Search 2, since:

$$\frac{\alpha}{2}(\|\nabla f(S(x, \alpha)) - \nabla f(L(x, \alpha))\| + \|\nabla f(L(x, \alpha)) - \nabla f(x)\|)$$

$$\leq \alpha \max\{\|\nabla f(S(x, \alpha)) - \nabla f(L(x, \alpha))\|, \|\nabla f(L(x, \alpha)) - \nabla f(x)\|\}.$$
Therefore, Line Search 3 stops after finitely many steps. We introduce an accelerated algorithm by utilizing Line Search 3 as follows (Algorithm 9).

**Algorithm 9.**

1. **Input** Given $x_0, x_1 \in \text{dom } g, \beta_n \geq 0, \sigma > 0, \theta \in (0, 1)$ and $\delta \in (0, \frac{1}{8}),$ for $n \in \mathbb{N},$

$$
\hat{x}_n = x_n + \beta_n (x_n - x_{n-1}),
$$

$$
y_n = P_{\text{dom } g} \hat{x}_n,
$$

$$
z_n = \text{prox}_{\gamma_n g} (y_n - \gamma_n \nabla f(y_n) ),
$$

$$
x_{n+1} = \text{prox}_{\gamma_n g} (z_n - \gamma_n \nabla f(z_n)),
$$

where $\gamma_n := \text{Line Search } 3(y_n, \sigma, \theta, \delta),$ and $P_{\text{dom } g}$ is a metric projection map onto $\text{dom } g.$

Next, we establish our first theorem.

**Theorem 1.** Let $H$ be a Hilbert space, $f : H \to \mathbb{R} \cup \{+\infty\}$ and $g : H \to \mathbb{R} \cup \{+\infty\}$ proper lower semicontinuous convex functions satisfying A1 and A2. Suppose that $\text{dom } g$ is closed and the following also hold, for all $n \in \mathbb{N}:

C1. $\gamma_n \geq \gamma > 0;$

C2. $\beta_n \geq 0$ and $\sum_{n=1}^{+\infty} \beta_n < +\infty.$

Then, a sequence $\{x_n\}$ generated by Algorithm 9 converges weakly to a point in $S_\ast.$

**Proof.** For any $x \in \text{dom } g$ and $n \in \mathbb{N},$ we claim that:

$$
||y_n - x||^2 - ||x_{n+1} - x||^2 \geq 2\gamma_n [(f + g)(z_n) + (f + g)(x_{n+1}) - 2(f + g)(x)] + (1 - 8\delta)(||z_n - y_n||^2 + ||x_{n+1} - z_n||^2).
$$ (5)

To prove our claim, we know, from (4) and the definition of $\partial g,$ that:

$$
\frac{y_n - z_n}{\gamma_n} - \nabla f(y_n) \in \partial g(z_n), \quad \text{and} \quad \frac{z_n - x_{n+1}}{\gamma_n} - \nabla f(z_n) \in \partial g(x_{n+1}).
$$

Then,

$$
g(x) - g(z_n) \geq \frac{y_n - z_n}{\gamma_n} - \nabla f(y_n), x - z_n, \quad \text{and} \quad \gamma_n.
$$

$$
g(x) - g(x_{n+1}) \geq \frac{z_n - x_{n+1}}{\gamma_n} - \nabla f(z_n), x - x_{n+1}, \quad \text{for all } n \in \mathbb{N}.
$$

Moreover,

$$
f(x) - f(y_n) \geq \langle \nabla f(y_n), x - y_n, \rangle,
$$

$$
f(x) - f(z_n) \geq \langle \nabla f(z_n), x - z_n, \rangle,
$$

$$
f(y_n) - f(z_n) \geq \langle \nabla f(z_n), y_n - z_n, \rangle, \quad \text{and} \quad \text{for all } n \in \mathbb{N},
$$

$$
f(z_n) - f(x_{n+1}) \geq \langle \nabla f(x_{n+1}), z_n - x_{n+1}, \rangle, \quad \text{for all } n \in \mathbb{N}.
$$

From the definition of $\gamma_n$ and the above inequalities, we have, for all $x \in \text{dom } g$ and $n \in \mathbb{N},$
As a result, we obtain:

\[
f(x) - f(y_n) + f(z_n) - f(z_n) + g(x) - g(z_n) + g(x) - g(x_{n+1}) \geq \frac{1}{\gamma_n} (y_n - z_n, x - z_n) + \langle \nabla f(y_n), z_n - y_n \rangle + \frac{1}{\gamma_n} (z_n - x_{n+1}, x - x_{n+1}) \\
+ \langle \nabla f(z_n), x_{n+1} - z_n \rangle \\
= \frac{1}{\gamma_n} (y_n - z_n, x - z_n) + \langle \nabla f(y_n) - \nabla f(z_n), z_n - y_n \rangle + \langle \nabla f(z_n), z_n - y_n \rangle \\
+ \frac{1}{\gamma_n} (z_n - x_{n+1}, x - x_{n+1}) + \langle \nabla f(z_n) - \nabla f(x_{n+1}), x_{n+1} - z_n \rangle \\
+ \langle \nabla f(x_{n+1}), x_{n+1} - z_n \rangle \\
\geq \frac{1}{\gamma_n} (y_n - z_n, x - z_n) + \frac{1}{\gamma_n} (z_n - x_{n+1}, x - x_{n+1}) + \langle \nabla f(z_n), z_n - y_n \rangle \\
+ \langle \nabla f(z_n) - \nabla f(y_n), \|z_n - y_n\| + \|x_{n+1} - z_n\| \rangle \\
- \langle \nabla f(x_{n+1}) - \nabla f(z_n), \|z_n - y_n\| + \|x_{n+1} - z_n\| \rangle + \langle \nabla f(x_{n+1}), x_{n+1} - z_n \rangle \\
= \frac{1}{\gamma_n} (y_n - z_n, x - z_n) + \frac{1}{\gamma_n} (z_n - x_{n+1}, x - x_{n+1}) + \langle \nabla f(z_n), z_n - y_n \rangle \\
+ \langle \nabla f(z_n), x_{n+1} - z_n \rangle - \frac{2\delta}{\gamma_n} (\|z_n - y_n\| + \|x_{n+1} - z_n\|)^2 \\
\geq \frac{1}{\gamma_n} (y_n - z_n, x - z_n) + \frac{1}{\gamma_n} (z_n - x_{n+1}, x - x_{n+1}) + \langle \nabla f(z_n), z_n - y_n \rangle \\
\geq \frac{1}{\gamma_n} (y_n - z_n, z_n - x) + \frac{1}{\gamma_n} (z_n - x_{n+1}, x_{n+1} - x) \\
= (f + g)(z_n) + (f + g)(x_{n+1}) - 2(f + g)(x) - \frac{4\delta}{\gamma_n} \|z_n - y_n\|^2 - \frac{4\delta}{\gamma_n} \|x_{n+1} - z_n\|^2.
\]

Hence,

\[
\frac{1}{\gamma_n} (y_n - z_n, z_n - x) + \frac{1}{\gamma_n} (z_n - x_{n+1}, x_{n+1} - x) \\
\geq (f + g)(z_n) + (f + g)(x_{n+1}) - 2(f + g)(x) - \frac{4\delta}{\gamma_n} \|z_n - y_n\|^2 - \frac{4\delta}{\gamma_n} \|x_{n+1} - z_n\|^2.
\]

Moreover, the following also hold, for all \( n \in \mathbb{N} \),

\[
\langle y_n - z_n, z_n - x \rangle = \frac{1}{2} (\|y_n - x\|^2 - \|y_n - z_n\|^2 - \|z_n - x\|^2), \text{ and,}
\]

\[
\langle z_n - x_{n+1}, x_{n+1} - x \rangle = \frac{1}{2} (\|z_n - x\|^2 - \|z_n - x_{n+1}\|^2 - \|x_{n+1} - x\|^2).
\]

As a result, we obtain:

\[
\frac{1}{2\gamma_n} (\|y_n - x\|^2 - \|y_n - z_n\|^2) - \frac{1}{2\gamma_n} (\|z_n - x_{n+1}\|^2 + \|x_{n+1} - x\|^2) \\
\geq (f + g)(z_n) + (f + g)(x_{n+1}) - 2(f + g)(x) - \frac{4\delta}{\gamma_n} \|z_n - y_n\|^2 - \frac{4\delta}{\gamma_n} \|x_{n+1} - z_n\|^2,
\]
for all \( x \in \text{dom } g \) and \( n \in \mathbb{N} \). Therefore,
\[
\|y_n - x\|^2 - \|x_{n+1} - x\|^2 \geq 2\gamma_n(\|f + g\|(z_n) + (f + g)(x_{n+1}) - 2(f + g)(x)) + (1 - 8\delta)(\|z_n - y_n\|^2 + \|x_{n+1} - z_n\|^2),
\]
for all \( x \in \text{dom } g \) and \( n \in \mathbb{N} \). Furthermore, putting \( x = x^* \in S_* \), we obtain:
\[
\|y_n - x^*\|^2 - \|x_{n+1} - x^*\|^2 \geq (1 - 8\delta)(\|z_n - y_n\|^2 + \|x_{n+1} - z_n\|^2).
\]
(6)

Next, we show that \( \lim_{n \to +\infty} \|x_n - x^*\| \) exists. From (6), we have:
\[
\|x_{n+1} - x^*\| \leq \|y_n - x^*\|,
\]
\[
= \|P_{\text{dom } g} \hat{x}_n - P_{\text{dom } g} x^*\|,
\]
\[
\leq \|\hat{x}_n - x^*\|,
\]
\[
\leq \|x_n - x^*\| + \beta_n\|x_n - x_{n-1}\|,
\]
\[
\leq (1 + \beta_n)\|x_n - x^*\| + \beta_n\|x_{n-1} - x^*\|, \quad \text{for all } n \in \mathbb{N}.
\]
(7)

This implies by Lemma 3 and C2 that \( \{x_n\} \) is bounded.

Consequently, \( \sum_{n=1}^{+\infty} \beta_n\|x_n - x_{n-1}\| < +\infty \), and:
\[
\|\hat{x}_n - x_n\| = \beta_n\|x_n - x_{n-1}\| \to 0, \quad \text{as } n \to +\infty.
\]

By (7) and Lemma 4, we obtain that \( \lim_{n \to +\infty} \|x_n - x^*\| \) exists. By the definitions of \( z_{n-1} \) and \( x_n \), we see \( x_n \in \text{dom } g \), for all \( n \in \mathbb{N} \). As a result, we have:
\[
\|\hat{x}_n - y_n\| \leq \|\hat{x}_n - x_n\|, \quad \text{for all } n \in \mathbb{N},
\]
so, \( \lim_{n \to +\infty} \|\hat{x}_n - y_n\| = 0 \).

Thus, we obtain from (7) that \( \lim_{n \to +\infty} \|x_n - x^*\| = \lim_{n \to +\infty} \|y_n - x^*\| \). Moreover, it follows from (6) that \( \lim_{n \to +\infty} \|z_n - y_n\| = 0 \), and hence, \( \lim_{n \to +\infty} \|z_n - x_n\| = 0 \).

Now, we prove that every weak-cluster point of \( \{x_n\} \) belongs to \( S_* \). In order to accomplish this, we first let \( w \) be a weak-cluster point of \( \{x_n\} \). Therefore, there exists a subsequence \( \{x_{n_k}\} \) of \( \{x_n\} \) such that \( x_{n_k} \rightharpoonup w \), and hence, \( z_{n_k} \rightharpoonup w \). Next, we prove that \( w \) belongs to \( S_* \). Since \( \forall f \) is uniformly continuous, we have \( \lim_{k \to +\infty} \|\nabla f z_{n_k} - \nabla f y_{n_k}\| = 0 \).

From (4), we obtain:
\[
\frac{y_{n_k} - \gamma_{n_k}\nabla f y_{n_k} - z_{n_k}}{\gamma_{n_k}} \in \partial g(z_{n_k}), \quad \text{for all } k \in \mathbb{N}.
\]

Hence,
\[
\frac{y_{n_k} - z_{n_k}}{\gamma_{n_k}} - \nabla f y_{n_k} + \nabla f z_{n_k} \in \partial g(z_{n_k}) + \nabla f z_{n_k} = \partial (f + g)(z_{n_k}), \quad \text{for all } k \in \mathbb{N}.
\]

By letting \( k \to +\infty \) in the above inequality, the demiclosedness of \( G\phi(\partial (f + g)) \) implies that \( 0 \in \partial (f + g)(w) \), and hence, \( w \in S_* \). Therefore, every weak-cluster point of \( \{x_n\} \) belongs to \( S_* \).

We derive from Lemma 5 that \( \{x_n\} \) converges weakly to \( w^* \) in \( S_* \). Therefore, \( \{x_n\} \) converges weakly to a solution of (1), and the proof is complete. \qed
In the next theorem, we provide the complexity theorem of Algorithm 9. First, we introduce the control sequence \( \{t_n\} \) defined in [22] by:

\[
t_n = 1 + \sum_{k=n}^{+\infty} (\prod_{i=n}^{k} \beta_i), \quad \text{for all } n \in \mathbb{N}.
\]

(8)

This sequence is well defined if the following assumption holds:

\[
\sum_{k=n}^{+\infty} (\prod_{i=n}^{k} \beta_i) < +\infty, \quad \text{for all } n \in \mathbb{N}.
\]

Hence, from (8), we can see that:

\[
\beta_n t_{n+1} + 1 = t_n - 1, \quad \text{for all } n \in \mathbb{N}.
\]

(9)

Next, we establish the following theorem.

**Theorem 2.** Given \( x_0 = x_1 \in \text{dom } g \) and letting \( \{x_n\} \) be a sequence generated by Algorithm 9, assume that all assumptions in Theorem 1 are satisfied. Furthermore, suppose that the following conditions hold, for all \( n \in \mathbb{N} \):

1. \( \sum_{k=n}^{+\infty} (\prod_{i=n}^{k} \beta_i) < +\infty, \) and \( 2t_{n+1}^2 - 2t_n^2 \leq t_n^2 \)

D1.

2. \( \gamma_n \geq \gamma_{n+1} \).

D2.

Then,

\[
(f + g)(x_{n+1}) - \min_{x \in H}(f + g)(x) \leq \frac{d(x_1, S_\ast)^2 + 2\gamma_1 t_1^2 [(f + g)(x_1) - \min_{x \in H}(f + g)(x)]}{2\gamma_n t_{n+1}^2},
\]

for all \( n \in \mathbb{N} \). In other words,

\[
(f + g)(x_{n+1}) - \min_{x \in H}(f + g)(x) = O\left(\frac{1}{t_{n+1}^2}\right), \quad \text{for all } n \in \mathbb{N}.
\]

**Proof.** For any \( x \in \text{dom } g \), we know that:

\[
\|y_n - x\|^2 - \|x_{n+1} - x\|^2 \geq 2\gamma_n [(f + g)(z_n) + (f + g)(x_{n+1}) - 2(f + g)(x)]
\]

for all \( n \in \mathbb{N} \). Since \( x \in \text{dom } g \), we obtain \( \|\hat{x}_n - x\| \geq \|y_n - x\| \). Thus, we conclude that:

\[
\|\hat{x}_n - x\|^2 - \|x_{n+1} - x\|^2 \geq 2\gamma_n [(f + g)(z_n) + (f + g)(x_{n+1}) - 2(f + g)(x)], \quad \text{for all } n \in \mathbb{N}.
\]

(10)

Let \( x^* \) be an element in \( S_\ast \). We know that \( x_n, x \in \text{dom } g \) and \( t_{n+1} \geq 1 \), so \( (1 - \frac{1}{t_{n+1}})x_n + \frac{1}{t_{n+1}}x^* \in \text{dom } g \). Next, we put \( x = (1 - \frac{1}{t_{n+1}})x_n + \frac{1}{t_{n+1}}x^* \) in (10) and obtain the following:
\[ \|x_{n+1} - (1 - \frac{1}{t_{n+1}})x_n - \frac{1}{t_{n+1}}x^*\|^2 \leq 2\gamma_n[(f + g)(1 - \frac{1}{t_{n+1}})x_n + \frac{1}{t_{n+1}}x^*) - (f + g)(x_{n+1})] \\
+ 2\gamma_n[(f + g)(1 - \frac{1}{t_{n+1}})x_n + \frac{1}{t_{n+1}}x^*) - (f + g)(z_n)] \]
\[ \leq 2\gamma_n[1 - \frac{1}{t_{n+1}})(f + g)(x_n) + \frac{1}{t_{n+1}}(f + g)(x^*) - (f + g)(x_{n+1})] \\
+ 2\gamma_n[1 - \frac{1}{t_{n+1}})(f + g)(x_n) + \frac{1}{t_{n+1}}(f + g)(x^*) - (f + g)(z_n)] \]
\[ = 2\gamma_n[1 - \frac{1}{t_{n+1}})(f + g)(x_n) - (f + g)(x^*)] - [(f + g)(x_{n+1}) - (f + g)(x^*)] \\
+ 2\gamma_n[1 - \frac{1}{t_{n+1}})(f + g)(x_n) - (f + g)(x^*)] - [(f + g)(z_n) - (f + g)(x^*)] \]
\[ \leq 4\gamma_n[1 - \frac{1}{t_{n+1}})(f + g)(x_n) - (f + g)(x^*)] - 2\gamma_n[(f + g)(x_{n+1}) - (f + g)(x^*)], \]
for all \( n \in \mathbb{N} \). From D1 and D2, we know that \( t_n \geq 2t_{n+1} \) and \( \gamma_n \geq \gamma_{n+1} \), so:

\[ 4\gamma_n(1 - \frac{1}{t_{n+1}})(f + g)(x_n) - (f + g)(x^*)] - 2\gamma_n[(f + g)(x_{n+1}) - (f + g)(x^*)] \]
\[ \leq \frac{1}{t_{n+1}^2}[2\gamma_n(2t_{n+1}^2 - 2t_{n+1} + 1)(f + g)(x_n) - (f + g)(x^*)] \\
- 2\gamma_{n+1}t_{n+1}^2[(f + g)(x_{n+1}) - (f + g)(x^*)] \]
\[ \leq \frac{1}{t_{n+1}^2}[2\gamma_n(2t_{n+1}^2 - 2t_{n+1} + 1)(f + g)(x_n) - (f + g)(x^*)] \\
- 2\gamma_{n+1}t_{n+1}^2[(f + g)(x_{n+1}) - (f + g)(x^*)], \] for all \( n \in \mathbb{N} \).

Moreover, we also have:

\[ \|x_{n+1} - (1 - \frac{1}{t_{n+1}})x_n - \frac{1}{t_{n+1}}x^*\|^2 \leq \|\hat{x}_n - (1 - \frac{1}{t_{n+1}})x_n - \frac{1}{t_{n+1}}x^*\|^2 \]
\[ = \frac{1}{t_{n+1}^2}(\|t_{n+1}x_{n+1} - (t_{n+1} - 1)x_n - x^*\|^2) \\
- \|t_{n+1}x_n + \beta_n(t_{n+1}(x_n - x_{n-1}) - (t_{n+1} - 1)x_n - x^*\|^2) \]
\[ = \frac{1}{t_{n+1}^2}(\|t_{n+1}x_{n+1} - (t_{n+1} - 1)x_n - x^*\|^2) \\
- \|t_{n+1}x_n - (t_{n+1} - 1)x_{n-1} - x^*\|^2) \]
\[ = \frac{1}{t_{n+1}^2}(\|t_{n+1}x_{n+1} - (t_{n+1} - 1)x_n - x^*\|^2) \\
- \|t_{n+1}x_n - (t_{n+1} - 1)x_{n-1} - x^*\|^2), \]
for all \( n \in \mathbb{N} \). By (11)–(13), we obtain:

\[ \|t_{n+1}x_{n+1} - (t_{n+1} - 1)x_n - x^*\|^2 \leq 2\gamma_n^2[(f + g)(x_{n+1}) - (f + g)(x^*)] \\
- 2\gamma_{n+1}t_{n+1}^2[(f + g)(x_{n+1}) - (f + g)(x^*)], \]
for all \( n \in \mathbb{N} \). It follows that:

\[
2\gamma_{n+1}t_{n+1}^2[(f + g)(x_{n+1}) - (f + g)(x)] \leq \|t_n x_n - (t_n - 1)x_{n-1} - x^*\|^2 \\
- \|t_{n+1} x_{n+1} - (t_{n+1} - 1)x_n - x^*\|^2 \\
+ 2\gamma_{n+1}t_{n+1}^2[(f + g)(x_n) - (f + g)(x^*)],
\]

for all \( n \in \mathbb{N} \). Moreover, we can inductively prove, from (14), that:

\[
2\gamma_{n+1}t_{n+1}^2[(f + g)(x_{n+1}) - (f + g)(x^*)] \leq \|t_n x_n - (t_n - 1)x_{n-1} - x^*\|^2 \\
+ 2\gamma_{n+1}t_{n+1}^2[(f + g)(x_n) - (f + g)(x^*)] \\
\leq \|t_{n-1} x_{n-1} - (t_{n-1} - 1)x_{n-2} - x^*\|^2 \\
+ 2\gamma_{n-1}t_{n-1}^2[(f + g)(x_{n-1}) - (f + g)(x^*)] \\
\vdots \\
\leq \|t_1 x_1 - (t_1 - 1)x_0 - x^*\|^2 \\
+ 2\gamma_1t_1^2[(f + g)(x_1) - (f + g)(x^*)],
\]

for all \( n \in \mathbb{N} \). From C1, we know that \( \gamma_{n+1} \geq \gamma \). Therefore, we obtain, for all \( n \in \mathbb{N} \),

\[
(f + g)(x_{n+1}) - \min_{x \in H}(f + g)(x) \leq \frac{1}{2\gamma_{n+1}t_{n+1}^2}||x_1 - x^*||^2 \\
+ \frac{2\gamma_1t_1^2}{2\gamma_{n+1}t_{n+1}^2}[(f + g)(x_1) - (f + g)(x^*)] \\
\|x_1 - x^*\|^2 + 2\gamma_1t_1^2[(f + g)(x_1) - \min_{x \in H}(f + g)(x)] \\
\leq \frac{d(x_1, S_*)^2 + 2\gamma_1t_1^2[(f + g)(x_1) - \min_{x \in H}(f + g)(x)]}{2\gamma_{n+1}t_{n+1}^2},
\]

Since \( x^* \) is arbitrarily chosen from \( S_* \), we have:

\[
(f + g)(x_{n+1}) - \min_{x \in H}(f + g)(x) \leq \frac{d(x_1, S_*)^2}{2\gamma_{n+1}t_{n+1}^2},
\]

for all \( n \in \mathbb{N} \), and the proof is complete. \( \Box \)

**Remark 1.** To justify that there exists a sequence \( \{\beta_n\} \) satisfying D1, we choose:

\[
\beta_n = \frac{1}{(n+1)^2}, \text{ for all } n \in \mathbb{N}.
\]

We see that, for all \( n \in \mathbb{N} \),

\[
t_n = 1 + \frac{1}{(n+1)^2} + \frac{1}{(n+2)^2} + \frac{1}{(n+3)^2} + \ldots.
\]

Therefore, we have:

\[
t_{n+1} \leq t_n \text{ for all } n \in \mathbb{N}.
\]

Furthermore, it can be seen that \( t_{n+1} < 1 + \sum_{k=3}^{\infty} \frac{1}{k^2} = \frac{\pi^2}{6} - \frac{1}{4} \) for all \( n \in \mathbb{N} \). Then:

\[
2t_{n+1}^2 - 2t_{n+1} = (t_{n+1})(2t_{n+1} - 2) \\
\leq (t_n)(\frac{\pi^2}{3} - \frac{5}{2}), \text{ for all } n \in \mathbb{N}.
\]
Since $\frac{5}{4} - \frac{5}{2} < 1$, we can conclude that

$$2t_{n+1}^2 - 2t_n < t_n^2,$$

for all $n \in \mathbb{N}$. Obviously, $\sum_{k=n}^{+\infty} (\prod_{i=n}^{k} \beta_i) < +\infty$, hence D1 is satisfied.

We also note that to obtain a sequence $\{\gamma_n\}$ satisfying D2, one could simply modify Algorithm 9 by choosing $\gamma_n = \text{Line Search 3}(y_n, \gamma_{n-1}, \theta, \delta)$.

4. Applications to Data Classification and Image Restoration Problems

In this section, the proposed algorithm is used to solve classification and image restoration problems. The performance of Algorithm 9 is evaluated and compared with Algorithms 3, 5, and 6.

4.1. Data Classification

Data classification is a major branch of problems in machine learning, which is an application of artificial intelligence (AI) possessing the ability to learn and improve from experience without being programmed. In this work, we focused on one particular learning technique called extreme learning machine (ELM) introduced by Huang et al. [29]. It is defined as follows.

Let $S := \{(x_k, t_k) : x_k \in \mathbb{R}^n, t_k \in \mathbb{R}^m, k = 1, 2, ..., N\}$ be a training set of $N$ samples, where $x_k$ is an input and $t_k$ is a target. The output of ELM with $M$ hidden nodes and activation function $G$ is defined by:

$$a_j = \sum_{i=1}^{M} \eta_i G(\langle w_i, x_j \rangle + b_i),$$

where $w_i$ is the weight vector connecting the $i$-th hidden node and the input node, $\eta_i$ is the weight vector connecting the $i$-th hidden node and the output node, and $b_i$ is the bias. The hidden layer output matrix $H$ is formulated as:

$$H = \begin{bmatrix}
G(\langle w_1, x_1 \rangle + b_1) & \cdots & G(\langle w_1, x_N \rangle + b_1) \\
\vdots & \ddots & \vdots \\
G(\langle w_M, x_1 \rangle + b_1) & \cdots & G(\langle w_M, x_N \rangle + b_M)
\end{bmatrix}.$$

The main goal of ELM is to find an optimal weight $\eta = [\eta_1^T, ..., \eta_M^T]^T$ such that $H\eta = T$, where $T = [t_1^T, ..., t_N^T]^T$ is the training set. If the Moore–Penrose generalized inverse $H^+ \mathbf{H}$ of $H$ exists, then $:\eta = H^+ \mathbf{H}$ is the desired solution. However, in general cases, $H^+$ may not exist or be challenging to find. Hence, to avoid such difficulties, we applied the concept of convex minimization to find $\eta$ without relying on $H^+$.

To prevent overfitting, we used the least absolute shrinkage and selection operator (LASSO) [30], formulated as follows:

$$\min_{\eta} \{H\eta - T\|_2^2 + \lambda \|\eta\|_1\},$$

where $\lambda$ is a regularization parameter. In the setting of convex minimization, we set $f(x) = \|Hx - T\|_2^2$ and $g(x) = \lambda \|x\|_1$.

In the experiment, we aimed to classify three datasets https://archive.ics.uci.edu, accessed on 1 May 2021.

Iris dataset [31]: Each sample in this dataset has four attributes, and the set contains three classes with fifty samples for each type.

Heart disease dataset [32]: This dataset contains 303 samples, each of which has 13 attributes. In this dataset, we classified two classes of data.
Wine dataset [33]: In this dataset, we classified three classes of one-hundred seventy-eight samples. Each sample contained 13 attributes.

In all experiments, we used the sigmoid as the activation function with the number of hidden nodes $M = 30$. The accuracy of the output is calculated by:

$$\text{accuracy} = \frac{\text{correctly predicted data}}{\text{all data}} \times 100.$$  

We also utilized 10-fold cross-validation to evaluate the performance of each algorithm and used the average accuracy as the evaluation tool. It is defined as follows:

$$\text{Average ACC} = \frac{1}{N} \sum_{i=1}^{N} \frac{x_i}{y_i} \times 100\% / N,$$

where $N$ is the number of sets considered during cross-validation ($N = 10$), $x_i$ is the number of correctly predicted data at fold $i$, and $y_i$ is the number of all data at fold $i$.

We used 10-fold cross-validation to split the data into training sets and testing sets; more information can be seen in Table 1.

Table 1. Number of samples in each fold for all datasets.

|         | Iris |         | Heart Disease |         | Wine |
|---------|------|---------|---------------|---------|------|
|         | Train | Test | Train | Test | Train | Test |
| Fold 1  | 135   | 15    | 273   | 30   | 161   | 17   |
| Fold 2  | 135   | 15    | 272   | 31   | 160   | 18   |
| Fold 3  | 135   | 15    | 272   | 31   | 160   | 18   |
| Fold 4  | 135   | 15    | 272   | 31   | 160   | 18   |
| Fold 5  | 135   | 15    | 273   | 30   | 160   | 18   |
| Fold 6  | 135   | 15    | 273   | 30   | 160   | 18   |
| Fold 7  | 135   | 15    | 273   | 30   | 160   | 18   |
| Fold 8  | 135   | 15    | 273   | 30   | 160   | 18   |
| Fold 9  | 135   | 15    | 273   | 30   | 160   | 18   |
| Fold 10 | 135   | 15    | 273   | 30   | 161   | 17   |

All parameters of Algorithms 3, 5, 6 and 9 were chosen as in Table 2.

Table 2. Chosen parameters for each algorithm.

|               | Algorithm 3 | Algorithm 5 | Algorithm 6 | Algorithm 9 |
|---------------|-------------|-------------|-------------|-------------|
| $\sigma$     | 0.49        | 0.124       | 0.49        | 0.124       |
| $\delta$     | 0.1         | 0.1         | 0.1         | 0.1         |
| $\theta$     | 0.1         | 0.1         | 0.1         | 0.1         |

The inertial parameters $\beta_n$ of Algorithm 9 may vary depending on the dataset, since some $\beta_n$ work well on specific datasets. We used the following two choices of $\beta_n$ in our experiments.

$$\beta_{n}^{1} = \frac{10^{8}}{\|x_{n} - x_{n-1}\|^{3} + n^{3} + 10^{8}},$$

and

$$\beta_{n}^{2} = \begin{cases} 0.9, & \text{if } n \leq 1000 \\ \frac{1}{(n+1)^{2}}, & \text{if } n \geq 1001 \end{cases}.$$

The regularization parameters $\lambda$ for each dataset and algorithm were chosen to prevent overfitting, i.e., a model obtained from the algorithm achieves high accuracy on the training set, but low accuracy on the testing set in comparison, so it cannot be used to predict the unknown data. It is known that when $\lambda$ is too large, the model tends to underfit, i.e., low accuracy on the training set, and cannot be used to predict the future data. On the other hand, if $\lambda$ is too small, then it may not be enough to prevent a model from
overfitting. In our experiment, for each algorithm, we chose a set of $\lambda$ that satisfies $|\text{Acc}_{\text{train}} - \text{Acc}_{\text{test}}| < 2\%$, where $\text{Acc}_{\text{train}}$ and $\text{Acc}_{\text{test}}$ are the average accuracy of the training set and testing set, respectively. Under this criterion, we can prevent the studied models from overfitting. Then, from these candidates, we chose $\lambda$, which yields high $\text{ACC}_{\text{test}}$ for each algorithm. Therefore, the models obtained from Algorithms 3, 5, 6 and 9 can be effectively used to predict the unknown data.

By this process, the regularization parameters $\lambda$ for Algorithms 5, 6, and 9 were as in Table 3.

**Table 3.** Chosen $\lambda$ for each algorithm.

| Regularization Parameter $\lambda$ | Iris | Heart Disease | Wine |
|----------------------------------|------|---------------|------|
| Algorithm 3                      | 0.001| 0.003         | 0.02 |
| Algorithm 5                      | 0.01 | 0.03          | 0.006|
| Algorithm 6                      | 0.9  | 0.2           | 0.003|
| Algorithm 9                      | 0.003| 0.16          | 0.17 |

We assessed the performance of each algorithm at the 300th iteration with the average accuracy. The results can be seen in Table 4.

**Table 4.** Average accuracy of each algorithm at the 300th iteration with 10-fold cv.

| Iris | Heart Disease | Wine |
|------|---------------|------|
|      | Train | Test | Train | Test | Train | Test |
| Algorithm 3 | 92.37 | 90.67 | 81.85 | 80.52 | 97.57 | 97.16 |
| Algorithm 5 | 94.37 | 94.00 | 83.53 | 81.84 | 98.00 | 97.19 |
| Algorithm 6 | 96.67 | 96.00 | 84.42 | 83.49 | 99.25 | 98.33 |
| Algorithm 9 | 98.52 | 98.67 | 84.30 | 83.82 | 99.38 | 99.44 |

As we see from Table 4, from the choice of $\lambda$, all models obtained from Algorithms 3, 5, 6 and 9 had reasonably high average accuracy on both the training and testing sets for all datasets. Moreover, we observed that a model from Algorithm 9 performed better than the models from other algorithms in terms of the accuracy in all experiments conducted.

4.2. Image Restoration

We first recall that an image restoration problem can be formulated as a simple mathematical model as follows:

$$Ax = b + w$$

where $x \in \mathbb{R}^{n \times 1}$ is the original image, $A \in \mathbb{R}^{m \times n}$ is a blurring matrix, $b$ is an observed image, and $w$ is noise. The main objective of image restoration is to find $x$ from given image $b$, blurring matrix $A$, and noise $w$.

In order to solve (16), one could implement LASSO [30] and reformulate the problem in the following form.

$$\min_x \{ \|Ax - b\|_2^2 + \lambda \|x\|_1 \},$$

where $\lambda$ is a regularization parameter. Hence, it can be viewed as a convex minimization problem. Therefore, Algorithms 3, 5, 6 and 9 can be used to solve an image restoration problem.

In our experiment, we used the 256 $\times$ 256 color image as the original image. We used Gaussian blur of size $9^2$ and standard deviation four on the original image and obtained
the blurred image. In order to assess the performance of each algorithm, we implemented the peak-signal-to-noise ratio (PSNR) \[34\] defined by:

\[
PSNR(x_n) = 10 \log_{10} \left( \frac{255^2}{MSE} \right).
\] (18)

For any original image \(x\) and deblurred image \(x_n\), the mean squared error (MSE) is calculated by \(MSE = \frac{1}{M} \|x_n - x\|^2\), where \(M\) is the number of pixels of \(x\). We also need to mention that an algorithm with a higher PSNR performs better than one with a lower PSNR.

The control parameters of each algorithm were chosen as \(\delta = \theta = \sigma = 0.1\). As the inertial parameter \(\beta_n\) of Algorithm 9, we used the following:

\[
\beta_n = \begin{cases} 
0.95, & \text{if } n \leq 1000 \\
\frac{1}{n^2}, & \text{if } n \geq 1001.
\end{cases}
\]

As for the regularization parameter \(\lambda\), we experimented on \(\lambda\) varying from zero to one for each algorithm. In Figure 1, we show the PSNR of Algorithms 3 and 5 with respect to \(\lambda\) at the 200th iteration. In Figure 2, we show the PSNR of Algorithms 6 and 9 with respect to \(\lambda\) at the 200th iteration.

---

**Figure 1.** PSNR of Algorithm 3 (left) and Algorithm 5 (right) with respect to \(\lambda\) at the 200th iteration.

**Figure 2.** PSNR of Algorithm 6 (left) and Algorithm 9 (right) with respect to \(\lambda\) at the 200th iteration.
We observe from Figures 1 and 2 that the PSNRs of Algorithms 3, 5, 6 and 9 increased as $\lambda$ became smaller. Based on this, for the next two experiments, we chose $\lambda$ to be small to obtain a high PSNR for all algorithms. Next, we observed the PSNR of each algorithm when $\lambda$ was small ($\lambda < 10^{-4}$). In Figure 3, we show the PSNR of each algorithm with respect to $\lambda < 10^{-4}$ at the 200th iteration.

![Figure 3](image3.png)

Figure 3. A graph of the PSNR of each algorithm with respect to $\lambda$ at the 200th iteration.

We see from Figure 3 that Algorithm 9 offered a higher PSNR than the other algorithms. In the next experiment, we chose $\lambda = 5 \times 10^{-5}$ for each algorithm and evaluated the performance of each algorithm at the 200th iteration; see Table 5 for the results.

In Figure 4, we show the PSNR of each algorithm at each step of iteration. In Figure 5, we show the original test image, blurred image and deblurred images obtained from Algorithms 3, 5, 6 and 9. As we see from Table 5 and Figure 4, Algorithm 9 achieved the highest PSNR.

![Figure 4](image4.png)

Figure 4. A graph of the PSNR of each algorithm at Iteration Number 1 to 200.
Table 5. PSNR of each algorithm at the 200th iteration.

| Algorithm | PSNR (dB) |
|-----------|-----------|
| 3         | 77.62     |
| 5         | 78.55     |
| 6         | 78.95     |
| 9         | 81.29     |

Figure 5. Deblurred images of each algorithm at the 200th iteration.

5. Conclusions

We introduced a new line search technique inspired by [9,19] and used this technique along with the inertial step to construct a new accelerated algorithm for solving convex minimization problems in the form of the sum of two lower semicontinuous convex functions. We proved its weak convergence to a solution of (1), which did not require an $L$-Lipschitz continuity of $\nabla f$, as well as its complexity theorem. We note that many forward-backward-type algorithms require $\nabla f$ to be $L$-Lipschitz continuous in order to obtain the convergence theorem; see [10,16,18] for examples. Our proposed algorithm is easy to employ, since an $L$-Lipschitz constant of $\nabla f$ does not need to be calculated. Moreover, we also utilized an inertial technique to improve the convergence behavior of our proposed algorithm. In order to show that our algorithm performed better than other line search algorithms mentioned in the literature, we conducted some experiments on classification and image restoration problems. In our experiments, we evaluated the performance of each algorithm based on selecting its suitable parameters, especially the regularization parameter. It was evidenced that our proposed algorithm performed better than the other algorithms on both data classification and image restoration problems. Moreover, we observed from [19,20] that the inertial parameter $\beta_n = \frac{t_n - 1}{t_n + 1}$ of FISTA and Algorithm 6 satisfied the condition $\sup_n \beta_n = 1$, while our algorithm required a more strict condition, $\sum_{n=1}^{+\infty} \beta_n < +\infty$, to ensure its convergence to a solution of (1), which was a limitation of our
algorithm. We also note that the convergence of FISTA and Algorithm 6 cannot be obtained under the condition sup $\beta_n = 1$. Therefore, it is very interesting to find a weaker condition on $\beta_n$ that still ensure the convergence of the algorithm to a solution of (1).

**Author Contributions:** Writing—original draft preparation, P.S.; software and editing, D.C.; supervision, S.S. All authors read and agreed to the published version of the manuscript.

**Funding:** Thailand Science Research and Innovation: IRN62W0007, Chiang Mai University.

**Data Availability Statement:** All data is available at https://archive.ics.uci.edu accessed on 17 May 2021.

**Acknowledgments:** P. Sarnmeta was supported by the Post-Doctoral Fellowship of Chiang Mai University, Thailand. This research was also supported by Chiang Mai University and Thailand Science Research and Innovation under the Project IRN62W0007.

**Conflicts of Interest:** The authors declare no conflict of interest.

**References**

1. Kaltenbacher, B.; Rundell, W. On the identification of a nonlinear term in a reaction-diffusion equation. *Inverse Probl.* 2019, 35, 115007. [CrossRef]

2. Kaltenbacher, B.; Rundell, W. The inverse problem of reconstructing reaction–diffusion systems. *Inverse Probl.* 2020, 36, 065011. [CrossRef]

3. Averós, J.C.; Llorens, J.P.; Uribe-Kaffure, R. Numerical simulation of non-linear models of reaction-diffusion for a DGT sensor. *Algorithms* 2020, 13, 98. [CrossRef]

4. Lukyanenko, D.; Yeleskina, T.; Prigorniy, I.; Isaev, T.; Borzunov, A.; Shishlenin, M. Inverse problem of recovering the initial condition for a nonlinear equation of the reaction-diffusion-advection type by data given on the position of a reaction front with a time delay. *Mathematics* 2021, 9, 342. [CrossRef]

5. Lukyanenko, D.; Borzunov, A.; Shishlenin, M. Solving coefficient inverse problems for nonlinear singularly perturbed equations of the reaction-diffusion-advection type with data on the position of a reaction front. *Commun. Nonlinear Sci. Numer. Simul.* 2021, 99, 105824. [CrossRef]

6. Egger, H.; Engl, H.W.; Klibanov, M.V. Global uniqueness and Hölder stability for recovering a nonlinear source term in a parabolic equation. *Inverse Probl.* 2004, 21, 271. [CrossRef]

7. Beilina, L.; Klibanov, M.V. A globally convergent numerical method for a coefficient inverse problem. *SIAM J. Sci. Comput.* 2008, 31, 478–509. [CrossRef]

8. Klibanov, M.V.; Li, J.; Zhang, W. Convexification for an inverse parabolic problem. *Inverse Probl.* 2020, 36, 085008. [CrossRef]

9. Kankam, K.; Pholasa N.; Cholamjiak, C. On convergence and complexity of the modified forward–backward method involving new line searches for convex minimization. *Math. Methods Appl. Sci.* 2019, 42, 1352–1362. [CrossRef]

10. Combettes, P.L.; Wajs, V. Signal recovery by proximal forward–backward splitting. *Multiscale Model. Simul.* 2005, 4, 1168–1200. [CrossRef]

11. Luo, Z.Q. Applications of convex optimization in signal processing and digital communication. *Math. Program.* 2003, 97, 177–207. [CrossRef]

12. Xiong, K.; Zhao, G.; Shi, G.; Wang, Y. A convex optimization algorithm for compressed sensing in a complex domain: The complex-valued split Bregman method. *Sensors* 2019, 19, 4540. [CrossRef]

13. Chen, M.; Zhang, H.; Lin, G.; Han, Q. A new local and nonlocal total variation regularization model for image denoising. *Cluster Comput.* 2019, 22, 7611–7627. [CrossRef]

14. Zhang, Y.; Li, X.; Zhao, G.; Cavalcante, C.C. Signal reconstruction of compressed sensing based on alternating direction method of multipliers. *Circuits Syst. Signal Process* 2020, 39, 307–323. [CrossRef]

15. Parekh, A.; Selesnick, I.W. Convex fused lasso denoising with non-convex regularization and its use for pulse detection. In Proceedings of the 2015 IEEE Signal Processing in Medicine and Biology Symposium, Philadelphia, PA, USA, 12 December 2015; pp. 21–30.

16. Hanjing, A.; Suantai, S. A fast image restoration algorithm based on a fixed point and optimization method. *Mathematics* 2020, 8, 378, doi10.3390/math8030378. [CrossRef]

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

18. Bui, R.L.; Csetnek, E.R. An inertial forward–backward-forward primal-dual splitting algorithm for solving monotone inclusion problems. *Numer. Algorithms* 2016, 71, 519–540. [CrossRef]

19. Beck, A.; Teboulle, M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sci.* 2009, 2, 183–202. [CrossRef]
20. Bello Cruz, J.Y.; Nghia, T.T. On the convergence of the forward–backward splitting method with line searches. *Optim. Methods Softw.* **2016**, *31*, 1209–1238. [CrossRef]
21. Polyak, B.T. Some methods of speeding up the convergence of iteration methods. *USSR Comput. Math. Math. Phys.* **1964**, *4*, 1–17. [CrossRef]
22. Attouch, H.; Cabot, A. Convergence rate of a relaxed inertial proximal algorithm for convex minimization. *Optimization* **2019**, *69*, 1281–1312. [CrossRef]
23. Alvarez, F.; Attouch, H. An inertial proximal method for maximal monotone operators via discretization of a nonlinear oscillator with damping. *Set-Valued Anal.* **2001**, *9*, 3–11. [CrossRef]
24. Van Hieu, D. An inertial-like proximal algorithm for equilibrium problems. *Math. Methods Oper. Res.* **2018**, *88*, 399–415. [CrossRef]
25. Chidume, C.E.; Kumam, P.; Adamu, A. A hybrid inertial algorithm for approximating solution of convex feasibility problems with applications. *Fixed Point Theory Appl.* **2020**, 2020. [CrossRef]
26. Takahashi, W. *Introduction to Nonlinear and Convex Analysis*; Yokohama Publishers: Yokohama, Japan, 2009.
27. Moudafi, A.; Al-Shemas, E. Simultaneous iterative methods for split equality problem. *Trans. Math. Program. Appl.* **2013**, *1*, 1–11.
28. Huang, G.B.; Zhu, Q.Y.; Siew, C.K. Extreme learning machine: Theory and applications. *Neurocomputing* **2006**, *70*, 489–501. [CrossRef]
29. Fisher, R.A. The use of multiple measurements in taxonomic problems. *Ann. Eugen.* **1936**, *7*, 179–188. [CrossRef]
30. Detrano, R.; Janosi, A.; Steinbrunn, W.; Pfeferer, M.; Schmid, J.J.; Sandhu, S.; Guppy, K.H.; Lee, S.; Froelicher, V. International application of a new probability algorithm for the diagnosis of coronary artery disease. *Am. J. Cardiol.* **1989**, *64*, 304–310. [CrossRef]
31. Forina, M.; Leardi, R.; Armanino, C.; Lanteri, S. *PARVUS: An Extendable Package of Programs for Data Exploration*; Elsevier: Amsterdam, The Netherlands, 1988.
32. Thung, K.-H.; Raveendran, P. A Survey of Image Quality Measures. In Proceedings of the IEEE Technical Postgraduates (TECHPOS) International Conference, Kuala Lumpur, Malaysia, 14–15 December 2009; pp. 1–4.