Journal of Industrial and Management Optimization

https://www.aimspress.com/journal/jimo

JIMO, 22(8): 3605–3667.
DOI:10.3934/jimo.2026131
Received: 14 March 2026
Revised: 21 May 2026
Accepted: 10 June 2026
Published: 09 July 2026

Research article

Fast and efficient optimization algorithms for split variational inequality problem with applications in data classification problems

Lovelyn Madu[1], **Olaniyi Iyiola**[2,*] **and Bruce Wade**[1]

[1] Department of Mathematics, University of Louisiana at Lafayette, 1401 Johnston St. Lafayette, LA 70503, USA

[2] Department of Mathematics, Morgan State University, 1700 East Cold Spring Lane, Baltimore, MD 21251, USA

* **Correspondence:** Email: olaniyi.iyiola@morgan.edu, niyi4oau@gmail.com.

Abstract: Split variational inequality problems (SVIPs) have become a comprehensive framework for representing intricate coupled systems in fields such as optimization, machine learning, signal processing, and inverse problems. In this study, we present three innovative Mann-type iterative methods that integrate self-adjusting step-size selection and inertial dynamics to address SVIPs. Theoretically, we proved weak convergence theorems for each of these algorithms under mild and standard conditions, expanding the traditional results by permitting inertial parameters to surpass the usual range of [0,1) without necessitating restrictive on-line rules. The algorithms were developed to overcome significant computational hurdles in large-scale nonlinear models, and were particularly effective for data-driven applications. Inspired by practical biomedical classification challenges, we incorporated these algorithms into the training process of the Extreme Learning Machine (ELM) to determine optimal output weights with enhanced numerical stability and precision. This improvement significantly boosted the predictive capabilities of ELM models for early disease detection. Empirical tests on four standard medical datasets, heart disease, lung cancer, heart failure, and prostate cancer, showed notable improvements in classification accuracy and robustness. The blend of relaxed inertial conditions, adaptive step-size updating, and Mann-type structures provided a versatile and efficient framework for solving SVIPs and advancing machine learning-based disease diagnostics.

Keywords: optimization; inertial algorithm; split variational inequality; data classification; disease prediction

Mathematics Subject Classification: 65K15, 47J25, 65J15, 90C33

1. Introduction

The split variational inequality problem (SVIP) in Hilbert spaces is a fundamental mathematical framework with broad applications in optimization and machine learning. In this work, we propose an efficient algorithm designed to solve SVIP with improved computational speed and stability. Existing methods often impose restrictive conditions on convergence, limiting their practical applicability in complex real-world scenarios. Our approach introduces an enhanced inertial-type algorithm, which enables more flexible inertial choices, surpassing conventional constraints typically found in the literature. This advancement enables better adaptability and performance, particularly in predictive modeling for critical medical conditions such as heart disease, heart failure, lung cancer, and prostate cancer.
Let H_1 and H_2 be two real Hilbert spaces. Given nonlinear operators $f : H_1 \to H_1$ and $g : H_2 \to H_2$, a bounded linear operator $A : H_1 \to H_2$, and nonempty, closed and convex subset $C \subseteq H_1$ and $Q \subseteq H_2$, the Split variational inequality problem (SVIP) is formulated as follows:
find a point $x^* \in C$ such that

$$\langle f(x^*), x - x^* \rangle \geq 0 \; for \; all \; x \in C. \tag{1.1}$$

and such that the point

$$y^* = Ax^* \in Q \; solves \; \langle g(y^*), y - y^* \rangle \geq 0 \; for \; all \; y \in Q. \tag{1.2}$$

It is worth noting that Equations (1.1) and (1.2) represent the classical Variational Inequality Problem (VIP), which have been extensively investigated in the literature (see, for instance, [1–10]). Viewed independently, the Split Variational Inequality (SVIP) consists of two VIPs: One solved in a given space H_1, and its image under bounded linear operator serving as the solution to the other VIP in a different space H_2. The SVIP defined by Equations (1.1) and (1.2) may be interpreted as the synthesis of the classical VIP (1.1) and the split feasibility problem introduced in [11]. The split variational inequality problem (SVIP) arises naturally in many real-world applications where two or more coupled constraints must be satisfied simultaneously. Its modeling power has led to broad use across domains, including intensity-modulated radiation therapy (IMRT), data compression, medical image reconstruction, signal processing, watermarking, material imaging, deep neural networks, and phase retrieval (see [11–14]). These applications demonstrate that SVIP provides a unifying mathematical framework for solving inverse problems, feasibility problems, and learning tasks that involve interacting operators or constraints. Motivated by these connections, we consider the extreme learning machine (ELM), a fast and widely used learning model in signal-processing and imaging pipelines due to its computational efficiency and strong generalization performance. By formulating the ELM training problem within the SVIP framework, we aim to leverage modern inertial and projection-type methods to improve stability, convergence, and predictive accuracy. Wickramasinghe et al. [15] considered ELM application in an inverse problem algorithm; further details can be found in their work.
Byrne [16] proposed the following CQ algorithm for solving the Split feasibility problem:

$$x_{n+1} = P_C(x_n - \tau_n A^*(I - P_Q)Ax_n), \tag{1.3}$$

where $\tau_n \in \left(0, \frac{2}{\|A\|^2}\right)$.
Hendrickx and Olshevsky in [17] observed that computing $\|A\|$ to select step - size τ_n while implementing Equation (1.3) is a difficult even sometimes impossible. This limitation in the CQ Algorithm Equation

(1.3) has been removed by several authors by introducing a step-size τ_n independent of $\|A\|$ during computations. For instance Lopez et al [18] chose τ_n as

$$\tau_n = \frac{\rho_n \|(I - P_Q)Ax_n\|^2}{\|A^*(I - P_Q)Ax_n\|^2}, \tag{1.4}$$

where $\rho_n \in (0, 2)$. Other improvements of Equation (1.3) with (1.4) can be found in [19–26].
Dang [27] studied the following CQ algorithms with inertial extrapolation step for SFP:

$$x_{n+1} = P_{C_n}(y_n - \tau A^*(I - P_{Q_n})Ay_n), \tag{1.5}$$

and

$$x_{n+1} = (1 - \beta_n)y_n + \beta_n P_{C_n}(y_n - \tau A^*(I - PQ_n)Ay_n), \tag{1.6}$$

where $y_n = x_n + \theta_n(x_n - x_{n-1})$, $\beta_n \in (0, 1)$, $\tau \in (0; \frac{2}{\|A\|^2})$ and $\theta_1 \in [0, \bar{\theta_1}]$ with $\overline{\theta}_n := \min\{\theta, (\max\{n^2\|y_n - y_{n-1}\|^2, n^2\|y_n - y_{n-1}\|\})^{-1}$, $\theta \in [0, 1)$.
Cholamjiak et al. [28] proposed the following CQ Algorithms with self adaptive step size for solving SFP

$$\begin{cases} y_n = (1 - \alpha)x_n + \alpha_n z_n, \\ x_{n+1} = P_C(y_n - \tau_n A^*(I - P_Q)Ay_n)\Big), \\ z_{n+1} = (1 - \beta_n)z_n + \beta_n x_n, \end{cases} \tag{1.7}$$

where

$$\tau_{n+1} = \begin{cases} 0, \quad if \quad y_n \in C \cap A^{-1}(Q), \\ \frac{\rho_n \|(I - P_Q)Ay_n\|^2}{\|A^*(I - P_Q)Ay_n\|^2}, \quad otherwise, \end{cases}$$

under the assumption that $0 < \liminf_{n\to\infty} \rho_n \leq \limsup_{n\to\infty} \rho_n < 2$ and the set of solutions, $C \cap A^{-1}(Q)$ to SFP is nonempty, they proved that the sequences $\{x_n\}$ and $\{z_n\}$ generated by Equation 1.7 with $0 \leq \beta_n \leq \limsup_{n\to\infty} \beta_n < 1$ converge weakly to a point in $C \cap A^{-1}(Q)$.
Many authors have also shown theoretically and numerically that Algorithms with an inertial extrapolation step converge numerically faster in terms of the number of iterations and CPU time in most cases over the base algorithms, as investigated in [29–38].
Cholamjiak et al. [28] presented an inertial-type CQ Algorithm as a special case of the Algorithm given by Equation 1.7 as follows:

$$\begin{cases} y_{n-1} = z_n + \Theta_{n-1}(z_n - z_{n-1}), \\ x_n = P_C(y_{n-1} - \tau_{n-1} A^*(I - P_Q)(Ay_{n-1})\Big), \\ z_{n+1} = (1 - \beta_n)z_n + \beta_n x_n, \end{cases} \tag{1.8}$$

where

$$\Theta_{n-1} := \frac{1 - 2\beta_{n-1}}{\beta_{n-1}},$$

and

$$\tau_n = \begin{cases} 0, & if \quad y_{n-1} \in C \cap A^{-1}(Q), \\ \frac{\rho_n \|(I-P_Q)Ay_{n-1}\|^2}{\|A^*(I-P_Q)Ay_{n-1}\|^2}, & otherwise, \end{cases}$$

given z_{n-1} and z_n. The sequences $\{x_n\}$ and $\{z_n\}$ generated by Equation (1.8) converge weakly to a point in $C \cap A^{-1}(Q)$ when $0 \leq \liminf_{n\to\infty} \leq \beta_n \leq \limsup_{n\to\infty} < 1$.

By simple modification, Cholamjiak et al. [28] developed another algorithm that has the same computational structure as Equation (1.7). The only difference is that $z_{n+1} = (1-\beta_n)z_n + \beta_n x_n$ in Equation (1.7) is now presented as $z_{n+1} = (1-\beta_n)z_n + \beta_n x_{n+1}$. Under the same assumption as in Equation 1.7, they proved that weak convergence in $C \cap A^{-1}(Q)$ with $0 \leq \beta_n \leq \beta_{n+1} \leq 1$.

Xu et al. [39] developed the Halpern-type algorithm to approximate the solution of the split feasibility problem as follows:

While $\sigma = \|(I - P_C)x_n + A^*(I - P_Q)Ax_n\| \neq 0$ do

$$\begin{aligned} y_n &= (I - P_C)x_n + A^*(I - P_Q)Ax_n \\ x_{n+1} &= \beta_n\omega + (1-\beta_n)(x_n - \tau_n y_n) \end{aligned} \tag{1.9}$$

where $\tau_n = \frac{\|(I-P_C)x_n\|^2 + \|(I-P_C)Ax_n\|^2}{2\|(I-P_C)x_n + A^*(I-P_Q)Ax_n\|^2}$.

As a generalization of SFP, Censor [14] introduced the Split Variational Inequality Problem (SVIP). They investigated the SVIP under the assumptions that the operators are monotone and Lipschitz continuous. Their approach began by reformulating the problem as an equivalent Constrained Variational Inequality Problem (CVIP) in the product space $H_1 \times H_2$ (see [14], Section 4), which was then solved using the well-known subgradient extragradient method. However, the product-space formulation presents certain drawbacks: It can be computationally demanding to project onto newly defined product subspaces of $H_1 \times H_2$; it fails to fully exploit the inherent splitting structure of the SVIP (1.1)-(1.2), and it may complicate the translation of results back to the original spaces H_1 and H_2 (see, for example, [14], p.12). To address these limitations, the authors proposed an alternative projection method that avoids product-space reformulation. This approach is more straightforward to implement, as it circumvents the difficulties associated with product-space projections. Specifically, for $x_1 \in H_1$, the sequence $\{x_n\}$ is generated by the following projection-based algorithm

$$x_{n+1} = G(y_n - \gamma A^*(I - T)(Ay_n)) \tag{1.10}$$

where $\gamma \in (0, \frac{1}{L})$, L is the spectral radius of A^*A, and A^* is the adjoint of A, I is the identity operator $G := P_C(I - \lambda f)$ and $T := P_Q(I - \lambda g)$ are projections onto C, Q, respectively. In its result, it was established that the sequence $\{x_n\}_{n=0}^{\infty}$ generated by Equation (1.10) converges weakly to a point in the solution set of the SVIP (1.1)-(1.2). They assumed that the solution set of problem (1.1)-(1.2) is nonempty and f, g are α_1, α_2 inverse strongly monotone, respectively, $\lambda \in [0, 2\alpha]$, $\alpha := \min\{\alpha_1, \alpha_2\}$ and for every x that solves (1.1), the following conditions holds:

$$\langle f(x), G(x) - x^* \rangle \geq 0, \ x \in H_1. \tag{1.11}$$

It should be noted that although Equation (1.10) effectively exploits the splitting structure of the SVIP (1.1)-(1.2) and avoids the need for the product-space reformulation, the method depends on the spectral

radius of $A \cdot A^*$, which can be computationally demanding. He et al. [40] introduced a novel method for addressing SVIP (1.1) - (1.2) as follows: Given a symmetric positive definite matrix $H \in \mathbb{R}^{m\times m}$, $\gamma \in (0,2)$, $\rho \in (\rho_{\min}, 3)$, where $\rho_{\min} := \max\left\{-3, \frac{2(v-1)\mu_1}{4\zeta_{\max}(A^*HA)}\right\}$ and $A : \mathbb{R}^N \to \mathbb{R}^m$ is a linear operator (A^* is the transpose of A). Pick an initial point $u_0 := (x_0, y_0, \lambda_0) \in \Omega = C \times Q \times \mathbb{R}^m$, where C and Q are nonempty closed and convex subsets of $\mathbb{R}^N$ and $\mathbb{R}^m$, respectively. Generate a predictor $\tilde{u}_n = (\tilde{x}_n, \tilde{y}_n, \tilde{\lambda}_n)$ with appropriate parameters μ_1 and μ_2.

$$\begin{cases} \bar{\lambda}_n = \lambda_n - H(Ax_n - y_n), \\ \tilde{x}_n = P_C\Big[x_n - \frac{1}{\mu_1}(f(x_n) - A^*\bar{\lambda}_n\Big], \\ \hat{\lambda}_n = \lambda_n - H(A\hat{x}_n - y_n), \ \textit{where } \hat{x}_n := \rho x_n + (1-\rho)\tilde{x}_n, \\ \tilde{y}_n = P_Q\Big[y_n - \frac{1}{\mu_2}(g(y_n) + \hat{\lambda}_n)\Big], \\ \tilde{\lambda}_n = \lambda_n - H(A\tilde{x}_n - \tilde{y}_n). \end{cases} \tag{1.12}$$

$u_{n+1} := (x_{n+1}, y_{n+1}, \lambda_{n+1})$ via $u_{n+1} := \gamma\alpha_k d(u_n, \tilde{u}_n$, where

$$\begin{cases} \alpha_n := \frac{\varphi(u_n,\tilde{u}_n)}{\|d(u_n,\tilde{u}_n)\|^2}, \\ d(u_n, \tilde{u}_n) := G(u_n - \tilde{u}_n) - \xi_n, \\ \varphi(u_n, \tilde{u}_n) := \left\langle \lambda_n - \tilde{\lambda}_n, \rho A(x_n - \tilde{x}_n) - (y_n - \tilde{y}_n)\right\rangle + \langle u_n - \tilde{u}_n, d(u_n, \tilde{u}_n)\rangle, \end{cases}$$

$$\xi_n := \begin{pmatrix} \xi_{n_x} \\ \xi_{n_y} \\ 0 \end{pmatrix} := \begin{pmatrix} f(x_n) - f(\tilde{x}_n + A^*HA(x_n - \tilde{x}) \\ g(y_n) - g(\tilde{y}_n + H(y_n - \tilde{y}_n) \\ 0 \end{pmatrix}$$

$$G := \begin{pmatrix} \mu_1 I_N + \rho A^*HA & 0 & 0 \\ 0 & \mu_2 I_m + H & 0 \\ 0 & 0 & H \end{pmatrix}$$

is the block diagonal matrix with identity matrices I_N and I_m size N and m, respectively. Parameters μ_1 and μ_2 are chosen such that $\|\xi_{n_x}\| \le v\mu_1\|x_n - \tilde{x}_n\|$ and $\|\xi_{n_y}\| \le v\mu_2\|y_n - \tilde{y}_n\|$, where $v \in (0,1)$. They assume that f and g are monotone and Lipschitz continuous in finite-dimensional spaces. Their approach employs decomposition techniques and effectively leverages the inherent splitting structure of the SVIP. Nonetheless, a notable limitation is that the method requires reformulating the problem as an equivalent VIP within a product space.

Some authors have made significant contributions to the study of split variational inequality and inclusion problems. Tan et al. [41] introduced four self-adaptive inertial iterative algorithms for solving split variational inclusion problems in Hilbert spaces. Their methods were successfully applied to the split feasibility problem, the split minimization problem, and compressed sensing. Zhou et al. [42] proposed two adaptive inertial schemes based on the Meir–Keeler contraction and a Mann-type framework, and demonstrated their effectiveness in signal recovery applications. Ofem et al. [43] developed a modified subgradient extragradient method that incorporates the Mann technique and doubles inertial steps to approximate the common minimum-norm solution of split variational inequality and fixed point problems.

Building on the preceding review, an important research question arises:

Question: Is it possible to develop a novel iterative scheme that integrate an inertia mechanism with

self adaptive step sizes to solve SVIP (1.1)-(1.2) while avoiding its reformulation as an equivalent CVIP in the product space?
The importance of this study lies in the development of innovative iterative methods designed to solve SVIP with accuracy and efficiency, thus providing a comprehensive framework that enhances the Extreme Learning Machine (ELM), a widely recognized predictive model. Despite its popularity, the ELM has been criticized for inefficiency and instability stemming from its reliance on a linear system to compute output weights. Consequently, establishing robust iterative approaches to address these shortcomings is essential, particularly in contexts where precision and stability are critical.

In this work, we offer a positive resolution to the preceding question by introducing a novel iterative method characterized by the following features:

- The proposed approach utilizes a Mann-type iterative scheme to approximate the solution of (1.1)–(1.2).
- The proposed method requires only a single space projection, thereby preserving the intrinsic splitting structure of the SVIP.
- It accommodates inertial parameters beyond the conventional range $[0, 1)$, in contrast to approaches reported in the literature.
- In convergence analysis, the method does not impose an on-line rule; instead, weak convergence is established under relaxed conditions.
- Furthermore, the method does not rely on knowledge of the operator's norm for its execution. Rather, it employs self-adaptive step sizes together with an inertial mechanism to accelerate the rate of convergence.

Finally, we present a series of numerical experiments and demonstrate the applicability of our results to a practical medical diagnostic problem. Our findings improve not only on the results of Censor et al. [14] and He et al. [40], but also extend the contributions of Byrne [16], Dang [27], Cholamjiak et al. [28], Xu et al. [39], among others.

The remainder of this paper is organized as follows: In section 2, we introduce the necessary definitions and Lemmas for analyzing the convergence of the proposed algorithm. In section 3, we present the algorithm and provide a detailed convergence analysis. In Section 4, we offer several numerical examples accompanied by graphical illustrations. In section 5, we report numerical experiments based on real-world data, together with their graphical representations. Finally, in section 6, we give the concluding remarks.

2. Preliminaries

Throughout this article, H is a real Hilbert space that is equipped with the inner product $\langle ., . \rangle$ and the corresponding norm $\|.\|$, and $C \subset H$ is non-empty and closed.

2.1. Important definitions

We recall the well-known definitions and concepts needed to establish our main results.

Definition 2.1. [44] Let H be a real Hilbert space. A mapping $T : H \to H$ is nonexpansive if

$$\|Tu - Tv\| \leq \|u - v\|, \ for\ all\ u, v \in H.$$

Definition 2.2. [45] Let H be a real Hilbert space. A mapping $T : H \to H$ is firmly nonexpansive if

$$\|Tu - Tv\|^2 \leq \|u - v\|^2 - \|(I - T)u - (I - T)v\|^2\ for\ all\ u, v \in H \Longleftrightarrow \|Tu - Tv\|^2 \leq \langle u - v, Tu - Tv\rangle$$

$for\ all\ u, v \in H.$

Definition 2.3. [46] Let H denote a real Hilbert space with inner product $\langle\cdot,\cdot\rangle$ and induced norm $\|\cdot\|$. Consider C as a nonempty, closed, and convex subset of H and $A : C \to H$ a continuous mapping. The variational inequality problem (for short, VI(A, C)) is defined as: find $x \in C$ such that

$$\langle Ax, y - x\rangle \geq 0, \ \forall\ y \in C. \tag{2.1}$$

We assume throughout this paper that S denotes the set of solutions of VI(A, C) Equation (2.1).

Definition 2.4. [47] Let $T : H \to H$ be a mapping. A point $x^* \in H$ is called a fixed point of T if and only if $Tx^* = x^*$.

Definition 2.5. [45] Let H denote a real Hilbert space with inner product $\langle\cdot,\cdot\rangle$ and induced norm $\|\cdot\|$. Consider C as a nonempty, closed, and convex subset of H. P_C is called the metric projection of H onto C if and only if for any point $u \in H$, there exists a unique point $P_C u \in C$ such that

$$\|u - P_C u\| \leq \|u - y\|, \ \forall\ y \in C.$$

Definition 2.6. [45] Let H be a real Hilbert space. A mapping $A : H \to H$ is called

(1) *monotone mapping* if for all $x, y \in H$,

$$\langle x - y, Ax - Ay\rangle \geq 0. \tag{2.2}$$

(2) η *-strongly monotone* on H if there exists constant $\eta > 0$ such that $\langle Ax - Ay, x - y\rangle \geq \eta\|x - y\|^2$ for all $x, y \in H$.

(3) δ *- strongly pseudo-monotone* on H if there exists $\delta > 0$ such that $\langle Ay, x - y\rangle \geq 0 \Rightarrow \langle Ax, x - y\rangle \geq \delta\|x - y\|^2,\ x, y \in H$.

(4) *inverse strongly monotone (ISM)* with constant η on $C \subseteq H$ if $\langle A(x) - A(y), x - y\rangle \geq \eta\|A(x) - A(y)\|^2$ for $x, y \in C$.

(5) *pseudo-monotone* on H if, for all $x, y \in H, \langle Ax, y - x\rangle \geq 0 \Rightarrow \langle Ay, y - x\rangle \geq 0$.

(5) L-Lipschitz continuous on H if there exists a constant $L > 0$ such that $\|Ax - Ay\| \leq L\|x - y\|$ for all $x, y \in H$.

(6) *sequentially weakly continuous* if for each sequence x_n we have: x_n converges weakly to x implies Ax_n converges weakly to Ax.

Definition 2.7. [11] Split Feasibility Problem (SFP) in Hilbert spaces H_1, H_2 is to find $u \in C$ such that

$$Au \in Q, \tag{2.3}$$

with $A : H_1 \to H_2$ bounded linear, $\emptyset \neq C \subseteq H_1$ and; $\emptyset \neq Q \subseteq H_2$, closed and convex sets.

Definition 2.8. [14] Let VI(C,f) and VI(Q,g) be the solution set of Equation (1.1) and (1.2) respectively. Observe that the solution set of the SVIP is $\Gamma := \Gamma(C, Q, f, g, A) := \{x^* \in VI(C, f) \mid Ax^* \in VI(Q, g)\}$.

Remark 2.9. (i) We know that P_C is a nonexpansive mapping of H onto C.

(ii) Goebel and Kirk [47] established that T is firmly nonexpansive $\Longleftrightarrow$ $I - T$ is firmly nonexpansive.

(iii) Given $x \in H$ and $C \subset H$ nonempty closed convex, there exists $P_C x$, unique in C (called metric projection) such that

$$\|x - P_C x\| \leq \|x - y\| \; for \; all \; y \in C. \tag{2.4}$$

(iv) It was established in Bauschke and Combettes [45] that P_C is a firmly nonexpansive mapping of H onto C. Furthermore,

$$\langle x - y, P_C x - P_C y\rangle \geq \|P_C x - P_C y\|^2 \; for \; all x, y \in H; \tag{2.5}$$

and

$$\langle x - P_C x, P_C x - y \geq 0 \; for \; all \; y \in C. \tag{2.6}$$

Thus,

$$\|x - y\|^2 \geq \|x - P_C x\|^2 + \|y - P_C x\|^2 \; for \; all x \in H, \; y \in C. \tag{2.7}$$

(v) It is well known that

$$x^* \in VI(C, f) \Longleftrightarrow x^* = P_C(x^* - \lambda f(x^*)). \tag{2.8}$$

That is $x^* \in Fix(P_C(I - \lambda f))$.

2.2. Useful Lemmas

The following Lemma are useful:

Lemma 2.10. *[14] Let $C \subseteq H$ be a nonempty, closed, and convex subset, and let $h : H \to H$ be an α-ISM operator on H. If $\lambda \in [0, 2\alpha]$, then*

(i) Operator $P_C(I - \lambda h)$ is nonexpansive on C.

(ii) If, in addition for all $x^* \in VI(C, h)$,

$$\langle h(x), P_C(I - \lambda h)(x) - x^*\rangle \geq 0 \; for \; all \; x \in H, \tag{2.9}$$

then the following inequalities hold:

(iia) For all $x \in H$ and $q \in Fix(P_C(I - \lambda h))$,

$$\langle P_C(I - \lambda h)(x) - x, P_C(I - \lambda h)(x) - q\rangle \leq 0. \tag{2.10}$$

(iib) For all $x \in H$ and $q \in Fix(P_C(I - \lambda h))$,

$$\|P_C(I - \lambda h)(x) - q\|^2 \leq \|x - q\|^2 - \|P_C(I - \lambda h)(x) - x\|^2. \tag{2.11}$$

Lemma 2.11. *[48] Given $p; q \in H$, then*

(i) $\|p + q\|^2 = \|p\|^2 + 2\langle p, q\rangle + \|q\|^2$.
(ii) $\|p + q\|^2 \leq \|p\|^2 + 2\langle q, p + q\rangle$.
(iii) $\|\alpha p + (1 - \alpha)q\|^2 = \alpha\|p\|^2 + (1 - \alpha)\|q\|^2 - \alpha(1 - \alpha)\|p - q\|^2$; *for all* $\alpha \in \mathbb{R}$.

Lemma 2.12. *[49] Let C be a nonempty subset of H and let $\{x_n\}$ be a sequence in H such that the following two conditions hold:*

(i) For any $x \in C$, $\lim_{n\to\infty} \|p_n - p\|$ exists.
(ii) Every sequential weak cluster point of $\{p_n\}$ is in C.

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

Lemma 2.13. *[14] Let H_1 and H_2 be real Hilbert spaces and let $A : H_1 \to H_2$ be a bounded linear operator. Let $f : H_1 \to H_1$ and $g : H_2 \to H_2$ be α_1-ISM and α_2-ISM operators on H_1 and H_2, respectively, and set $\alpha := \min\{\alpha_1, \alpha_2\}$. Assume that $\Gamma \neq$ and that $\rho \in (0, \frac{1}{L})$. Consider the operators $G = P_C(I - \lambda f)$ and $T = P_Q(I - \lambda g)$ with $\lambda \in [0, 2\alpha]$. Then any sequence $\{x_n\}_{n=0}^{\infty}$, generated by Equation* (1.10)*, is Fejér-monotone with respect to the solution set Γ. i.e., for every $u \in C$,*

$$\|x_{n+1} - u\| \leq \|x_n - u\| \; for\; all\; n \geq 0, \tag{2.12}$$

3. Main results: proposed algorithms and convergence analysis

In this section, we present our proposed method for solving the SVIP Equation (1.1) and Equation (1.2). We analyze the convergence of the algorithm under the following conditions.

3.1. Proposed algorithms

Let H_1 and H_2 be real Hilbert spaces, and $A : H_1 \to H_2$ is a bounded linear operator. Let $f : H_1 \to H_1$ be α_1-inverse strongly monotone operator on H_1 and $g : H_2 \to H_2$ be α_2-inverse strongly monotone operator on H_2, and set $\alpha = \min\{\alpha_1, \alpha_2\}$. Assume further that for all $x^* \in VI(C, f)$

$$\langle f(x), G(x) - x^*\rangle \geq 0. \tag{3.1}$$

The given parameters satisfy the following conditions:

- $\beta_n \in [0, 1)$.
- $\lambda \in [0, 2\alpha]$ and define $G := P_C(I - \lambda f)$ and $T := P_Q(I - \lambda g)$.

In addition, we make the following assumptions.

Assumption A:

(A1) $A : H \to H$ is *monotone* on C.

(A2) A is *L- Lipschitz continuous* on H and A satisfies the following condition: whenever $\{x_n\} \subset H$ and x_n converges weakly to x^*, it follows that $Ax^* \leq \liminf_{n\to\infty} Ax_n$.

(A3) $0 < \liminf_{n\to\infty} \rho_n \leq \limsup_{n\to\infty} \rho_n < 2$.

(A4) The solution set Γ of SVIP is nonempty.

Now, we propose the following three algorithms.

Algorithm 3.1 Subgradient extragradient method with adaptive step size

1: Choose $\beta_n \in [0, 1)$. Select initial data $x_0 = \omega_0 \in H$ arbitrarily and set $n = 0$.

2: Given $x_n,\ \omega_n$, compute

$$y_n = (1 - \beta_n)x_n + \beta_n\omega_n, \tag{3.2}$$

3: Compute

$$x_{n+1} = G\Big(y_n - \tau_n A^*(I - T)(Ay_n)\Big), \tag{3.3}$$

where

$$\tau_n = \begin{cases} 0, & if \quad y_n \in \Gamma, \\ \frac{\rho_n\|(I-T))Ay_n\|^2}{\|A^*((I-T))Ay_n\|^2}, & otherwise. \end{cases} \tag{3.4}$$

4: Compute

$$\omega_{n+1} = (1 - \beta_n)w_n + \beta_n x_n, \tag{3.5}$$

5: Set $n \leftarrow n + 1$, and go to number 2.

We consider another new accelerated Algorithm below for SVIP Equation (1.1) and Equation (1.2).

Algorithm 3.2 Inertia - type algorithm

1: Select initial data $\omega_0 = \omega_{-1} \in H$, Select inertial parameter $\beta_n \in (0, 1)$, set $n = 0$.

2: Given ω_{n-1} and ω_n, compute

$$y_{n-1} = \omega_n + \Theta_{n-1}(\omega_n - \omega_{n-1}), \tag{3.6}$$

where

$$\Theta_{n-1} := \frac{1 - 2\beta_{n-1}}{\beta_{n-1}}. \tag{3.7}$$

3: Compute

$$x_n = G\Big(y_{n-1} - \tau_{n-1}A^*(I - T)(Ay_{n-1})\Big), \tag{3.8}$$

where

$$\tau_{n-1} = \begin{cases} 0, & if \quad y_{n-1} \in \Gamma, \\ \frac{\rho_n \|(I-T)Ay_{n-1}\|^2}{\|A^*(I-T)Ay_{n-1}\|^2}, & otherwise. \end{cases} \tag{3.9}$$

4: Compute

$$\omega_{n+1} = (1 - \beta_n)\omega_n + \beta_n x_n, \tag{3.10}$$

5: Set $n \leftarrow n + 1$, and go to number 2.

Algorithm 3.3 Fast algorithm

1: Initial data: Choose $\beta_n \in [0, 1]$, pick $x_0 = \omega_0 \in H$ arbitrarily and set $n = 0$.

2: Given x_n, ω_n, compute

$$y_n = (1 - \beta_n)x_n + \beta_n\omega_n, \tag{3.11}$$

3: Compute

$$x_{n+1} = G(y_n - \tau_n A^*(I - T)(Ay_n)), \tag{3.12}$$

where

$$\tau_n = \begin{cases} 0, \ if \ y_n \in \Gamma, \\ \frac{\rho_n \|(I-T)Ay_n\|^2}{\|A^*(I-T)Ay_n\|^2}, \ otherwise. \end{cases} \tag{3.13}$$

4: Compute

$$\omega_{n+1} = (1 - \beta_{n+1})\omega_n + \beta_{n+1}x_{n+1}, \tag{3.14}$$

5: Set $n \leftarrow n + 1$, and go to number 2.

Remark 3.1.

(a) Our Algorithm 3.1 reduces to algorithm 6.1 studied in [14] when $\beta_n = 0$; and reduces to Algorithm 1 in [28] when $f = 0$ and $g = 0$. In addition to $f = 0$ and $g = 0$, when $\beta_n = 0$, our algorithm 3.1 reduces to the CQ algorithm studied in [16, 18, 50].

(b) Observe that $Ay_n - P_Q(I - \lambda g)(Ay_n) = 0$ implies that $y_n \in \Gamma$. In our analysis, we assume that $Ay_n - P_Q(I - \lambda g)(Ay_n) \neq 0$ after finite iterations for Algorithm 3.1 to generate infinite iterations with $Ay_n - P_Q(I - \lambda g)(Ay_n) \neq 0$ for each n.

(c) Algorithms 3.1 and 3.3 have the same computational structure. In Algorithm 3.1, $\omega_{n+1} = (1 - \beta_n)\omega_n + \beta_n x_n$, while in Algorithm 3.3, $\omega_{n+1} = (1 - \beta_{n+1})\omega_n + \beta_{n+1}x_{n+1}$.

(d) In Algorithm 3.2, the sequence of inertial factors $\theta_n \leq 0$ is permitted when $\beta_n \in [\frac{1}{2}, 1)$, and $\theta_n \geq 1$ when $\beta_n \in (0, \frac{1}{3}]$. To the best of our knowledge, these parameter ranges have not been investigated for some inertial-type algorithms in the literature.

(e) The on-line rule is defined as: choose $\theta_n \in [0, 1)$ such that $0 \leq \theta_n \leq \bar{\theta}_n$ and

$$\bar{\theta}_n = \begin{cases} \min\left\{\theta, \frac{\epsilon_n}{\|y_n - y_{n-1}\|^2}\right\}, & y_n \neq y_{n-1} \\ \theta, & otherwise, \end{cases} \tag{3.15}$$

with $\epsilon_n \in l_1$ and $\theta \geq 0$ or the assumption $\sum_{n=1}^{\infty} \theta_n \|y_n - y_{n-1}\|^2 < \infty$, which has been assumed by several authors (see, for example, [27, 51–53]) are dispensed of in our convergence analysis for Algorithm 3.2.

(f) The dynamical systems interpretation of inertial algorithms explains the effect of inertial parameters outside the classical interval $[0, 1)$ on oscillation or convergence.

(g) As shown by Alvarez & Attouch [54], the inertial term obtained by discretizing the following second order dissipative system

$$\frac{d^2x(t)}{dt^2} + \gamma(t)\frac{dx(t)}{dt} + \nabla f(x(t)) = 0,$$

where $\gamma(t) > 0$ is a damping or friction parameter. The inertial term may be negative in the overdamped regime, where it acts as a stabilizing correction that suppresses oscillation. In contrast, when the damping is small, the discretization yields an extrapolation factor greater than one, corresponding to the accelerating behavior of the underlying continuous system. Our Algorithm 3.2 follows the framework, and $\theta_n < 0$ acts as a damping correction, reducing oscillation in some settings, and $\theta_n > 1$ can produce aggressive extrapolation, which may accelerate convergence when controlled by adaptive step size rules.

3.2. Convergence analysis

In this subsection, we present convergence analyses for all three algorithms proposed above.

3.2.1. Convergence analysis of algorithm 3.1

We now give the following weak convergence results for Algorithm 3.1.

Theorem 3.2. *The sequences $\{x_n\}$ and $\{\omega_n\}$ be generated by Algorithm 3.1 with $0 \leq \beta_n \leq \limsup \beta_n < 1$, converges weakly to a solution point $x^* \in \Gamma$.*

Proof.
Let $z \in \Gamma$. This implies that $z \in VI(C, f)$ and by Equation (2.8) and Lemma 2.10(i), we have

$$\begin{aligned} \|x_{n+1} - z\|^2 &= \|G(y_n - \tau_n A^*(I-T)(Ay_n)) - z\|^2 \\ &= \|G(y_n - \tau_n A^*(I-T)(Ay_n)) - G(z)\|^2 \\ &\leq \|y_n - \tau_n A^*(I-T)(Ay_n) - z\|^2 \\ &= \|y_n - z\|^2 + \tau_n^2 \|A^*(I-T)(Ay_n)\|^2 - 2\tau_n \langle A^*(I-T)Ay_n, y_n - z\rangle. \end{aligned} \tag{3.16}$$

Since $I - T$ is firmly nonexpansive, we obtain

$$\begin{aligned} \langle A^*(I-T)Ay_n, y_n - z\rangle &= \langle (I-T)Ay_n, Ay_n - Az\rangle \\ &= \langle (I-T)Ay_n - (I-T)Az, Ay_n - Az\rangle \\ &\geq \|(I-T)Ay_n - (I-T)Az\|^2 \\ &= \|(I-T)Ay_n\|^2. \end{aligned}$$

Therefore,

$$\begin{aligned} \|x_{n+1} - z\|^2 &\leq \|y_n - z\|^2 + \tau_n^2 \|A^*(I-T)(Ay_n)\|^2 \\ &\quad -2\tau_n \langle A^*(I-T)Ay_n, y_n - z\rangle \\ &\leq \|y_n - z\|^2 + \frac{{\rho_n}^2 \|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^4} \|A^*(I-T)Ay_n\|^2, \\ &\quad -2\frac{{\rho_n}^2 \|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^2} \|(I-T)Ay_n\|^2 \\ &= \|y_n - z\|^2 + \frac{{\rho_n}^2 \|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^2} - 2\frac{{\rho_n}^2 \|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^2} \\ &= \|y_n - z\|^2 - \rho_n(2-\rho_n)\frac{\|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^2}. \end{aligned} \tag{3.17}$$

Furthermore,

$$\begin{aligned} \|y_n - z\|^2 &= \|(1-\beta_n)x_n + \beta_n \omega_n - z\|^2 \\ &= (1-\beta_n)\|x_n - z\|^2 + \beta_n \|\omega_n - z\|^2 - \beta_n(1-\beta_n)\|x_n - \omega_n\|^2, \end{aligned} \tag{3.18}$$

and

$$\begin{aligned} \|\omega_{n+1} - z\|^2 &= \|(1-\beta_n)(\omega_n - z) + \beta_n(x_n - z)\|^2 \\ &= (1-\beta_n)\|\omega_n - z\|^2 + \beta_n \|x_n - z\|^2 - \beta_n(1-\beta_n)\|x_n - \omega_n\|^2. \end{aligned} \tag{3.19}$$

Substituting Equation (3.18) into Equation (3.17), we obtain

$$\begin{aligned} \|x_{n+1} - z\|^2 &\leq (1-\beta_n)\|x_n - z\|^2 + \beta_n \|\omega_n - z\|^2 - \beta_n(1-\beta_n)\|x_n - \omega_n\|^2 \\ &\quad -\rho_n(2-\rho_n)\frac{\|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^2}. \end{aligned} \tag{3.20}$$

Combining Equation (3.19) and Equation (3.20) gives

$$\begin{aligned}
\|x_{n+1}-z\|^2+\|\omega_{n+1}-z\|^2 &\leq (1-\beta_n)\|x_n-z\|^2+\beta_n\|\omega_n-z\|^2-\beta_n(1-\beta_n)\|x_n-\omega_n\|^2\\
&\quad -\rho_n(2-\rho_n)\frac{\|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^2}+(1-\beta_n)\|\omega_n-z\|^2\\
&\quad +\beta_n\|x_n-z\|^2-\beta_n(1-\beta_n)\|x_n-\omega_n\|^2,\\
&= \|x_n-z\|^2+\|\omega_n-z\|^2-2\beta_n(1-\beta_n)\|x_n-\omega_n\|^2\\
&\quad -\rho_n(2-\rho_n)\frac{\|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^2}. \qquad (3.21)
\end{aligned}$$

Define $m_n := \|x_n+z\|^2+\|\omega_n-z\|^2,\ \forall\ n\in\mathbb{N}$.
Then Equation (3.21) becomes

$$m_{n+1}\leq m_n-2\beta_n(1-\beta_n)\|x_n-\omega_n\|^2-\rho_n(2-\rho_n)\frac{\|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^2}. \qquad (3.22)$$

Notice that $0<\beta_n<1 \implies \beta_n(1-\beta_n)>0,\ 0<\rho<2 \implies \rho_n(2-\rho_n)>0,\ \|x_n-\omega_n\|^2\geq 0$ and $\frac{\|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^2}>0$ so that Equation (3.22) reduces to

$$m_{n+1}\leq m_n.$$

Thus, $\{m_n\}$ is monotone non-increasing. Because m_n is defined as the sum of nonnegative terms, it is bounded below by 0. This implies that $\{m_n\}$ is monotone non-increasing and bounded below. Thus, $\{m_n\}$ must converge. Consequently, $\{m_n\}$ is bounded.
Observe that if $\beta_n=0$ *and* $\forall\ n\in\mathbb{N}$, then Algorithm 3.1 is reduced to the basic algorithm in [16, 18, 50], where the weak convergence is established. Thus, it suffices to consider when $0<\liminf\beta_n\leq\limsup\beta_n<1$.
Since $\{m_n\}$ converges,

$$m_n-m_{n+1}\to 0,\ \text{as } n\to\infty.$$

Therefore,

$$m_n-m_{n+1}=2\beta_n(1-\beta_n)\|x_n-\omega_n\|^2+\rho_n(2-\rho_n)\frac{\|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^2}\to 0,\ \text{as } n\to\infty.$$

Let $C_n := 2\beta_n(1-\beta_n)\|x_n-\omega_n\|^2+\rho_n(2-\rho_n)\frac{\|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^2}$.
Because C_n is a sum of nonnegative terms, the only way for the sum to tend to zero is for each term to tend to zero. In particular,

$$2\beta_n(1-\beta_n)\|x_n-\omega_n\|^2\to 0,\ \text{as } n\to\infty$$

If $0<a<\beta_n<b<1$, then $\beta_n(1-\beta_n)\geq a(1-b)>0$. Thus,

$$\beta_n(1-\beta_n)\|x_n-\omega_n\|^2\to 0 \implies \|x_n-\omega_n\|^2\to 0,\ \text{as } n\to\infty.$$

Thus, we obtain from Equation (3.22) that

$$\lim_{n\to\infty}\|x_n-\omega_n\|=0. \qquad (3.23)$$

Since $\|x_n - z\|^2 \leq m_n$ and $\|\omega_n - z\|^2 \leq m_n$ with $\{m_n\}$ bounded, we obtain $\{x_n\}$ and $\{\omega_n\}$ bounded. This implies that $\{y_n\}$ is also bounded.

Since $\{x_n\}$ is bounded, it has a weakly convergent subsequence $\{x_{n_j}\}$ such that $x_{n_j} \rightharpoonup x^*$.

We show that $x^* \in \Gamma$.
From Equation (3.2), we have the following:

$$\|y_n - x_n\| = \beta_n \|x_n - \omega_n\| \to 0 \; as \; n \to \infty.$$

Consequently, $\{y_{n_j}\} \subset \{y_n\}$ and $y_{n_j} \rightharpoonup x^* \; as \; j \to \infty$.

We also obtain from Equation (3.21) that

$$\lim_{n\to\infty} \frac{\|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^2} = 0.$$

Since $z \in \Gamma$, we have $A^*(I-T)Az = 0$.
Hence,

$$\|A^*(I-T)Ay_n - A^*(I-T)Az\| \leq \|A\|^2 \|y_n - z\|.$$

Therefore, $\{\|A^*(I-T)Ay_n\}$ is bounded.

From Equation (3.22), we get the following result

$$\lim_{n\to\infty} \|(I-T)Ay_n\| = 0. \tag{3.24}$$

By the assumption for λ and g, we get from Lemma 2.10 (i) that T is nonexpansive. Applying the demiclosedness of $I - T$ at 0 to (3.24), we have

$$T(Ax^*) = Ax^*. \tag{3.25}$$

This means that $Ax^* \in VI(Q, g)$.

Denote

$$r_n := y_n - \tau_n A^*(I-T)(Ay_n). \tag{3.26}$$

Then

$$r_{n_j} = y_{n_j} - \tau_n A^*(I-T)(Ay_{n_j}). \tag{3.27}$$

Since $y_{n_j} \rightharpoonup x^*$, Equations (3.27) and (3.24) imply that $r_{n_j} \to x^*$.

We now show that $x^* \in VI(C, f)$. Assume to the contrary that $x^* \notin VI(C, f)$, that is, $Gx^* \neq x^*$.
By the assumptions on λ and f, we get from Lemma 2.10(i) that G is nonexpansive and, thus, $I - G$ is demiclosed at 0. Therefore, the contrary assumption must lead to

$$\lim_{j\to\infty} \|G(r_{n_j}) - r_{n_j}\| \neq 0. \tag{3.28}$$

Therefore, there exists an $\epsilon > 0$ and a subsequence $\{r_{n_{j_s}}\}$ of $\{r_{n_j}\}$ such that

$$\|G(r_{n_{j_s}}) - r_{n_{j_s}}\| > \epsilon \; for \; all \; s \geq 0. \tag{3.29}$$

Condition (3.1) justifies the use of Lemma 2.10 by giving Equation (2.9). Therefore, inequality (2.11) now gives, for all $s \geq 0$,

$$\begin{aligned} \|G(r_{n_{j_s}}) - G(z)\|^2 &= \|G(r_{n_{j_s}}) - z\|^2 \\ &\leq \|r_{n_{j_s}} - z\|^2 - \|G(r_{n_{j_s}}) - r_{n_{j_s}}\|^2, \\ &< \|r_{n_{j_s}} - z\|^2 - \epsilon^2. \end{aligned} \tag{3.30}$$

By arguments similar to those in the proof of Lemma 1.4 [14], we have

$$\begin{aligned} \|r_n - z\| &= \|(y_n - \tau_n A^*(I - T)(Ay_n)) - z\|, \\ &\leq \|x_n - z\|. \end{aligned} \tag{3.31}$$

Since G is nonexpansive, it follows that

$$\|x_{n+1} - z\| = \|G(r_n) - z\| \leq \|r_n - z\|. \tag{3.32}$$

combining Equation (3.31) and Equation (3.32), we have

$$\|x_{n+1} - z\| \leq \|r_n - z\| \leq \|x_n - z\|. \tag{3.33}$$

By lemma 2.13, this implies that the sequence $\{x_1, r_1, x_2, r_2, \cdots\}$ is Fej*é*r - monotone with respect to Γ.

Since $x_{n_{j_{s+1}}} = G(r_{n_{j_s}})$, we obtain

$$\|r_{n_{j_{s+1}}} - z\|^2 \leq \|r_{n_{j_s}} - z\|^2. \tag{3.34}$$

Hence, $\{r_{n_{j_s}}\}_{s=0}^{\infty}$ is also Fej*é*r - monotone with respect to Γ. Now, Equation (3.30) and Equation (3.33) imply that

$$\|r_{n_{j_{s+1}}} - z\|^2 < \|r_{n_{j_s}} - z\|^2 - \epsilon^2 \; for \; all \; s \geq 0. \tag{3.35}$$

Therefore, Equation (3.35) cannot be true for infinitely many vectors $r_{n_{j_s}}$.

Hence, $x^* \in VI(C, f)$ and thus, $x^* \in \Gamma$.

Now, define

$$d_n := \|\omega_n - x_n\|^2 + 2\langle \omega_n - x_n, x_n - z\rangle,$$

where $z \in \Gamma$. Since $\omega_n - x_n \to 0$ as $n \to \infty$ and $\{x_n\}$ is bounded, we have $d_n \to 0$ as $n \to \infty$. Observe also that

$$\|\omega_n - z\|^2 = \|\omega_n - x_n\|^2 + 2\langle \omega_n - x_n, x_n - z\rangle + \|x_n - z\|^2.$$

Therefore, we have

$$m_n - d_n = 2\|x_n - z\|^2.$$

This implies that

$$\|x_n - z\|^2 = \frac{1}{2}(m_n - d_n).$$

Since $\lim_{n\to\infty} m_n$ exists, we have that $\lim_{n\to\infty} \|x_n - z\|^2$ exists. By Lemma 2.12, we conclude that $\{x_n\}$ converges weakly to a point in Γ. Consequently, $\{\omega_n\}$ and $\{y_n\}$ converge weakly to a point in Γ. □

3.2.2. Convergence analysis of algorithm 3.2

We now give the following weak convergence for Algorithm 3.2.

Theorem 3.3. *The sequences* $\{x_n\}$ *and* $\{\omega_n\}$ *generated by Algorithm 3.2 with* $0 \leq \beta_n \leq \limsup \beta_n < 1$ *weakly converge to a solution point* $x^* \in \Gamma$.

Proof. : It suffices to show that Algorithm 3.2 is a special case of Algorithm 3.1.
Suppose that $v_n := \omega_n - x_n$. Then, Equation (3.6) becomes

$$\begin{aligned} y_n &= \omega_{n+1} + \Theta_n(\omega_{n+1} - \omega_n), \\ &= x_n + v_n + (1 + \Theta_n)(\omega_{n+1} - \omega_n), \end{aligned} \tag{3.36}$$

where, from Equation (3.6), we have

$$\begin{aligned} \omega_{n+1} - \omega_n &= -\beta_n(\omega_n - x_n) \\ &= -\beta_n v_n. \end{aligned}$$

Note that for $\Theta_n = \frac{1-2\beta_n}{\beta_n}$, we have

$$\begin{aligned} y_n &= x_n + v_n + (1 + \Theta_n)(-\beta_n v_n) \\ &= x_n + v_n - \beta_n v_n - \theta_n \beta_n v_n \\ &= x_n + (1 - \beta_n - \Theta_n \beta_n) v_n \\ &= x_n + \left[1 - \beta_n - \left(\frac{1 - 2\beta_n}{\beta_n}\right)\beta_n\right](\omega_n - x_n) \\ &= x_n + [1 - \beta_n - 1 + 2\beta_n](\omega_n - x_n) \\ &= x_n + \beta_n(\omega_n - x_n). \end{aligned}$$

Therefore, we obtain the first equation in Equation (3.2). Thus, we obtain the desired result from Theorem 3.2. □

3.2.3. Convergence analysis of algorithm 3.3

We now give the following weak convergence for Algorithm 3.3.

Theorem 3.4. *Sequences* $\{x_n\}$ *and* $\{\omega_n\}$ *generated by Algorithm 3.3 converge weakly to a point in* Γ *with* $0 \leq \beta_n \leq \beta_{n+1} \leq 1$, *and assumption 3.1 is satisfied.*

Proof. : Let $z \in \Gamma$. Then

$$\begin{aligned} \|\omega_{n+1} - z\|^2 &= \|(1 - \beta_{n+1})(\omega_n - z) + \beta_{n+1}(x_{n+1} - z)\|^2 \\ &= (1 - \beta_{n+1})\|\omega_n - z\|^2 + \beta_{n+1}\|x_{n+1} - z\|^2 - \beta_{n+1}(1 - \beta_{n+1})\|x_{n+1} - \omega_n\|^2. \end{aligned} \tag{3.37}$$

Summing Equation (3.37) and Equation (3.20), we have

$$\|x_{n+1} - z\|^2 + \|\omega_{n+1} - z\|^2 \leq (1 - \beta_n)\|x_n - z\|^2 + \beta_n\|\omega_n - z\|^2 - \beta_n(1 - \beta_n)\|x_n - \omega_n\|^2$$

$$-\rho_n(2-\rho_n)\frac{\|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^2} + (1-\beta_{n+1})\|\omega_n - z\|^2$$
$$+\beta_{n+1}\|x_{n+1}-z\|^2 - \beta_{n+1}(1-\beta_{n+1})\|x_{n+1}-\omega_n\|^2.$$

Noting that $\beta_n \leq \beta_{n+1}$, we have

$$\begin{aligned}
\|x_{n+1}-z\|^2 &- \beta_{n+1}\|x_{n+1}-z\|^2 + \|\omega_{n+1}-z\|^2 + \beta_{n+1}(1-\beta_{n+1})\|x_{n+1}-\omega_n\|^2 \\
&\leq (1-\beta_n)\|x_n-z\|^2 + (\beta_n+1-\beta_{n+1})\|\omega_n-z\|^2 - \beta_n(1-\beta_n)\|x_n-\omega_n\|^2 \\
&\quad -\rho_n(2-\rho_n)\frac{\|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^2} - \beta_n(1-\beta_n)\|x_n-\omega_{n-1}\|^2 \\
&\quad +\beta_n(1-\beta_n)\|x_n-\omega_{n-1}\|^2 \\
&\leq (1-\beta_n)\|x_n-z\|^2 + \|\omega_n-z\|^2 + \beta_n(1-\beta_n)\|x_n-\omega_{n-1}\|^2 \\
&\quad -\beta_n(1-\beta_n)\|x_n-\omega_{n-1}\|^2 - \rho_n(2-\rho_n)\frac{\|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^2} \\
&\quad -\beta_n(1-\beta_n)\|x_n-\omega_n\|^2. \qquad (3.38)
\end{aligned}$$

Define

$$c_n := (1-\beta_n)\|x_n-z\|^2 + \|\omega_n-z\|^2 + \beta_n(1-\beta_n)\|x_n-\omega_{n-1}\|^2. \qquad (3.39)$$

Thus, from (3.38), we obtain

$$c_{n+1} \leq c_n - \beta_n(1-\beta_n)\|x_n-\omega_{n-1}\|^2 - \rho_n(2-\rho_n)\frac{\|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^2} - \beta_n(1-\beta_n)\|x_n-\omega_n\|^2. \qquad (3.40)$$

Therefore, following the similar proof approach for $\{m_n\}$ in theorem 3.2 gives that $\{c_n\}$ is monotone non-increasing and bounded below. Consequently, $\{c_n\}$ is convergent. Moreover, $\{c_n\}$ is bounded.

Consider these two cases

Case 1:
Suppose $0 < a < \beta_n \leq \beta_{n+1} \leq b < 1$. Then, from (3.40), we obtain

$$c_{n+1} \leq c_n - \beta_n(1-\beta_n)\|x_n-\omega_{n-1}\|^2.$$

A similar proof approach as in theorem 3.2 gives

$$\lim_{n\to\infty}\|x_n-\omega_{n-1}\|^2 = 0.$$

Also,

$$\lim_{n\to\infty}\rho_n(2-\rho_n)\frac{\|(I-T)Ay_n\|^4}{\|A^*(I-T)Ay_n\|^2} = 0,$$

and similarly

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

Now,

$$\|\omega_n-\omega_{n-1}\| \leq \|x_n-\omega_{n-1}\| + \|x_n-\omega_n\| \to 0,\ as\ n\to\infty.$$

Also, from

$$\begin{aligned} y_n &= (1-\beta_n)x_n + \beta_n\omega_n \\ &= x_n - \beta_n x_n + \beta_n\omega_n \\ &= x_n + \beta_n(\omega_n - x_n). \end{aligned}$$

We get

$$y_n - x_n = \beta_n(\omega_n - x_n) \to 0 \; as \; n \to \infty.$$

Since

$$c_n := (1-\beta_n)\|x_n - z\|^2 + \|\omega_n - z\|^2 + \beta_n(1-\beta_n)\|x_n - \omega_{n-1}\|^2.$$

We have

$$(1-\beta_n)\|x_n - z\|^2 \leq c_n$$

and $\{c_n\}$ is bounded. Thus, we conclude that $\{x_n\}$ is bounded.

Also from

$$\begin{aligned} \omega_{n+1} &= (1-\beta_{n+1})\omega_n + \beta_{n+1}x_{n+1} \\ &= \omega_n - \beta_{n+1}\omega_n + \beta_{n+1}x_{n+1} \\ &= \omega_n + \beta_{n+1}(x_{n+1} - \omega_n). \end{aligned}$$

Then, We get

$$\omega_{n+1} - \omega_n = \beta_{n+1}(x_{n+1} - \omega_n) \to \infty \; as \; n \to \infty.$$

Since

$$c_n := (1-\beta_n)\|x_n - z\|^2 + \|\omega_n - z\|^2 + \beta_n(1-\beta_n)\|x_n - \omega_{n-1}\|^2,$$

$$\|\omega_n - z\|^2 \leq c_n$$

and $\{c_n\}$ is bounded. This concludes that $\{\omega_n\}$ is also bounded.

Let $x^* \in H$ be a weak cluster point of $\{x_n\}$. Then there exists $\{x_{n_j}\} \subset \{x_n\}$ such that $x_{n_j} \rightharpoonup x^*, \; j \to \infty$. Similar to the proof of Theorem 3.2, we have $x^* \in \Gamma$.

Case II:
Define

$$d_n^* := \|\omega_n - x_n\|^2 + 2\langle \omega_n - x_n, x_n - z\rangle$$

and

$$e_n^* := -\beta_n(1-\beta_n)\|x_n - \omega_{n-1}\|^2.$$

Since $\{x_n\}$ is bounded and $\lim_{n\to\infty} \|x_n - \omega_n\| = 0$, we have $\lim_{n\to\infty} d_n^* = 0$ and $e_n^* \to 0, \; as \; n \to \infty$.

Observe that

$$\|\omega_n - z\|^2 = \|(\omega_n - x_n) + (x_n - z)\|^2$$

$$= \|\omega_n - x_n\|^2 + 2\langle \omega_n - x_n, x_n - z\rangle + \|x_n - z\|^2,$$

and

$$\begin{aligned} d_n* &= \|\omega_n - x_n\|^2 + 2\langle \omega_n - x_n, x_n - z\rangle \\ &= \|\omega_n - z\|^2 - \|x_n - z\|^2. \end{aligned}$$

Then, we obtain

$$\begin{aligned} c_n - d_n^* + e_n^* &= (1-\beta_n)\|x_n - z\|^* + \|\omega_n - z\|^2 + \beta_n(1-\beta_n)\|x_n - \omega_{n-1}\|^2 \\ &\quad -(\|\omega_n - z\|^2 - \|x_n - z\|^2) - \beta_n(1-\beta_n)\|x_n - \omega_{n-1}\|^2 \\ &= (1-\beta_n)\|x_n - z\|^2 + \|x_n - z\|^2 \\ &= (2-\beta_n)\|x_n - z\|^2. \end{aligned}$$

Since $\lim_{n\to\infty} d_n^*$ and $\lim_{n\to\infty} \beta_n$ exist, then $\lim_{n\to\infty} \|x_n - z\|$ exists.

Hence, by Lemma 2.12, $\{x_n\}$ converges weakly to a point in Γ. Similarly, $\{\omega_n\}$ and $\{y_n\}$ converge weakly, respectively, to a point in Γ. This completes the proof. □

Remark 3.5. If we set $\beta_n \in \{0, 1\}$ and $\tau_n \equiv \tau \in (0, \frac{1}{L})$ for all $n \in \mathbb{N}$, where L is the spectral radius of A^*A, then Algorithm 3.3 reduces to the form $x_{n+1} = G(x_n - \tau A^*(I - T)(Ax_n))$ as in [14]. It has been shown that every weak cluster point of $\{x_n\}$ lies in Γ.

4. Numerical examples

In this section, we examine the performance of Algorithm 3.1, Algorithm 3.2, and Algorithm 3.3 numerically using a variety of test cases. We aim to solve the SVIP, Equation (1.1), and Equation (1.2) to assess the efficiency and reliability of these algorithms. Additionally, we compare these schemes with the methods proposed by Censor [14] [Algorithm 6.1], Cholamjiak et al. Alg. 1 – 3 and Xu et al. Alg.3.3. All numerical calculations are executed using MATLAB2024a on a personal HP-Laptop equipped with a 12th Gen Intel(R) Core(TM) i7-1255U processor, running at 1700 MHz, with 10 cores, 12 logical processors, and 16.00GB of RAM. In all the examples, we use

$$L = \max\Big\{|eigenvalue\ of\ A^*A|\Big\}.$$

Example 4.1. We consider the minimum-norm solution of the SFP discussed in [7, 14, 26]. The mathematical model can be described as an optimization problem:

$$\min_x \Big\{\frac{1}{2}\|x\|^2 \mid x \in C \text{ and } Ax \in Q\Big\}. \tag{4.1}$$

This problem can be reformulated into SVIP, Equation (1.1) and Equation (1.2) with $f(x) = x$ and $g(y) = 0$.
The data for solving (4.1) is generated as follows:

Let $H = \mathbb{R}^m$. $H_1 = H_2 = \mathbb{R}^m$ and linear operator $A \in \mathbb{R}^{m\times n}$ be generated with independent Gaussian components distributed in the interval $(0, 1)$ by MATLAB script $A = rand(m, n)$ just as defined in [40], and also normalize each column of A with the unit norm. Since $f(x) = x$ is Lipschitz continuous and $g(y) = 0$, $L = \|A\|$, $\alpha_1 = 1$, and $\alpha_2 > 0$. In particular, set $\alpha_2 = 1$. The convex set C is the ball centered at the origin with a radius of $r = 30$, that is, $C := B_{30}(0) = (-30, 30)$. In addition, the box $Q := \{y \in \mathbb{R}^m \mid s \le y \le t\}$, $\rho_n \in (0, 2)$.

Generate a point x^*, whose entries are uniformly distributed in $(-30, 30)$ and obtain the vector $y^* = Ax^*$. Let all entries of s and t be the smallest and largest components of y^*, respectively. The corresponding MATLAB Scripts are as follows: $x^* = 60^* rand(n, 1) - 30$, $y^* = Ax^*$, $s = \min(y*)$, $t = \max(y*)$.

Consider the following different initial data for the numerical experiment :
case 1: $m = 30$ & $n = 60$.
case 2: $m = 10$ & $n = 40$.
case 3: $m = 20$ & $n = 50$.
case 4: $m = 30$ & $n = 80$.

Choose $x_0 = \omega_0 = 60 * rand(n, 1) - 30$, and for the Xu et al Alg.3.3, choose the fixed data, $\omega = x_0$. Set the stopping criterion as follows:

$$D_n := \|x_{n+1} - x_n\| \le tol,$$

where $tol = 10^{-4}$.

Table 1. Numerical performance of different λ for proposed algorithm 3.1.

	$\lambda = 0.1$		$\lambda = 0.01$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.1	2	1.9500×10^{-5}	135	9.6550×10^{-4}
	$\lambda = 0.001$		$\lambda = 0.0001$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.1	87	9.6130×10^{-4}	2	1.8200×10^{-5}

Table 2. Numerical performance of different λ for Proposed Algorithm 3.2.

	$\lambda = 5.0 \times 10^{-15}$		$\lambda = 1.0 \times 10^{-15}$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.2	2	1.8600×10^{-5}	266	9.5970×10^{-4}
	$\lambda = 1.0 \times 10^{-10}$		$\lambda = 1.0 \times 10^{-5}$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.2	2	1.1700×10^{-5}	2	1.3100×10^{-5}

Table 3. Numerical performance of different λ for proposed algorithm 3.3.

	$\lambda = 1.0 \times 10^{-15}$		$\lambda = 1.0 \times 10^{-10}$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.3	11	1.2574×10^{-2}	11	1.9168×10^{-3}
	$\lambda = 1.0 \times 10^{-5}$		$\lambda = 1.0 \times 10^{-2}$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.3	11	5.9320×10^{-4}	813	1.9275×10^{-2}

Table 4. Example 4.1: comparison results of proposed alg. 3.1, Cholamjiak et al. Alg. 1, Censor et al. Alg., and Xu et al. Alg. 3.3 at different cases of initial points.

	Proposed 3.1		**Cholamjiak et al. Alg. 1**		**Censor et al. Alg.**		**Xu et al. Alg. 3.3**	
	Iter.	CPU (Time)	Iter.	CPU (Time)	Iter.	CPU (Time)	Iter.	CPU (Time)
Case 1	2	2.3900×10^{-5}	23	3.4870×10^{-4}	5	4.4790×10^{-4}	1451	6.6445×10^{-3}
Case 2	2	4.16×10^{-5}	91	1.2769×10^{-3}	5	1.841×10^{-4}	1302	3.6373×10^{-3}
Case 3	2	4.8000×10^{-5}	84	4.0850×10^{-4}	5	5.1100×10^{-5}	1283	1.9700×10^{-2}
Case 4	2	3.9000×10^{-5}	47	5.7200×10^{-4}	5	1.4400×10^{-4}	1533	1.3243×10^{-2}

Table 5. Example 4.1: comparison results of proposed Alg. 3.2, Cholamjiak et al. Alg. 2, and Xu et al. Alg. 3.3 at different cases of initial points.

	Proposed Alg. 3.2		**Cholamjiak et al. Alg. 2**		**Xu et al. Alg. 3.3**	
	Iter.	CPU (Time)	Iter.	CPU (Time)	Iter.	CPU (Time)
Case 1	2	2.5000×10^{-5}	30	8.6230×10^{-4}	1371	4.4024×10^{-3}
Case 2	2	2.4900×10^{-5}	28	2.1030×10^{-4}	1301	4.0720×10^{-3}
Case 3	2	2.5200×10^{-5}	30	2.7340×10^{-4}	1446	5.1142×10^{-3}
Case 4	2	1.7400×10^{-5}	30	8.9720×10^{-4}	1612	5.4536×10^{-3}

Table 6. Example 4.1: comparison results of proposed Alg. 3.3, Cholamjiak et al. Alg. 3, and Xu et al. Alg. 3.3 at different cases of initial points.

	Proposed Alg. 3.3		**Cholamjiak et al. Alg. 3**		**Xu et al. Alg. 3.3**	
	Iter.	CPU (Time)	Iter.	CPU (Time)	Iter.	CPU (Time)
Case 1	11	1.7277×10^{-3}	795	2.0715×10^{-2}	1451	9.2328×10^{-3}
Case 2	11	1.2270×10^{-4}	848	5.2581×10^{-3}	1324	3.8343×10^{-3}
Case 3	11	3.2760×10^{-4}	710	5.0220×10^{-3}	1221	3.2792×10^{-3}
Case 4	11	8.9760×10^{-4}	965	4.2912×10^{-2}	1579	6.1740×10^{-3}

Table 7. Parameters of each algorithm compared for Example 4.1.

Algorithm	**Parameters**		
Proposed Alg. 3.1	$\lambda = 1.0 \times 10^{-4}$	$\rho_n = 1.5$	–
Cholamjiak et al. Alg. 1	$\tau_0 = 0.99$	$\rho_n = 1.5$	–
Censor et al. Alg.	$L = \max \lvert\text{eigenvalue}(A^T A)\rvert$	$\gamma = 0.9999 \times \frac{1}{L}$	$\lambda = 1.0 \times 10^{-4}$
Proposed Alg. 3.2	$\lambda = 1.0 \times 10^{-5}$ $\beta_{-1} = 0.9$	$\rho_n = 1.9$	$\beta_0 = 0.9$
Cholamjiak et al. Alg. 2	$\tau_{-1} = 0.999$ $\beta_{-1} = 0.9$	$\rho_n = 1.9$	$\beta_0 = 0.9$
Proposed Alg. 3.3	$\lambda = 1.0 \times 10^{-5}$	$\rho_n = 1.5$	$\beta_{n+1} = 0.5$
Cholamjiak et al. Alg. 3	$\tau_0 = 0.99$	$\rho_n = 1.5$	$\beta_{n+1} = 0.5$

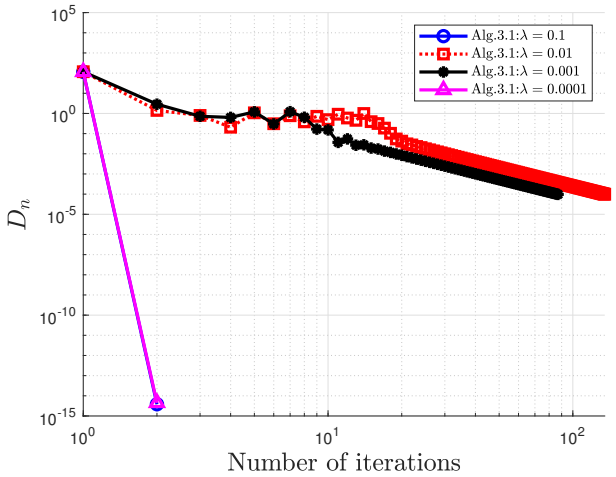

Figure 1. Example 4.1: Different values of λ for Algorithm 3.1.

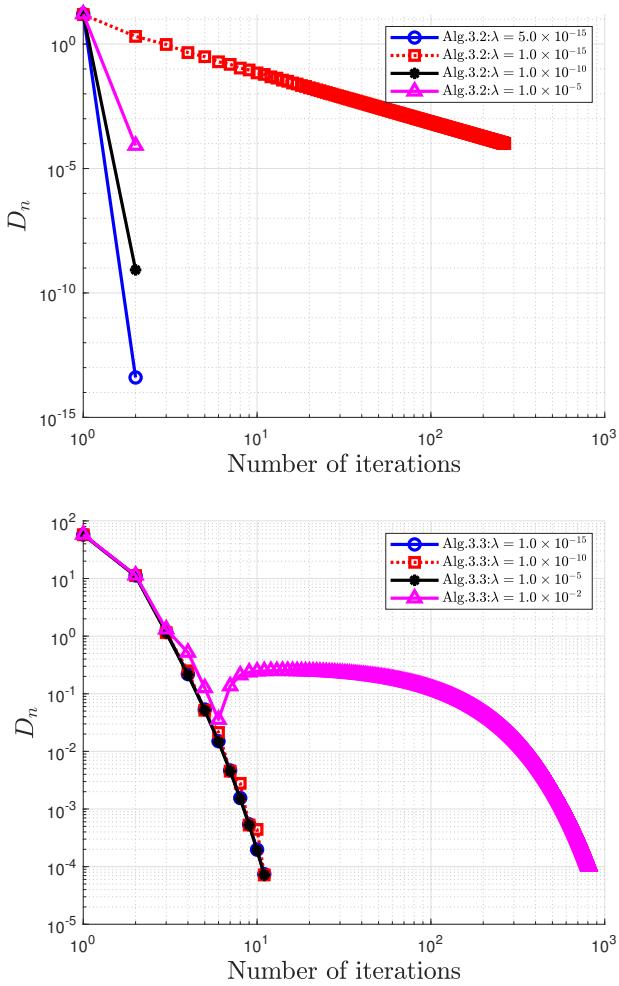

Figure 2. Example 4.1: Different values of λ for Algorithms 3.2 and 3.3.

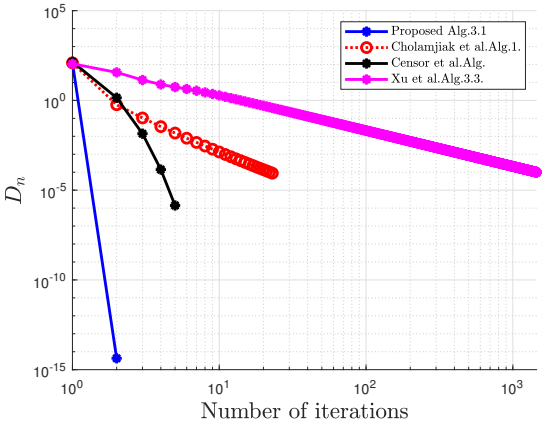

Figure 3. Example 4.1 case 1: category I.

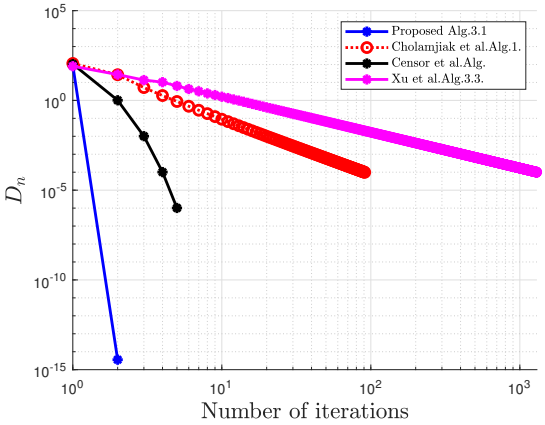

Figure 4. Example 4.1 case 2: category I.

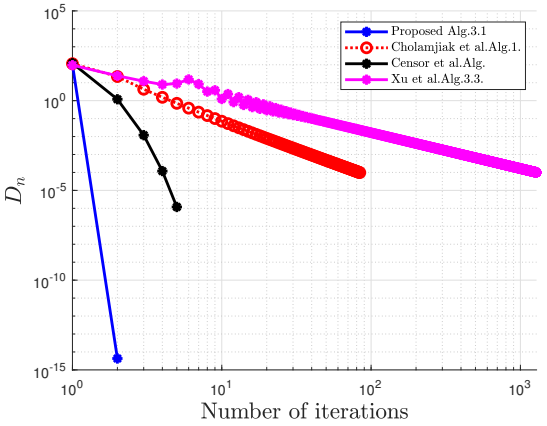

Figure 5. Example 4.1 case 3: category I.

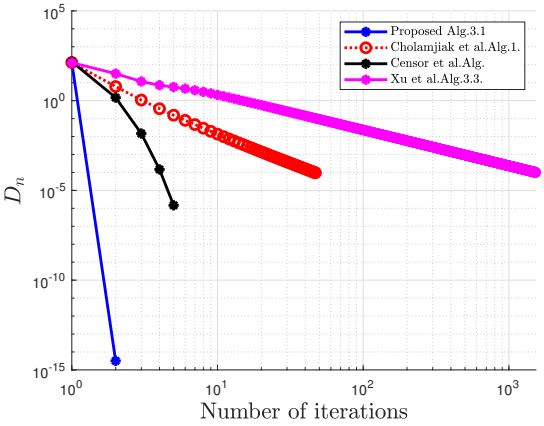

Figure 6. Example 4.1 case 4: category I.

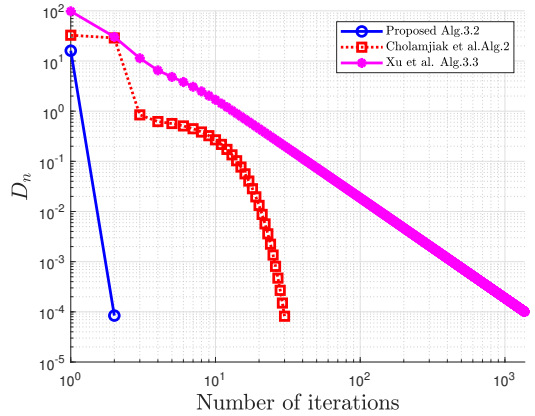

Figure 7. Example 4.1 case 1: category II.

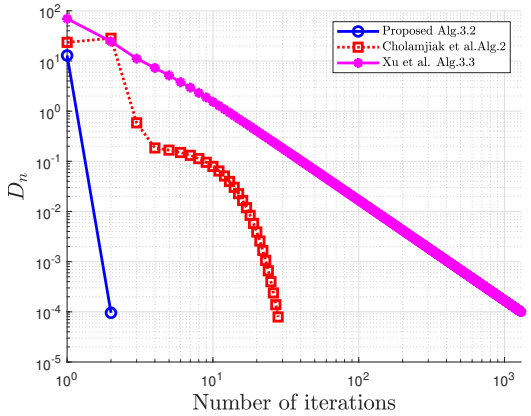

Figure 8. Example 4.1 case 2: category II.

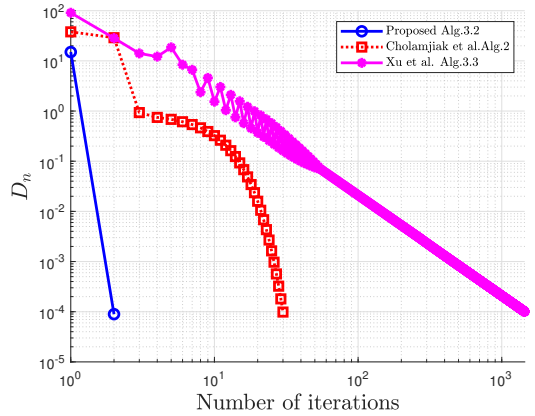

Figure 9. Example 4.1 case 3: category II.

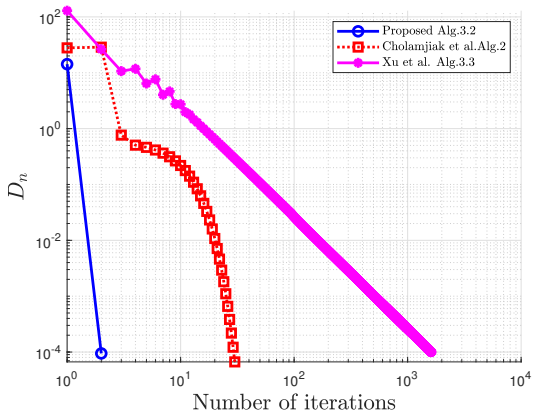

Figure 10. Example 4.1 case 4: category II.

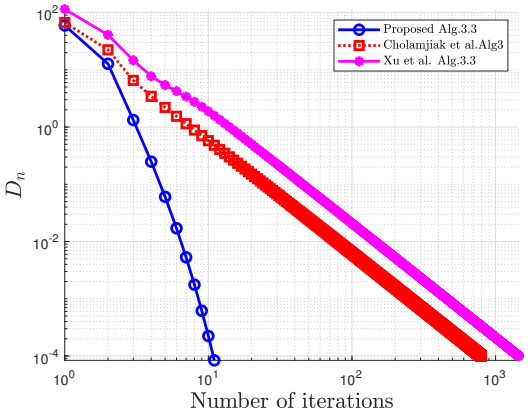

Figure 11. Example 4.1 case 1: category III.

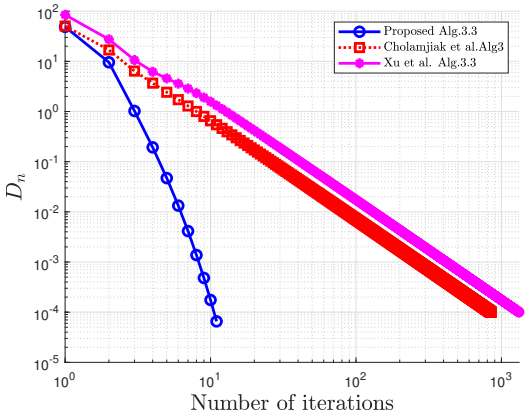

Figure 12. Example 4.1 case 2: category III.

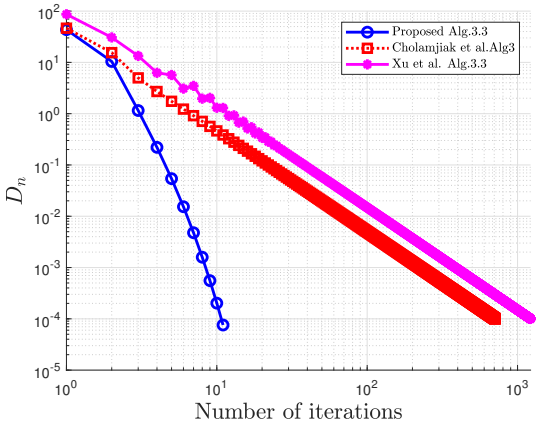

Figure 13. Example 4.1 case 3: category III.

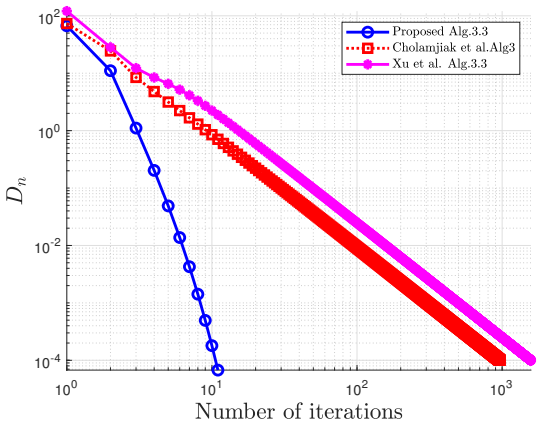

Figure 14. Example 4.1 case 4: category III.

Example 4.2. We consider this example as discussed in Example 5.3 and Example 4.2 in [55, 56]. Let $H_1 = H_2 = L_2([0,1])$ with the inner product

$$\langle x, y\rangle = \int_0^1 x(t)y(t)dt, \quad \forall x, y \in L_2([0,1]),$$

and the induced norm

$$\|x\| := \Big(\int_0^1 |x(t)|^2 dt\Big)^{\frac{1}{2}}, \quad x, y \in L_2([0,1]),$$

Let h be an arbitrary fixed element in $C([0,1])$ (the space of all real-valued continuous functions on $[0,1)$. Let $A : H_1 \to H_2$, a bounded linear operator be defined by

$$A(x) = kx, \quad \forall x \in H \quad and \quad k \in [0,1].$$

Let $C = Q = \Big\{x \in L_2([0,1]) : \|hx\| \leq 1\Big\}$. Specifically, for all $t \in [0,1]$, we define $h(t) = e^{-t}$ for C and $h(t) = e^{-5t}$ for Q. Then C and Q are nonempty, closed, convex subsets of $L_2([0,1])$. Let

$f : L_2([0,1]) \to L_2([0,1])$ be defined by

$$f(x(t)) = \max\{0, x(t)\}, \quad \forall x \in L_2([0,1])$$

and let $g : L_2([0,1]) \to L_2([0,1])$ be defined by

$$g(x)(t) = \int_0^t x(s)ds, \quad \forall x \in L_2([0,1]),\ t \in [0,1],$$

with $\alpha_1 = 1$ and $\alpha_2 = 0.4$.
Then f and g are monotone and Lipschitz continuous.
Since $0 \in C$ and $0 \in Q$, it follows that $0 \in C \cap Q$. Moreover, $A0 = 0$, so $A0 \in Q$. In addition, $f(0)(t) = 0$ for all $t \in [0,1]$, and therefore

$$\langle f(0), x - 0\rangle = \langle 0, x\rangle = 0 \quad \text{for all } x \in C.$$

Similarly, $g(0)(t) = 0$ for all $t \in [0,1]$, and hence

$$\langle g(0), y - 0\rangle = \langle 0, y\rangle = 0 \geq 0 \quad \text{for all } y \in Q.$$

Therefore, 0 satisfies both variational inequalities and the coupling condition, and thus

$$0 \in \Gamma.$$

Consequently, $\Gamma \neq \emptyset$.

Consider the following initial values.

Case 1: $x_0 = \omega_0 = 6t^2$.
Case 2: $x_0 = \omega_0 = 150\pi t \sin t$.
Case 3: $x_0 = \omega_0 = 60e^{2t}$.
Case 4: $x_0 = \omega_0 = 133t^2$.

For the Xu et al. Alg.3.3, the fixed data, $\omega = 7t^2$

Set the stopping criterion as follows:

$$D_n := \|x_{n+1} - x_n\| \leq tol = 10^{-5}.$$

Table 8. Parameters of each algorithm compared for Example 4.2.

Algorithm	**Parameters**		
Proposed Alg. 3.1	$\lambda = 1.0 \times 10^{-10}$	$\rho_n = 1.5$	–
Cholamjiak et al. Alg. 1	$\tau_0 = 0.99$	$\rho_n = 1.5$	–
Censor et al. Alg.	$\lambda = 1.0 \times 10^{-10}$	$L = 0.4$	$\gamma = 0.9999 \times \frac{1}{L}$
Proposed Alg. 3.2	$\lambda = 1.0 \times 10^{-10}$ $\beta_{-1} = 0.3050$	$\rho_n = 1.5$	$\beta_0 = 0.999$
Cholamjiak et al. Alg. 2	$\tau_{-1} = 0.99$ $\beta_{-1} = 0.3050$	$\rho_n = 1.5$	$\beta_0 = 0.999$
Proposed Alg. 3.3	$\lambda = 1.0 \times 10^{-10}$	$\rho_n = 1.5$	$\beta_{n+1} = 0.5$
Cholamjiak et al. Alg. 3	$\tau_0 = 0.99$	$\rho_n = 1.5$	$\beta_{n+1} = 0.5$

Table 9. Example 4.2: Numerical performance of different λ for Algorithm 3.1.

	$\lambda = 1.0\times10^{-15}$		$\lambda = 1.0\times10^{-10}$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.1	2	6.515×10^{-4}	2	4.81×10^{-5}
	$\lambda = 1.0 \times 10^{-5}$		$\lambda = 1.0 \times 10^{-2}$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.1	2	4.18×10^{-5}	3518	8.817×10^{-2}

Table 10. Example 4.2: Numerical performance of different λ for Algorithm 3.2.

	$\lambda = 1.0\times10^{-15}$		$\lambda = 1.0\times10^{-10}$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.2	21	8.7003×10^{-3}	21	3.0273×10^{-3}
	$\lambda = 1.0 \times 10^{-5}$		$\lambda = 1.0 \times 10^{-2}$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.2	500	1.3322×10^{-2}	500	1.2502×10^{-2}

Table 11. Example 4.2: Numerical performance of different λ for Algorithm 3.3.

	$\lambda = 5.0\times10^{-15}$		$\lambda = 1.0\times10^{-15}$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.3	2	7.2100×10^{-5}	2	4.0600×10^{-5}
	$\lambda = 1.0\times10^{-10}$		$\lambda = 1.0 \times 10^{-5}$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.3	2	9.2700×10^{-5}	241707	1.292508×10^{2}

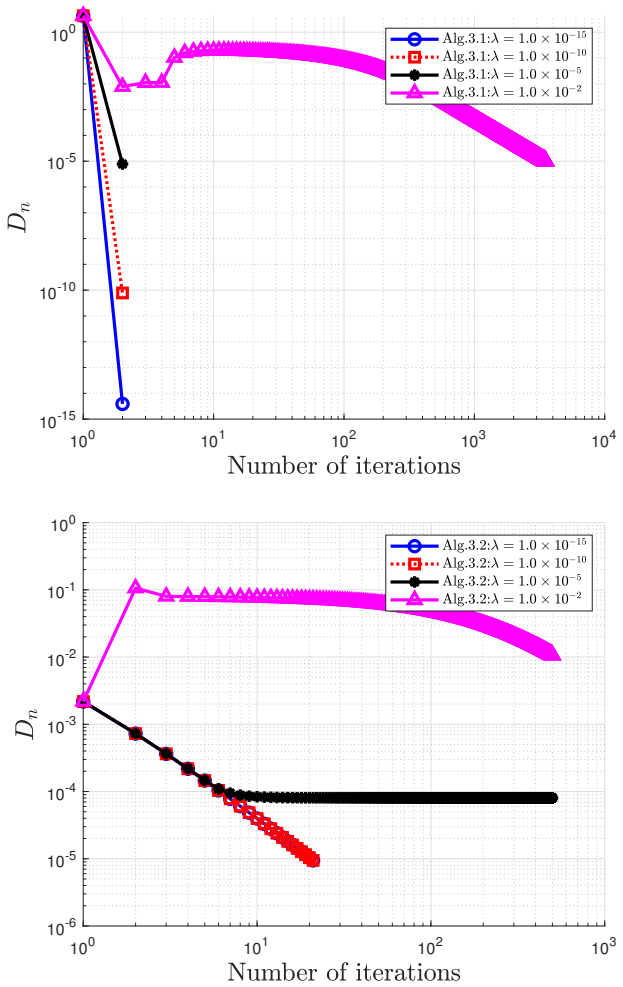

Figure 15. Example 4.2: Different values of λ for Algorithms 3.1 and 3.2.

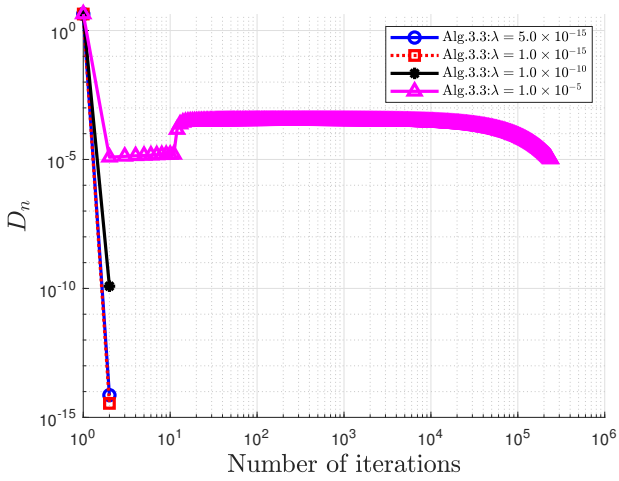

Figure 16. Example 4.2: Different values of λ for Algorithm 3.3.

Table 12. Example 4.2: Comparison results of proposed Alg. 3.1, Cholamjiak et al. Alg. 1, Censor et al. Alg., and Xu et al. Alg. 3.3 for different initial points.

	Proposed 3.1		**Cholamjiak et al. Alg. 1**		**Censor et al. Alg.**		**Xu et al. Alg. 3.3**	
	Iter.	CPU (Time)	Iter.	CPU (Time)	Iter.	CPU (Time)	Iter.	CPU (Time)
Case 1	2	5.2400×10^{-5}	2	1.2380×10^{-3}	2	1.3070×10^{-4}	1333	1.6533×10^{-2}
Case 2	2	5.6100×10^{-5}	2	8.8180×10^{-4}	2	6.5100×10^{-5}	1333	1.7880×10^{-2}
Case 3	2	4.1700×10^{-5}	2	6.7270×10^{-4}	2	4.7100×10^{-5}	1333	1.8621×10^{-2}
Case 4	2	4.1800×10^{-5}	2	6.2901×10^{-3}	2	6.5900×10^{-5}	1333	3.2920×10^{-2}

Table 13. Example 4.2: Comparison results of proposed Alg. 3.2, Cholamjiak et al. Alg. 2, and Xu et al. Alg. 3.3 for different initial points.

	Proposed Alg. 3.2		**Cholamjiak et al. Alg. 2**		**Xu et al. Alg. 3.3**	
	Iter.	CPU (Time)	Iter.	CPU (Time)	Iter.	CPU (Time)
Case 1	21	6.1350×10^{-4}	21	1.3042×10^{-3}	1333	2.0027×10^{-2}
Case 2	431	1.3241×10^{-2}	431	1.5951×10^{-2}	1333	1.6329×10^{-2}
Case 3	468	1.4206×10^{-2}	468	1.1447×10^{-2}	1333	1.7120×10^{-2}
Case 4	240	5.2959×10^{-3}	240	5.3832×10^{-3}	1333	1.7266×10^{-2}

Table 14. Example 4.2: Comparison results of proposed Alg. 3.3, Cholamjiak et al. Alg.3, and Xu et al. Alg.3.3 for different initial points.

	Proposed Alg. 3.3		**Cholamjiak et al. Alg.3**		**Xu et al. Alg.3.3**	
	Iter.	CPU (Time)	Iter.	CPU (Time)	Iter.	CPU (Time)
Case 1	2	4.3700×10^{-5}	2	4.5200×10^{-5}	1333	1.8945×10^{-2}
Case 2	2	5.1400×10^{-5}	2	1.5860×10^{-4}	1333	2.5082×10^{-2}
Case 3	2	6.3100×10^{-5}	2	6.6500×10^{-5}	1333	1.8365×10^{-2}
Case 4	2	5.2100×10^{-5}	2	1.2790×10^{-4}	1333	1.8173×10^{-2}

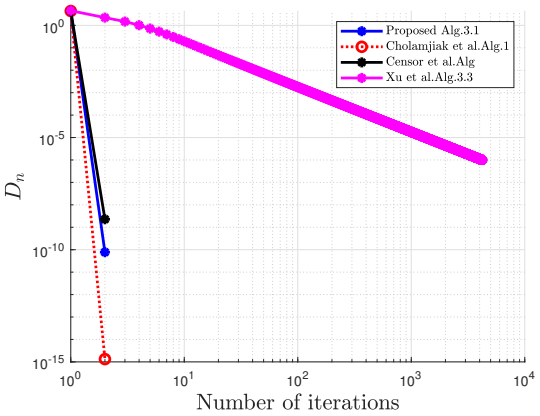

Figure 17. Example 4.2 case 1: category I.

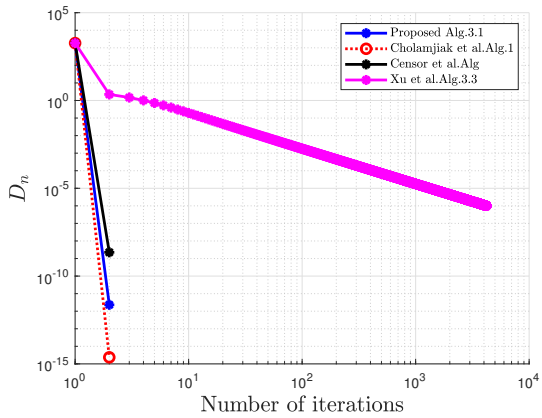

Figure 18. Example 4.2 case 2: category I.

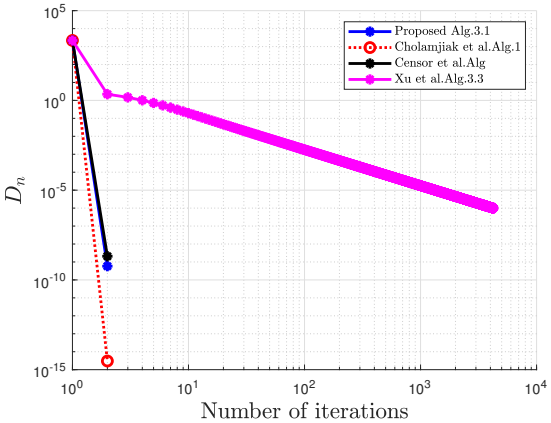

Figure 19. Example 4.2 case 3: category I.

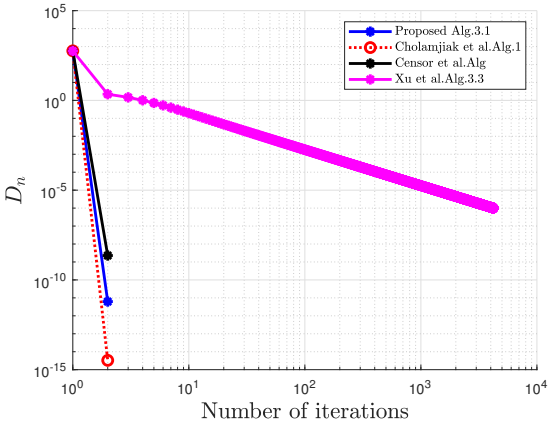

Figure 20. Example 4.2 case 4: category I.

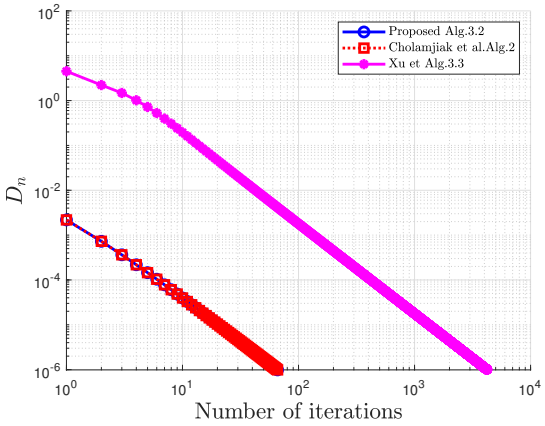

Figure 21. Example 4.2 case 1: category II.

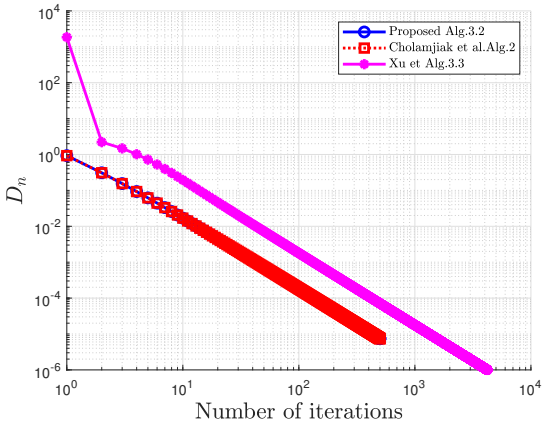

Figure 22. Example 4.2 case 2: category II.

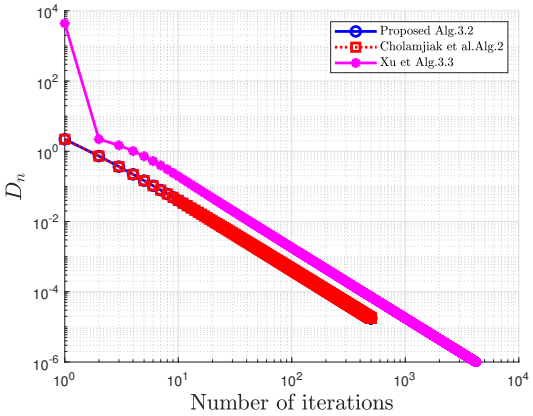

Figure 23. Example 4.2 case 3: category II.

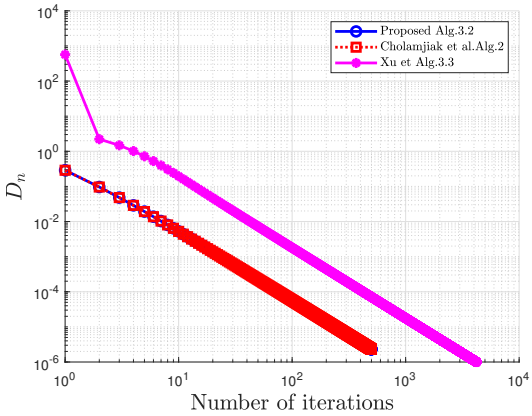

Figure 24. Example 4.2 case 4: category II.

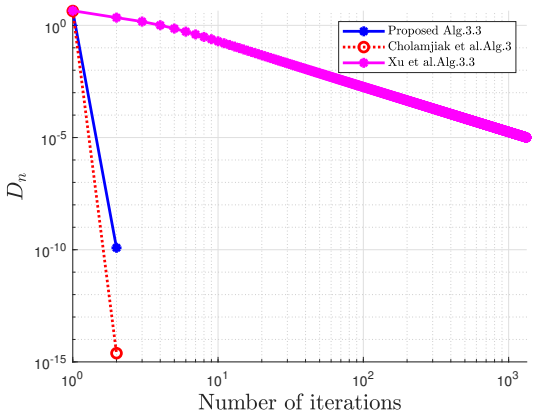

Figure 25. Example 4.2 case 1: category III.

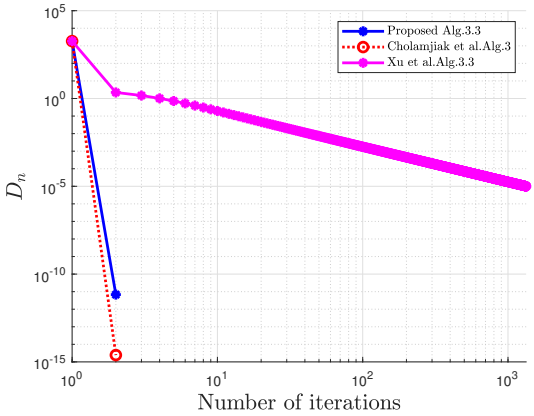

Figure 26. Example 4.2 case 2: category III.

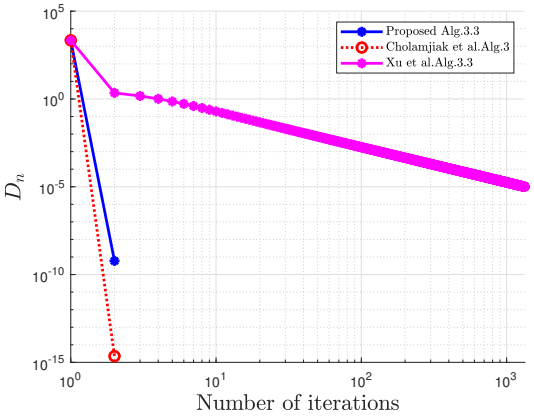

Figure 27. Example 4.2 case 3: category III.

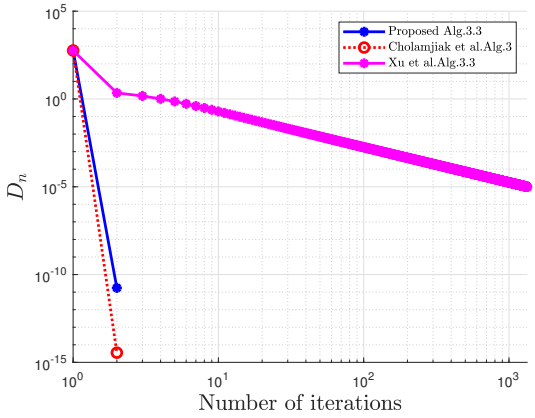

Figure 28. Example 4.2 case 4: category III.

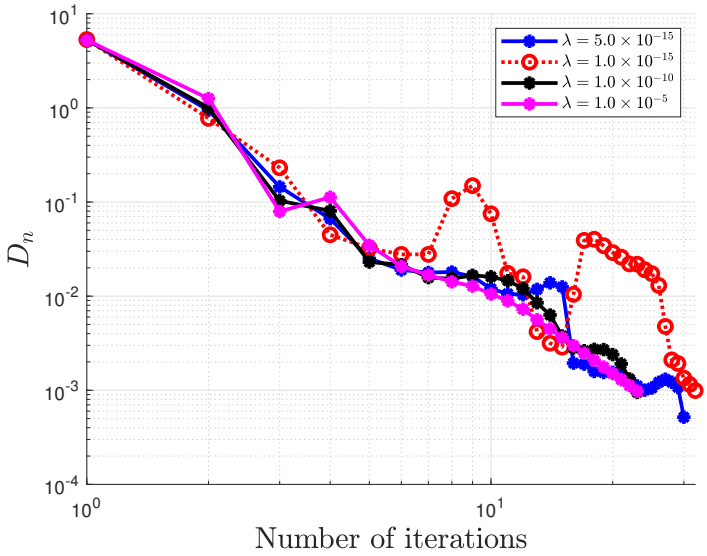

Figure 29. Example 4.3: different values of λ for Algorithm 3.1.

Example 4.3. Consider the separable, convex, and quadratic programming problem, which has been derived from [57] and discussed in [40] for their numerical experiments.

$$\min_{x,y}\{F_1(x) + F_2(y) \mid Ax = y, x \in C, y \in Q\} \tag{4.2}$$

where

$$F_1(x) = \frac{1}{2}x^T M_1 x + q_1^T x \quad \text{and} \quad F_2(y) = \frac{1}{2}y^T M_2 y + q_2^T y.$$

Problem (4.2) can be rewritten into the SVIP (1.1)-(1.2) with $f(x) = M_1 x + q_1$ and $g(y) = M_2 y + q_2$. This problem is more general than Example 4.1.

For $i = 1, 2$, the matrices M_i are formed as: $M_i = V_i \Sigma_i V_i^T$, where $V_i = I - \frac{2v_i v_i^T}{\|v_i\|^2}$ and $\Sigma_i = diag(\sigma_{i1}, \sigma_{i2}, \sigma_{i3}, \cdots, \sigma_{in_i})$ are a Householder and diagonal matrix, respectively, with $n_1 = n$ and $n_2 = m$ being the dimensionality of x and y, respectively.
Furthermore, set

$$\sigma_{ij} = \cos\frac{j\pi}{n_i + 1} + 1 + \frac{\cos\frac{\pi}{n_i+1} + 1 - \hat{C}_i\Big(\cos\frac{n_i\pi}{n_i+1} + 1\Big)}{\hat{C}_i - 1},\ j = 1, 2, \cdots, n_i$$

where $\hat{C}_i$ is the preset condition number of M_i and $\hat{C}_i = 10^4$ for all $i = 1, 2$ as presented in [40]. The vector $v_i \in \mathbb{R}_i^n$ $(i = 1, 2)$ is uniformly distributed in $(-1, 1)$. The linear operator A is defined in the same way as in Example 4.1. The convex set $C := B_1(0)$ (i.e., the ball centered at 0 with radius 1) and the box set Q is as defined in 4.1 with x^* a sparse vector, and its components uniformly distributed in $(0, 1)$. We have $\alpha_1 = \frac{1}{\sigma_{11}} \approx 0.5$, and $\alpha_2 = \frac{1}{\sigma_{21}} \approx 0.5$. Since the strongly convex quadratic program (4.2) has a unique minimizer $(x^*, y^*) \in C \times Q$ and its first order optimality conditions are equivalent to the SVIP (1.1)-(1.2), the solution set Γ of the SVIP is nonempty.

Consider the same initial data as in Example 4.1:
case 1: $m = 30$ & $n = 60$.
case 2: $m = 10$ & $n = 40$.
case 3: $m = 20$ & $n = 50$.
case 4: $m = 30$ & $n = 80$.
Choose $x_0 = \omega_0 = rand(n, 1)$ and $\omega = \frac{x_0}{\|x_0\|}$.

Set the stopping criterion as follows:

$$D_n := \|x_{n+1} - x_n\| \le tol = 10^{-3}.$$

Table 15. Parameters of each Algorithm compared for Example 4.3.

Algorithm	**Parameters**		
Proposed Alg. 3.1	$\lambda = 1.0 \times 10^{-15}$	$\rho_n = 1.8$	–
Cholamjiak et al. Alg. 1	$\tau_0 = 0.999$	$\rho_n = 1.8$	–
Censor et al. Alg.	$L = \max \lvert\text{eigenvalue}(A^T A)\rvert$	$\gamma = 0.9999 \times \frac{1}{L}$	$\lambda = 1.0 \times 10^{-15}$
Proposed Alg. 3.2	$\lambda = 1.0 \times 10^{-15}$ $\beta_{-1} = 0.01$	$\rho_n = 1.9$	$\beta_0 = 0.5$
Cholamjiak et al. Alg. 2	$\tau_{-1} = 0.999$ $\beta_{-1} = 0.01$	$\rho_n = 1.9$	$\beta_0 = 0.5$
Proposed Alg. 3.3	$\lambda = 1.0 \times 10^{-15}$	$\rho_n = 1.9$	$\beta_{n+1} = 0.5$
Cholamjiak et al. Alg. 3	$\tau_0 = 0.999$	$\rho_n = 0.5$	$\beta_{n+1} = 0.5$

Table 16. Example 4.3: numerical performance of different λ for Algorithm 3.1.

	$\lambda = 5.0 \times 10^{-15}$		$\lambda = 1.0 \times 10^{-15}$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.1	30	2.1960×10^{-4}	32	1.8530×10^{-4}
	$\lambda = 1.0 \times 10^{-10}$		$\lambda = 1.0 \times 10^{-5}$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.1	23	2.5380×10^{-4}	23	2.4110×10^{-4}

Table 17. Example 4.3: numerical performance of different λ for Algorithm 3.2.

	$\lambda = 5.0 \times 10^{-15}$		$\lambda = 1.0 \times 10^{-15}$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.2	27	1.0647×10^{-2}	22	2.4460×10^{-3}
	$\lambda = 1.0 \times 10^{-10}$		$\lambda = 1.0 \times 10^{-5}$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.2	22	2.9550×10^{-3}	25	1.6352×10^{-3}

Table 18. Example 4.3: numerical performance of different λ for Algorithm 3.3.

	$\lambda = 5.0 \times 10^{-15}$		$\lambda = 1.0 \times 10^{-15}$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.3	9	4.4674×10^{-3}	8	3.8440×10^{-4}
	$\lambda = 1.0 \times 10^{-10}$		$\lambda = 1.0 \times 10^{-5}$	
	Iter.	CPU Time	Iter.	CPU Time
Proposed Alg. 3.3	9	4.8820×10^{-4}	7	2.3300×10^{-4}

Table 19. Example 4.3: Comparison results of proposed Alg. 3.1, Cholamjiak et al. Alg.1, Censor et al. Alg., and Xu et al. Alg.3.3 at different cases of initial points.

	Proposed 3.1		**Cholamjiak et al. Alg.1**		**Censor et al. Alg.**		**Xu et al. Alg.3.3**	
	Iter.	CPU (Time)	Iter.	CPU (Time)	Iter.	CPU (Time)	Iter.	CPU (Time)
Case 1	22	2.2880×10^{-4}	54	3.7540×10^{-4}	27	1.1038×10^{-3}	31	1.3040×10^{-4}
Case 2	18	2.9820×10^{-4}	53	4.0080×10^{-4}	43	1.0378×10^{-3}	47	3.8480×10^{-4}
Case 3	21	1.4610×10^{-4}	52	1.5040×10^{-4}	23	1.5900×10^{-3}	51	3.0880×10^{-4}
Case 4	23	1.9250×10^{-4}	70	2.8220×10^{-4}	26	1.6706×10^{-3}	47	3.7990×10^{-4}

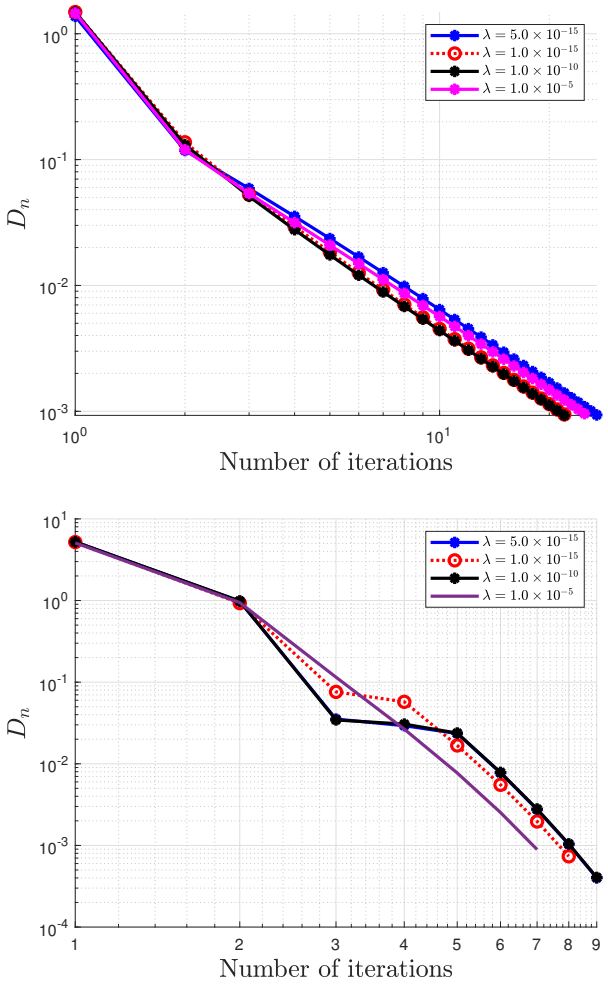

Figure 30. Example 4.3: different values of λ for Algorithms 3.2 and 3.3.

Table 20. Example 4.3: Comparison results of proposed Alg. 3.2, Cholamjiak et al Alg. 2, and Xu et al Alg. 3.3 at different cases of initial points.

	Proposed Alg.3.2		**Cholamjiak et al. Alg. 2**		**Xu et al. Alg. 3.3**	
	Iter.	CPU (Time)	Iter.	CPU (Time)	Iter.	CPU (Time)
Case 1	9	1.7082×10^{-3}	21	3.4120×10^{-4}	54	3.4120×10^{-4}
Case 2	2	3.7100×10^{-5}	15	1.0720×10^{-4}	50	1.9590×10^{-4}
Case 3	2	4.6800×10^{-5}	11	9.9100×10^{-5}	40	1.4330×10^{-4}
Case 4	20	2.2097×10^{-3}	20	2.7830×10^{-4}	47	4.0090×10^{-4}

Table 21. Example 4.3: Comparison results of proposed Alg. 3.3, Cholamjiak et al Alg. 3, and Xu et al Alg. 3.3 at different cases of initial points.

	Proposed Alg.3.3		**Cholamjiak et al. Alg. 3**		**Xu et al. Alg. 3.3**	
	Iter.	CPU (Time)	Iter.	CPU (Time)	Iter.	CPU (Time)
Case 1	8	6.8000×10^{-5}	11	1.1530×10^{-4}	49	2.2680×10^{-4}
Case 2	10	7.6400×10^{-5}	11	1.1680×10^{-4}	44	2.0540×10^{-4}
Case 3	8	7.9300×10^{-5}	16	1.0580×10^{-4}	39	1.7480×10^{-4}
Case 4	8	1.1460×10^{-4}	11	1.3500×10^{-4}	49	3.5820×10^{-4}

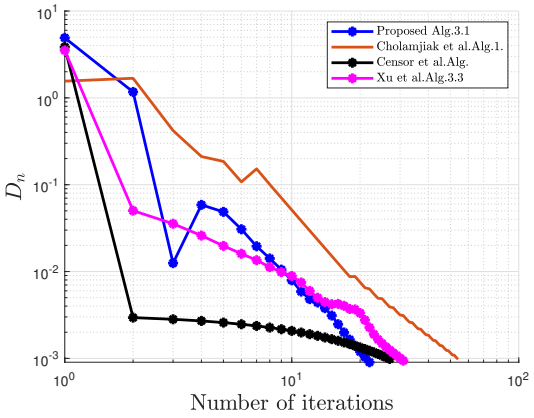

Figure 31. Example 4.3 case 1: category I.

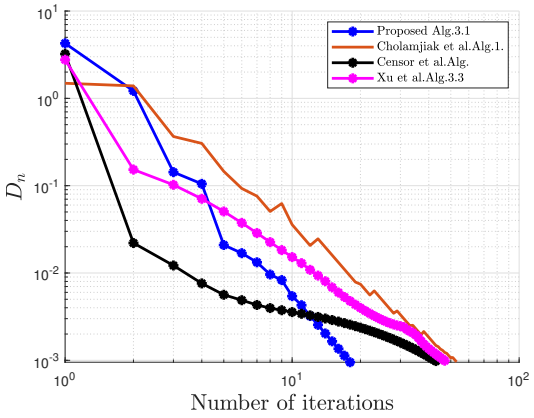

Figure 32. Example 4.3 case 2: category I.

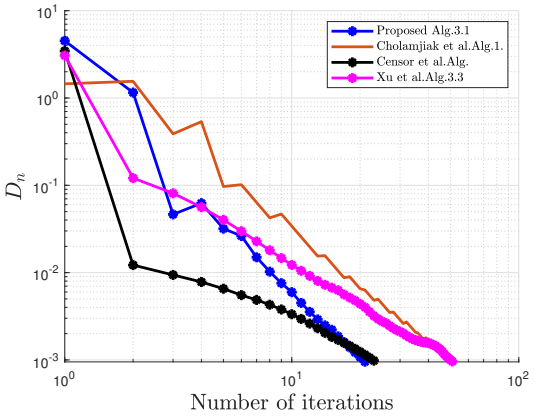

Figure 33. Example 4.3 case 3: category I.

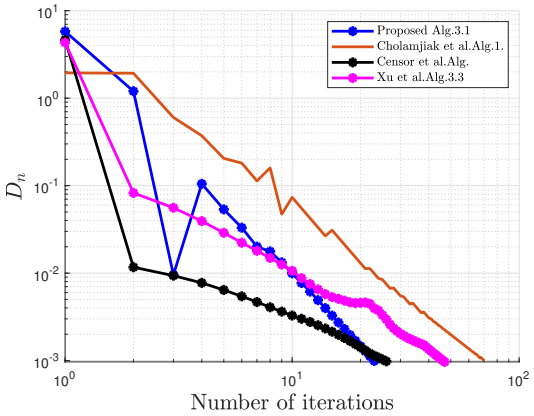

Figure 34. Example 4.3 case 4: category I.

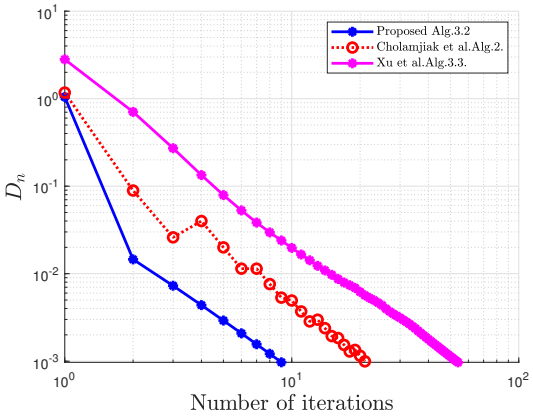

Figure 35. Example 4.3 case 1: category II.

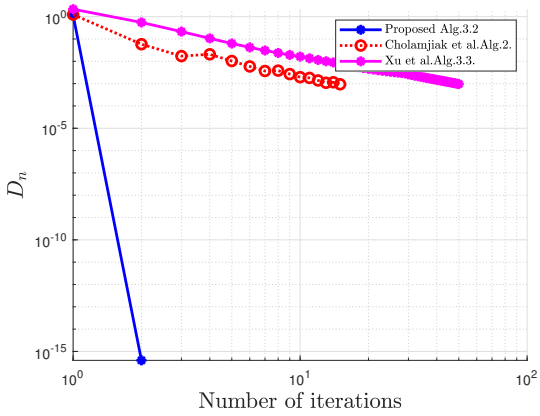

Figure 36. Example 4.3 case 2: category II.

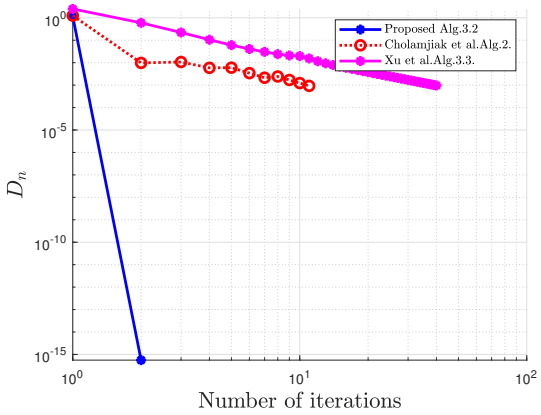

Figure 37. Example 4.3 case 3: category II.

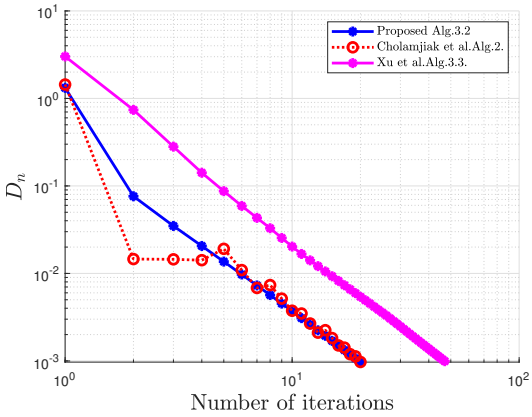

Figure 38. Example 4.3 case 4: category II.

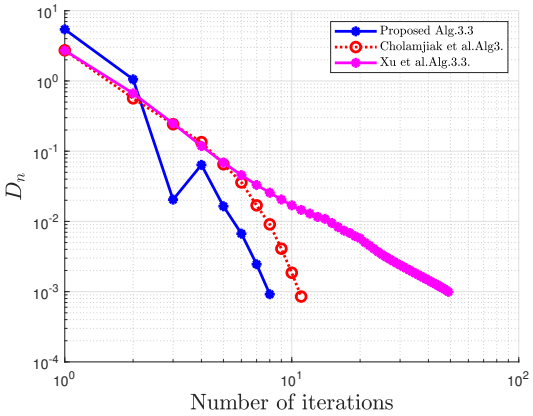

Figure 39. Example 4.3 case 1: category III.

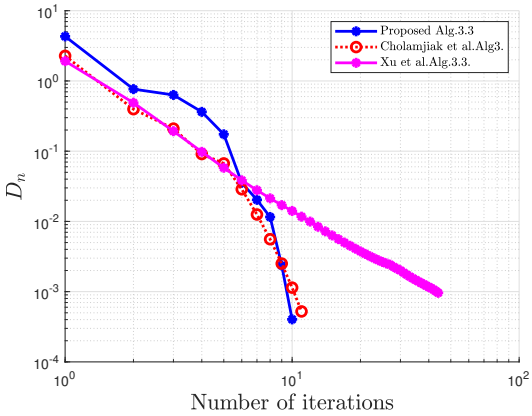

Figure 40. Example 4.3 case 2: category III.

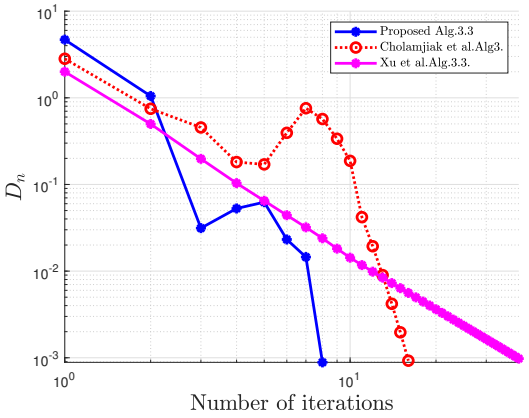

Figure 41. Example 4.3 case 3: category III.

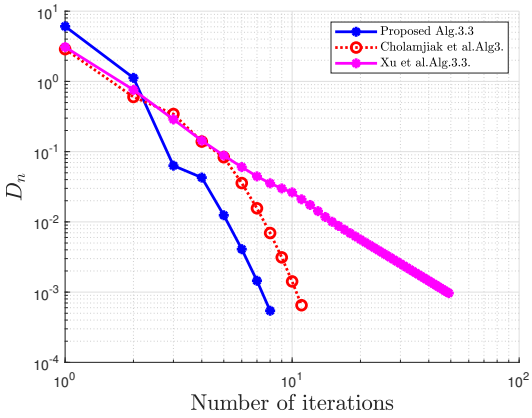

Figure 42. Example 4.3 case 4: category III.

Remark 4.4.

1. To assess the sensitivity of the proposed algorithms to parameter choices, we test several values of λ for the problems considered. For Example 4.1, the results in Table 1 and Figure 1 show that Algorithm 3.1 performs best when $\lambda = 0.0001$. Table 2 and Figure 2 indicate that the optimal λ for Algorithm 3.2 is 1.0×10^{-10}, while Table 3 and Figure 2 reveal that Algorithm 3.3 achieves its best performance at $\lambda = 1.0 \times 10^{-5}$.
2. For Example 4.2, Tables 9 and 10, and Figure 15 indicate that $\lambda = 1.0 \times 10^{-5}$ is the optimal value for Algorithm 3.1, while $\lambda = 1.0 \times 10^{-10}$ is the optimal value for Algorithms 3.2. Additionally, Table 11 and Figure 16 show that $\lambda = 1.0 \times 10^{-15}$ is the optimal value for Algorithm 3.3.
3. For Example 4.3, Table 17 and Table 18, and Figure 30 indicate that the optimal λ for Algorithms 3.2 and 3.3 is 1.0×10^{-5}, while Table 16 and Figure 29 show that Algorithm 3.1 achieves it best performance at $\lambda = 1.0 \times 10^{-15}$.
4. Across both Examples 4.1 4.2 , and 4.3, the proposed algorithms exhibit strong numerical performance. In Example 4.1, Table 4 and Figures 3–6 demonstrate that Algorithm 3.1 outperforms the methods of Cholamjiak et al. (Algorithm 1), Censor et al.Algorithm, and Xu et al. (Algorithm 3.3) in terms of iteration count and CPU time.
5. Table 5 and Figures 7–10 further show that Algorithm 3.2 surpasses Cholamjiak et al. (Algorithm 2) and Xu et al. (Algorithm 3.3) in iteration count and CPU time. Likewise, Table 6 and Figures 11–14 indicate that Algorithm 3.3 achieves superior performance compared with the Cholamjiak et al. (Algorithm 2) and Xu et al. (Algorithm 3.3).
6. For Example 4.2, the results in Tables 12–14 and Figures 17–28 confirm that all three proposed algorithms (3.1–3.3) outperform Cholamjiak et al. Algorithms 1–3, Censor et al. Algorithm, and the Xu et al. in terms of CPU time. Then, they perform competitively in iteration count with the Cholamjiak et al. Algorithm.
7. For Example 4.3, Tables 19–21 and Figures 31–42 show that our Algorithms 3.1 and 3.3 outperform Censor et al.'s algorithm, Cholamjiak et al.'s Algorithm Alg.1 andd Alg.3, and Xu et al.'s Algorithm 3.3 in terms of iteration count and CPU time. Additionally, our Algorithm 3.2 performs competitively with Cholamjiak et al. Alg.2 and Xu et al.'s Algorithm 3.3 in terms of CPU time while outperforming their Algorithms in terms of iteration count.

5. Classification problem applications

5.1. Description of extreme learning machine

In this section, the Extreme Learning Machine (ELM), introduced by Huang et al. [58], is used for the classification of heart disease, lung cancer, heart failure, and prostate cancer. We apply our efficient algorithms 3.1, 3.2, and 3.3 to enhance the predictive performance of the ELM by accurately computing the optimal output weights, thus improving early detection and classification of diseases.
ELM is a feedforward neural network with one hidden layer. It randomly assigns input weights ω_j and biases b_j between the input and N hidden layers at the j-th hidden node, while only the output weights are computed by solving a linear system. The hidden layer uses activation functions to process the input, and our algorithm focuses on optimizing the output weight for better performance. The objective of this experiment is to apply an efficient algorithm (SVIP) that enhances the predictive performance of the ELM by accurately computing the optimal output weights, thus improving the early detection and classification of heart disease, lung cancer, heart failure, and prostate cancer.

The output linear weights β_j are the only unknown parameters that need to be determined when

$$V := \Big\{(x_j, t_j) : \ x_j \in \mathbb{R}^n, \ t_j \in \mathbb{R}^m, \ j = 1, 2, \cdot, M\Big\}$$

is a set of training data with M distinct samples such that x_j is an input training data and t_j is a target. The objective of the ELM method is to find the optimal output weight using the output function passing the activation function $g(x)$. The output linear weights for single-hidden layer feed forward neural networks (SLFNs) is

$$c_i = \sum_{j=1}^{N} \beta_j h\Big(\langle w_j, x_i\rangle + b_j\Big),$$

and the hidden layer output matrix $\mathcal{H}$ is

$$\mathcal{H} = \begin{bmatrix} g\Big(\langle \omega_1, x_1\rangle + b_1\Big) & \cdots & g\Big(\langle \omega_N, x_1\rangle + b_N\Big) \\ \vdots & \ddots & \vdots \\ g\Big(\langle \omega_1, x_N\rangle + b_1\Big) & \cdots & g\Big(\langle \omega_N, x_N\rangle + b_N\Big) \end{bmatrix}.$$

Suppose that an optimal weight $\beta = [\beta_1^T, \cdots, \beta_N^T]^T$ is such that $\mathcal{H}\beta = \mathcal{T}$, where $\mathcal{T} = [t_1^T, \cdots, t_M^T]^T$ is the training target data. The least squares problem is utilized to solve the linear system $\mathcal{H}\beta = \mathcal{T}$, noting that it is difficult to obtain the Moore-Penrose generalized inverse of $\mathcal{H}$.

Consider solving the least square problem

$$\min_{\beta \in H} \Big\{\|\mathcal{H}\beta - \mathcal{T}\|_2^2\Big\}. \tag{5.1}$$

In this case, take $f(\beta) = \frac{1}{2}\|(I - P_Q(I - \lambda g))\mathcal{H}\beta\|_2^2$, and $Q = \{\mathcal{T}\}$ in Algorithms 3.1-3.3 to solve the problem in Equation 5.1. This formulation captures the core optimization objective in ELM, which is to minimize the squared discrepancy between actual output ($\mathcal{T}$) and model predictions using the hidden layer output matrix ($\mathcal{H}$) and the output weight vector β.

Cross-validations are used to estimate the performance of machine learning models to avoid overfitting in a predictive model. The data is divided into 5 folds and trained by the following cross-validations.

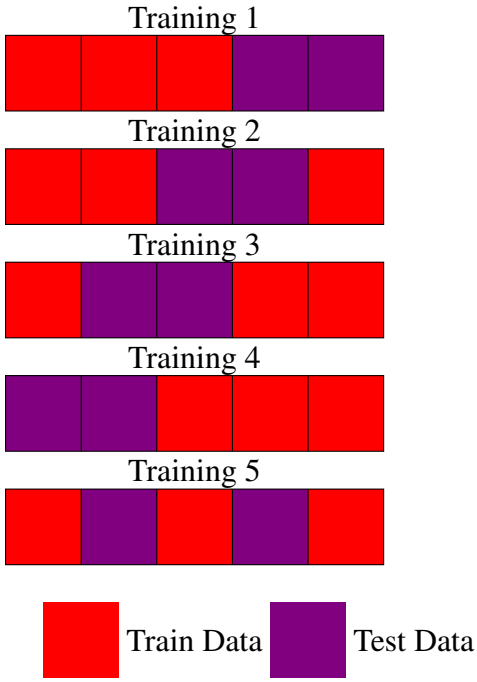

Figure 43. 5-fold data cross-validation.

In Figure 43, we employ five-fold stratified cross-validation, wherein the dataset is divided into five equally sized folds while preserving the original class distribution within each fold. The model is iteratively trained on four folds and validated on the remaining one, cycling through all combinations. This approach mitigates the risk of overfitting by exposing the model to varied training and validation sets, and it provides a more reliable estimate of predictive accuracy across the dataset [59].

5.2. Description of statistical measures

The mean square error (MSE) represents the average of the squared difference between the observed values and predicted values.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{Y}_i)^2.$$

where n is the observations in the dataset, y_i is the actual value of the observation, and $\hat{Y}_i$ is the predicted values of the ith observation.
The root (RMSE) is the square root of the mean square error. That is,

$$RMSE = \sqrt{MSE}.$$

The mean squared error percentage difference is given by the formula

$$\frac{|MSE_{Train} - MSE_{Test}|}{|MSE_{Train}|} \times 100.$$

The percentage difference of the root mean squared error is given by the formula

$$\frac{|RMSE_{Train} - RMSE_{Test}|}{|RMSE_{Train}|} \times 100.$$

The percentage differences in the MSE and the RMSE are essential metrics for assessing model performance, as they offer insight in measuring the accuracy of predictive models. In particular, MSE measures the magnitude of the error and its implications for the performance of the model. A lower MSE indicates that the model predictions are closer to the actual values, signifying better accuracy, while higher MSE denotes that the model's predictions deviate further from the actual value, showing poorer performance. These metrics are valuable for:

1. Model Evaluation: The performance of different machine learning algorithms is compared. A smaller percentage difference indicates stability, while a larger one points to instability.
2. Generalization Assessment: This helps evaluate a model's ability to generalize data by assessing the goodness of fit. A large percentage difference between training and testing errors may indicate overfitting.
3. Optimization: It involves minimizing MSE during model training to improve predictive accuracy.
4. Predictive modeling: MSE evaluates the accuracy of the forecasting models.

5.3. Numerical experiment

The preprocessing pipeline for the four biomedical datasets follow a consistent structure to ensure fair model training across all 5 folds. All numerical predictors are first standardized using Z-score normalization so that each feature had zero mean and unit variance, preventing scale-dependent variables (such as cholesterol, blood pressure, and maximum heart rate in the case heart disease) from dominating the learning process. Missing values, when present, are handled through dataset-specific cleaning procedures to ensure that no incomplete records distort the training distribution. Because the heart disease, lung cancer, heart failure, and prostrate cancer dataset exhibit class imbalance, the minority class is upsampled using bootstrap resampling to match the majority class, producing a balanced dataset before cross-validation. After preprocessing, the data are partitioned using 5-fold cross-validation, where each fold uses approximately 80% of the data for training and 20% for testing, ensuring that every observation served once as test data and four times as training data. This procedure provides an unbiased and robust estimate of model performance across all algorithms evaluated.

5.3.1. Experiment 1: heart disease classification

Heart disease remains a critical global health concern, with early-stage underdiagnosis especially in low-resource settings. It remains the leading cause of death worldwide, accounting for over 18 million deaths annually (WHO, 2025), driving disproportionately high mortality rates. We apply our proposed algorithms 3.1, 3.2, and 3.3 by updating the optimal weights in our machine learning model using the heart disease dataset. The data comes from the University of California Irvine's Machine Learning Repository at https://archive.ics.uci.edu/ml/datasets/Heart+Disease. The Heart disease data set is used as a training set to demonstrate the superiority of our algorithms over existing ones in the literature. This dataset comprises 150 No heart disease and 120 heart disease identified by heart disease diagnosis, as represented in the bar chart below.

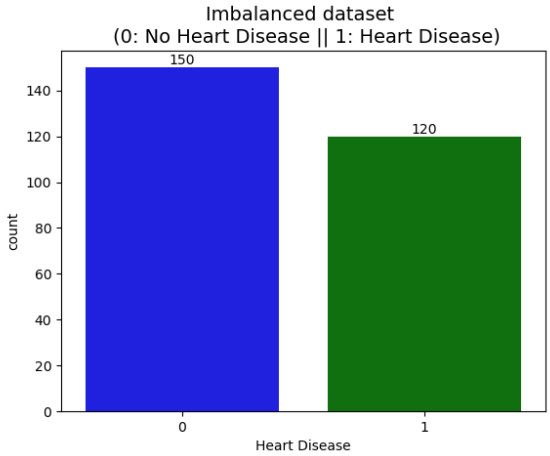

Figure 44. Bar chart of heart disease imbalanced data.

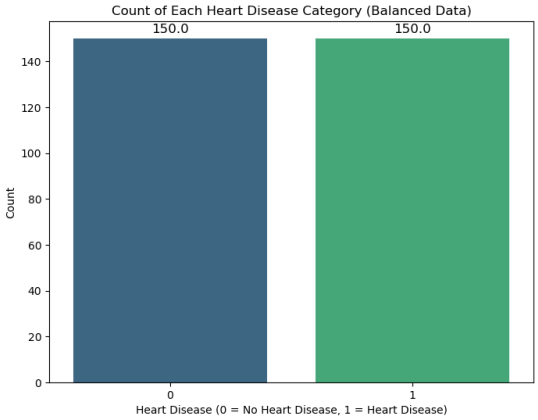

Figure 45. Bar chart of heart disease balanced data.

The descriptive statistics of the heart disease dataset is shown in Table 22.

Table 22. Heart disease dataset description from UC Irvine

Attribute	$\bar{x}$	min	max	std	25%	50%	75%
Age	54.43	29	77	9.11	48	55	61
Sex	0.67	0	1	0.47	0	1	1
Chest pain type (cpt)	3.17	1	4	0.95	3	3	4
Blood pressure (BP)	131.34	94	200	17.86	120	130	140
Cholesterol (chol)	249.66	126	564	51.69	213	245	280
Fasting blood sugar over 120 (FBS)	0.15	0	1	0.36	0	0	0
Electrocardiogram result (EKG)	1.02	0	2	1	0	2	2
Maximum heart rate(MHR)	149.68	71	202	23.17	133	153	166
Exercise angina (Exer)	0.33	0	1	0.47	0	0	1
Segment depression (STDep)	1.05	0	0.50	1.15	0	0.8	1.6
Slope of Segment (SloST)	1.59	1	3.0	0.61	1	2	2
Number of vessels fluoro (vesflu)	0.67	0	3.0	0.94	0	0	1
Thallium (Thal)	4.70	3	7.0	1.94	3	3	7

To visualize the relationships between the attributes, we use statistical tools to analyze the data.

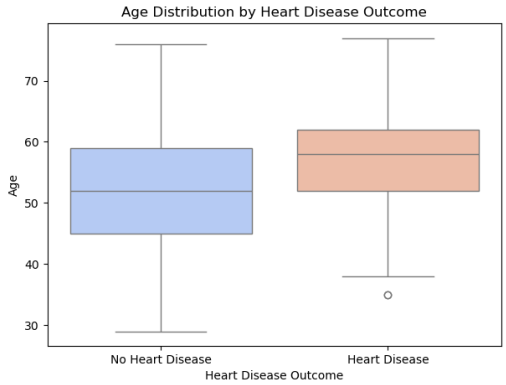

Figure 46. Box plot of age distribution versus heart disease outcome.

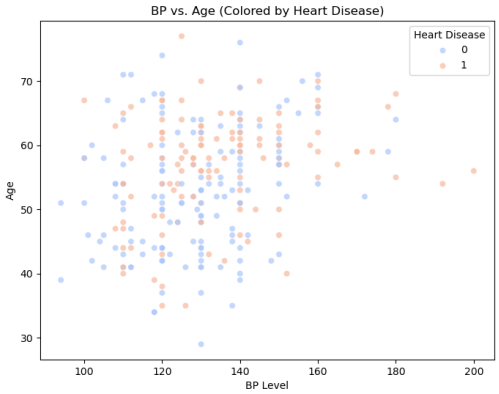

Figure 47. Scatter plot of BP versus age.

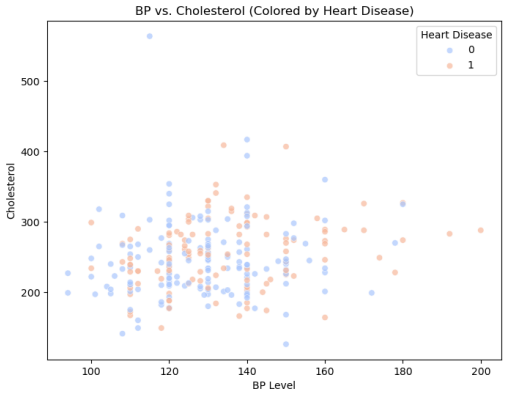

Figure 48. Scatter plot of blood pressure versus cholesterol outcome.

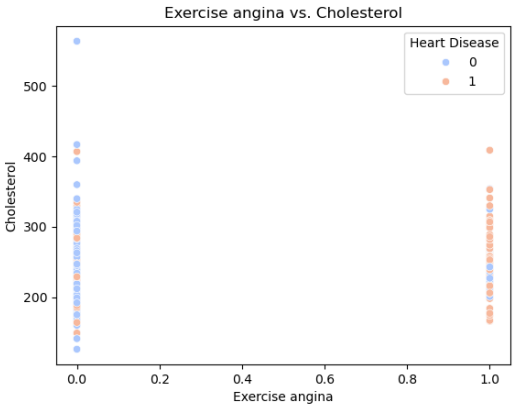

Figure 49. Scatter plot of exercise angina versus cholesterol.

Figure 50. Box plot of age versus ST depression outcome.

Figure 51. Box plot of cholesterol versus heart disease outcome.

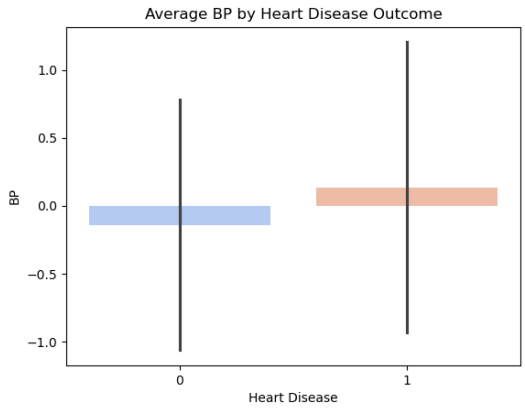

Figure 52. Bar plot of average BP by heart disease outcome.

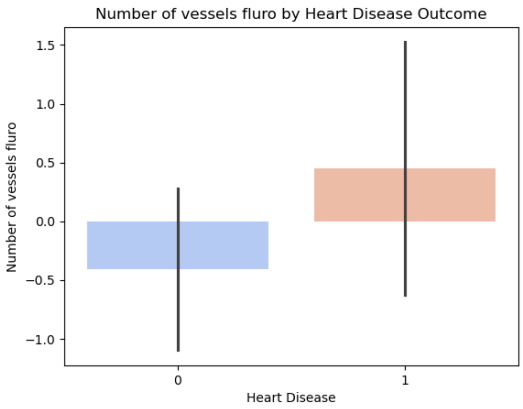

Figure 53. Bar plot of number of vessels fluro by heart disease outcome.

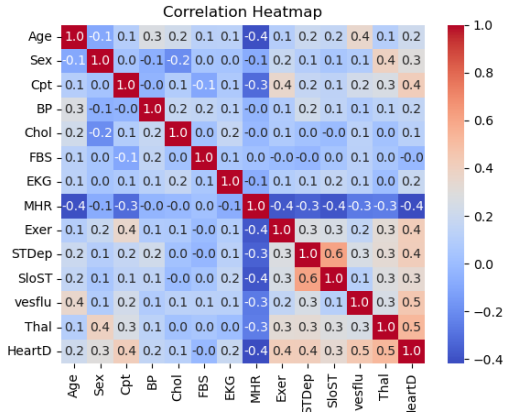

Figure 54. Heart disease unbalanced data correlation heatmap.

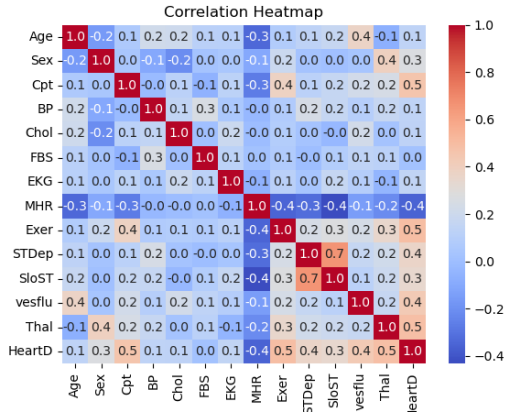

Figure 55. Heart disease balanced data correlation heatmap.

Table 23. Parameter settings for different algorithms.

Algorithm	λ	ρ_n	α	θ	β	μ	γ_0
Alg 3.1	0.0001	0.0001	–	–	–	–	–
Alg 3.2	0.0001	0.0001	–	–	–	–	–
Alg 3.3	0.0001	0.0001	–	–	–	–	–
FISTA	0.0001	0.0001	–	0.0001	–	–	–
IREM	0.0001	0.001	0	0.001	-0.30	0.0001	0.0001
REM	0.0001	0.001	0.0001	–	0	–	–

Table 24. Comparison results for the train and test dataset (heart disease).

	Train Data		Test Data			
	Iter.	CPU Time	Iter.	CPU Time	%△ in MSE	%△ in RMSE
Algorithm 3.1	2	1.6000×10^{-4}	2	1.6000×10^{-4}	1.1049×10^{-2}	5.5262×10^{-2}
Algorithm 3.2	2	2.0000×10^{-4}	2	1.6000×10^{-4}	1.9798×10^{-1}	9.8941×10^{-2}
Algorithm 3.3	2	2.0000×10^{-4}	2	1.6000×10^{-4}	2.8436×10^{-1}	1.4208×10^{-1}
FISTA	> 10000	1.19862	> 10000	1.19862	12.379175	6.009044
IREM	701	1.2203×10^{-1}	701	1.2203×10^{-1}	11.553418	5.618852
REM	701	1.1684×10^{-1}	701	1.1684×10^{-1}	11.555642	5.619905

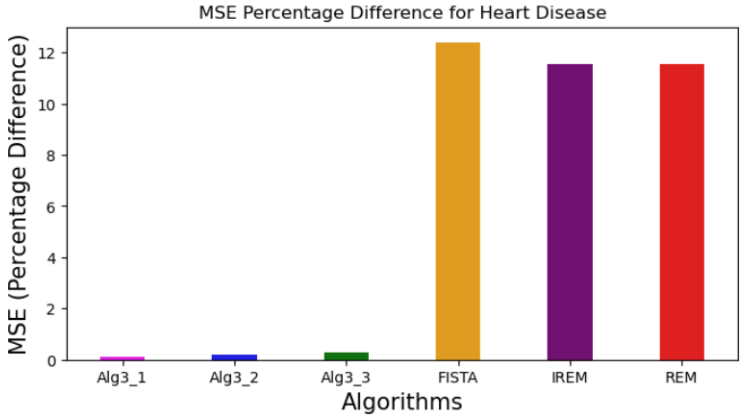

Figure 56. %△ in mean squared error for heart disease train data.

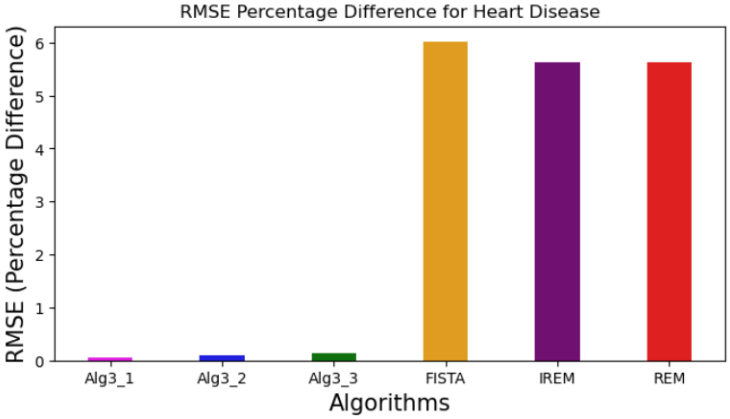

Figure 57. %△ in root mean squared error for heart disease train data.

5.3.2. Experiment 2: Lung cancer classification

Lung cancer ranks as the most commonly diagnosed cancer globally and remains the leading cause of cancer related mortality. Globally, lung cancer contributed to an estimated 2.5 million new diagnoses in 2022, constituting 12% of all cancer cases, and was responsible for approximately 1.8 million deaths, 19% of the total cancer mortality. In this section, we apply our proposed algorithms 3.1, 3.2, and 3.3 to demonstrate the effectiveness of our algorithms by updating the optimal weights in our machine learning model using the lung cancer dataset. The data comes from the website of the online lung cancer prediction system downloaded from Kaggle. By integrating algorithms 3.1 through 3.3 into our machine learning framework, our goal is to minimize the prediction error. The lung cancer dataset used as a training set comprises 39 No lung cancer and 270 lung cancer identified by lung cancer diagnosis, as represented in the bar chart below.

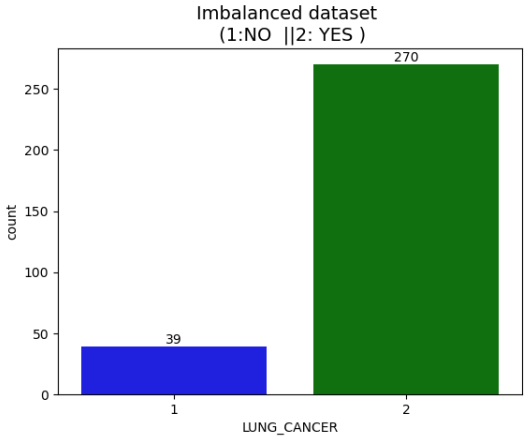

Figure 58. Bar chart of lung cancer imbalanced data.

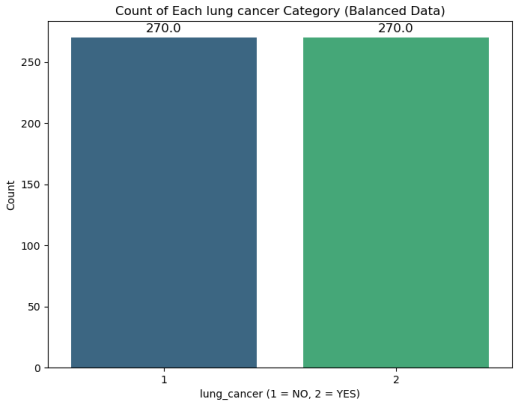

Figure 59. Bar chart of lung cancer balanced data.

The descriptive statistics of the features in the lung cancer dataset are shown in Table 25.
The relationship between each feature and lung cancer is shown below using some statistical tools below.

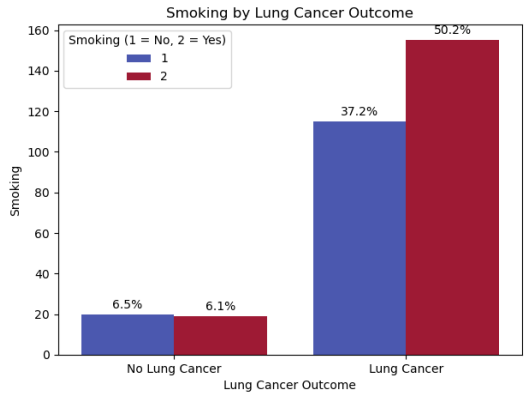

Figure 60. Count plot of smoking by lung cancer.

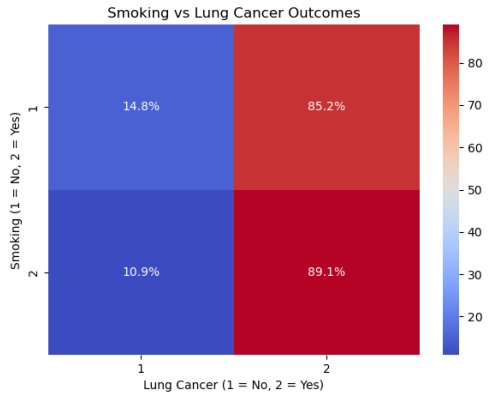

Figure 61. Cross tab of Smoking by lung cancer.

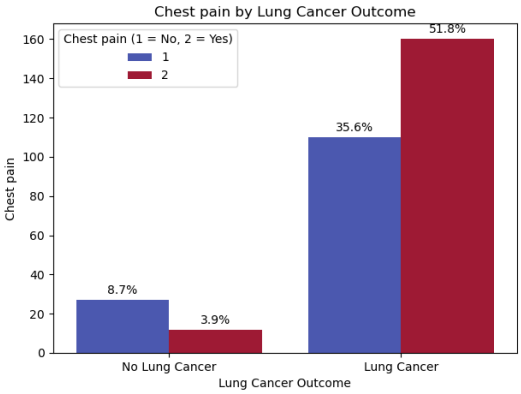

Figure 62. Count plot of chest pain by lung cancer.

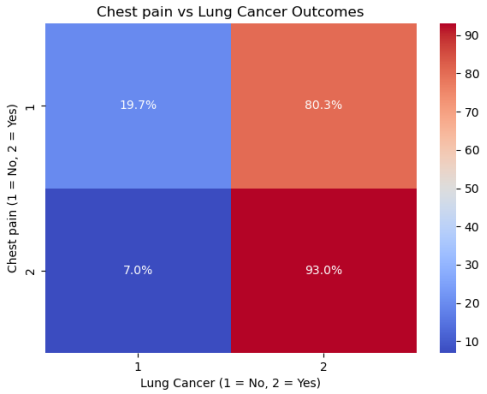

Figure 63. Cross tab of chest pain by lung cancer.

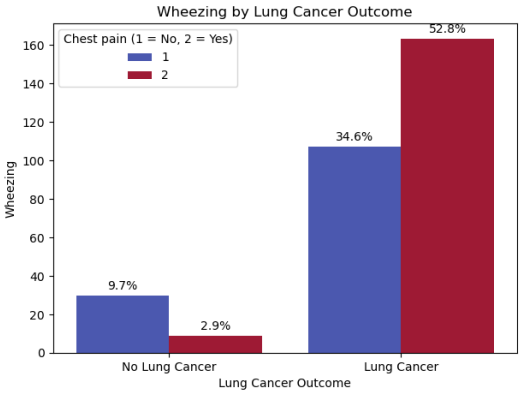

Figure 64. Count plot of wheezing by lung cancer outcome.

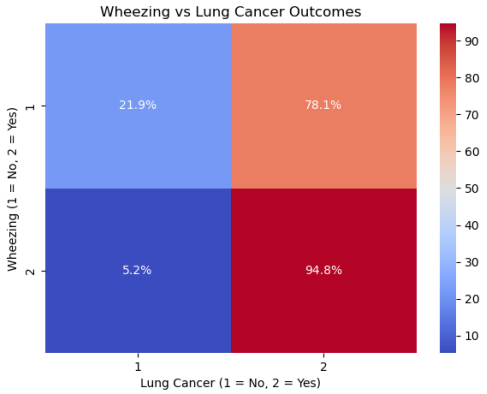

Figure 65. Cross tab of wheezing by lung cancer.

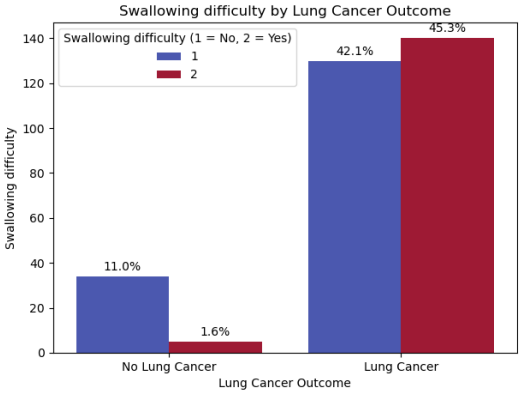

Figure 66. Count plot of swallowing difficulty by lung cancer outcome.

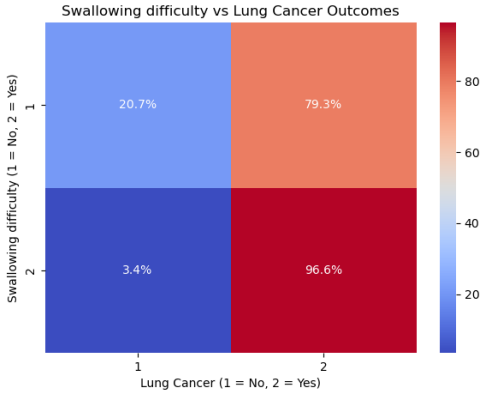

Figure 67. Cross tab of swallowing difficulty by lung cancer.

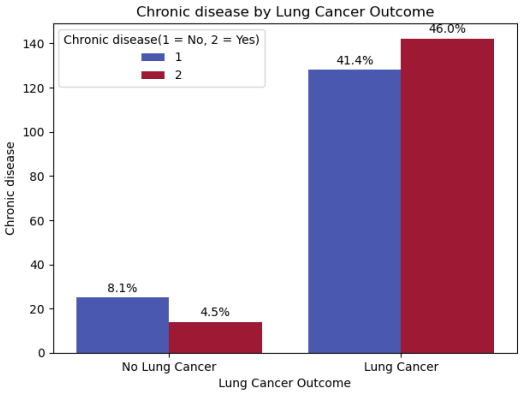

Figure 68. Count plot of chronic disease by lung cancer.

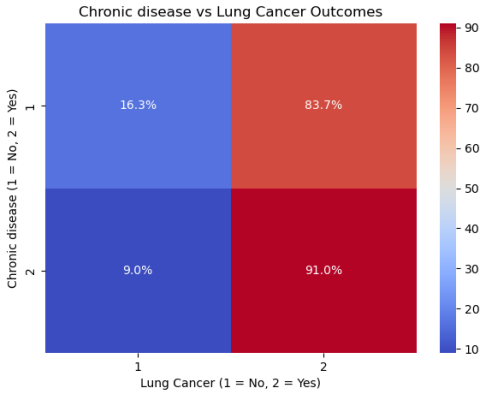

Figure 69. Cross tab of chronic disease by lung cancer.

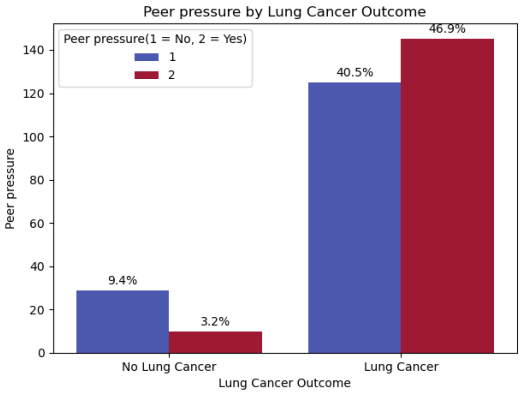

Figure 70. Count plot of peer pressure by lung cancer outcome.

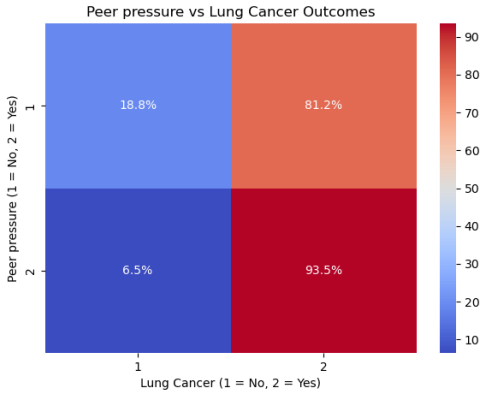

Figure 71. Cross tab plot of peer pressure by lung cancer.

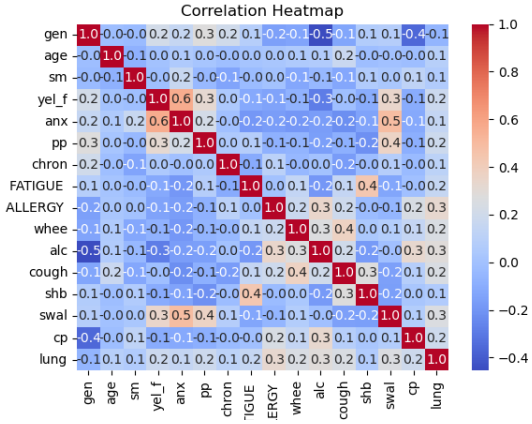

Figure 72. Lung cancer imbalanced data correlation heatmap.

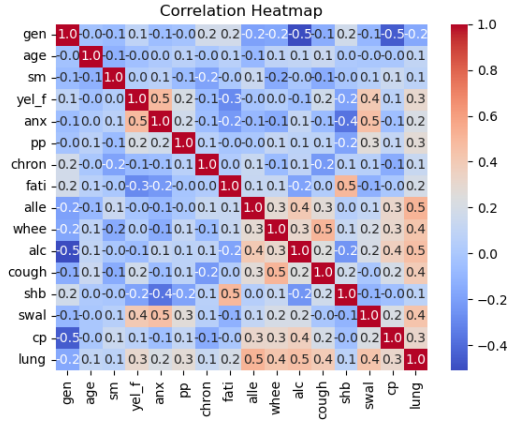

Figure 73. Lung cancer balanced data correlation heatmap.

Table 25. Lung cancer dataset description.

Attribute	$\bar{x}$	min	max	std	25%	50%	75%
Gender	1.48	1	2	0.50	1	1	2
Age	62.67	21	87	8.21	57	62	69
Smoking (sm)	1.56	1	2	0.50	1	2	2
Yellow fingers (yel_f)	1.57	1	2	0.50	1	2	2
Anxiety	1.50	1	2	0.50	1	1	2
Peer pressure (pp)	1.50	1	2	0.50	1	2	2
Chronic disease (chron)	1.50	1	2	0.50	1	2	2
Fatigue	1.673139	1	2	0.47	1	2	2
Allergy	1.556634	1	2	0.50	1	2	1
Wheezing (whee)	1.56	1	2	0.50	1	2	2
Alcohol consumption (alc)	1.56	1	2	0.50	1	2	2
Coughing (cough)	1.58	1	2	0.49	1	2	2
Shortness of breath (shb)	1.64	1	2	0.49	1	2	2
Difficulty swallowing (swal)	1.47	1	2	2	1	2	2
Chest pain (cp)	1.56	1	2	0.50	1	2	2

Table 26. Comparison results for the train and test dataset (lung cancer).

	Train Data		Test Data			
	Iter.	CPU Time	Iter.	CPU Time	%△ in MSE	%△ in RMSE
Algorithm 3.1	2.0	7.1000×10^{-4}	2.0	7.1000×10^{-4}	6.798808	3.459236
Algorithm 3.2	2.0	2.0000×10^{-4}	2.0	2.0000×10^{-4}	6.847710	3.484566
Algorithm 3.3	2.0	7.9000×10^{-4}	2.0	7.9000×10^{-4}	6.845957	3.483658
FISTA	> 10000	1.98925	> 10000	1.98925	10.742734	5.523936
IREM	513.6	0.12019	513.6	0.12019	10.451737	5.370056
REM	514.0	0.12772	514.0	0.12772	10.449784	5.369024

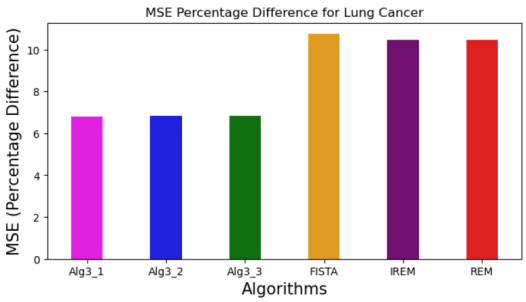

Figure 74. %△ in mean square error for lung cancer train data.

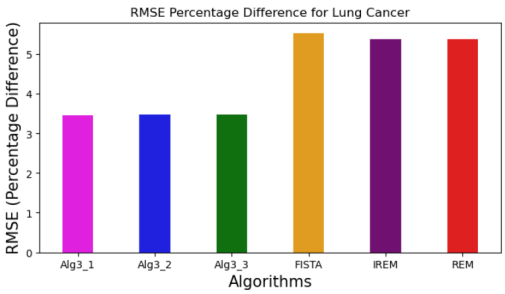

Figure 75. %△ in root mean square error for lung cancer train data.

5.3.3. Experiment 3: Heart failure classification

Heart failure prevalence among adolescents and young adults has shown a significant upward trend globally from 1990 to 2019, with comparable annual growth rates in both sexes. Specifically, the average annual percent change (AAPC) was 0.53% in women and 0.54% in men, indicating a consistent rise across sexes. By 2021, the prevalence rate per 100,000 was higher in men (158.0) than in women (137.6), with men accounting for 1.53 million cases and women 1.27 million cases. The burden of disease, measured by years lived with disability (YLDs), mirrored these prevalence patterns, underscoring the functional and societal impact of HF in younger cohorts. These findings highlight the growing global burden of heart failure among youth, with sex-specific differences that warrant targeted public health strategies and early intervention frameworks. In 2022, heart failure (HF) contributed to 425,147 deaths in the United States, representing 45% of all cardiovascular-related mortality. Age-adjusted HF mortality rates have shown a persistent upward trajectory since 2012, with a marked acceleration during the 2020–2021 period. Notably, the 2021 mortality rate exceeded that of 1999, underscoring the urgent need for enhanced preventive strategies, broader implementation of guideline-directed medical therapies, and intensified research efforts to mitigate the growing burden of HF. In this classification, we apply our proposed algorithms 3.1, 3.2, and 3.3 to improve the ELM by optimizing the output weights in our machine learning model using the heart failure dataset. The heart failure dataset is collected from various sources, including clinical records, diagnostic tests, and patient demographic and is downloaded from kaggle. The dataset comprises a substantial number of individuals diagnosed with heart failure, providing an invaluable resource for studying this prevalent condition and its associated factors.

By integrating algorithms 3.1 through 3.3 into our machine learning framework, our goal is to minimize the prediction error.

The heart failure data set is used as a training set to demonstrate the superiority of our algorithms over existing ones in the literature. This dataset comprises 203 No heart failure (no death) and 96 heart failure (death) identified by heart failure diagnosis, as seen in the bar chart below.

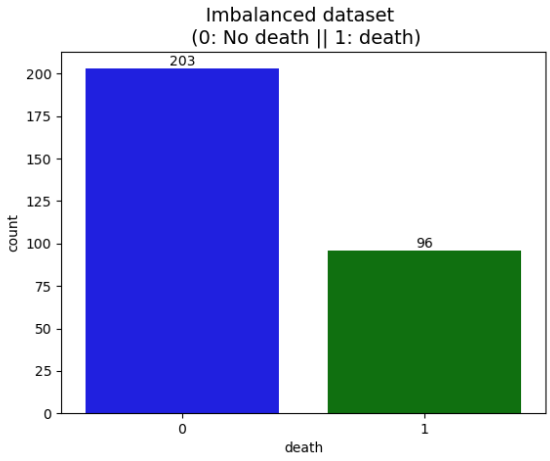

Figure 76. Bar chart of heart failure imbalanced data.

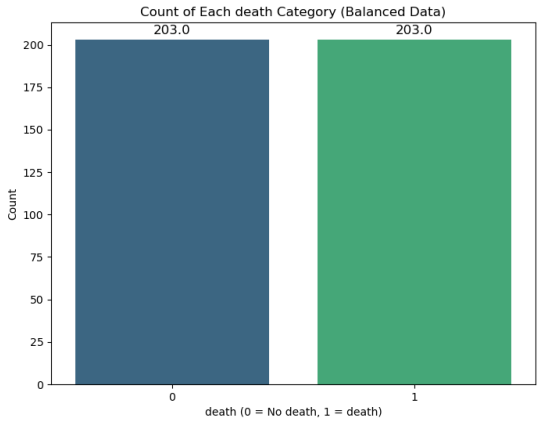

Figure 77. Bar chart of heart failure balanced data.

The statistic description of the features in the heart disease data set is shown in Table 27.

Table 27. Heart failure dataset description.

Attribute	$\bar{x}$	min	max	std	25%	50%	75%
age	60.83	40	95	11.89	51	60	70
anaemia (anae)	0.43	0	1	0.50	0	0	691
creatinine phosphokinase (cpk)	581.84	23.00	7861.00	970.29	116.50	250.00	582.00
diabetes (diab)	0.42	0	1	0.50	0	0	1
ejection fraction	38.08	14	80	11.83	0	0	1
high bp (bp)	0.35	0	1	0.49	0	0	1
platelets	263358.03	25100	850000	97804.24	212500	262000	303500
serum creatinine (serum)	1.39	0.50	9.40	1.03451	0.90	1.10	1.40
sex	0.648829	0	1	0.49	0	1.00	1
smoking (smk)	0.32	0	1	0.47	0	0	1

The relationship between each attribute and heart failure is shown below using some statistical tools.

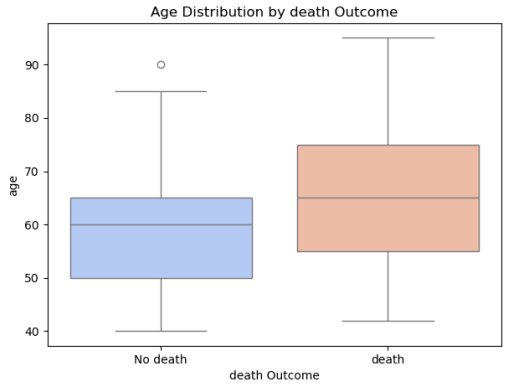

Figure 78. Box plot of age distribution by death outcome.

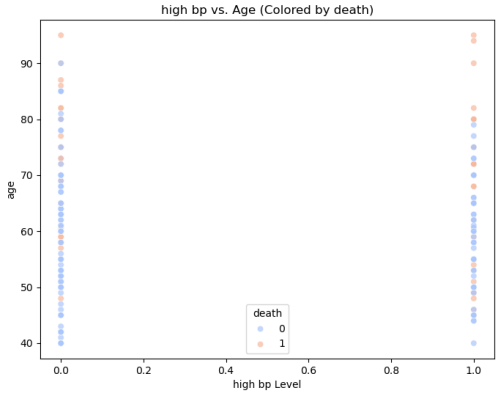

Figure 79. Scatter plot of high bp vs age.

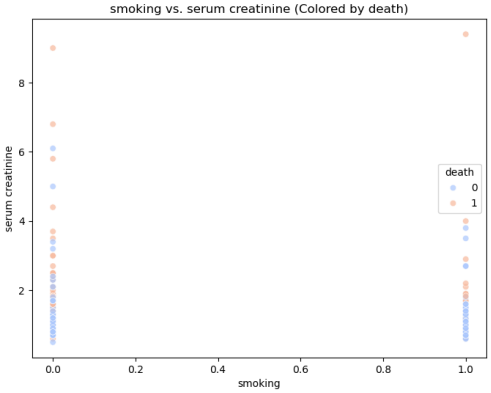

Figure 80. Scatter plot of smoking vs serum creatinine.

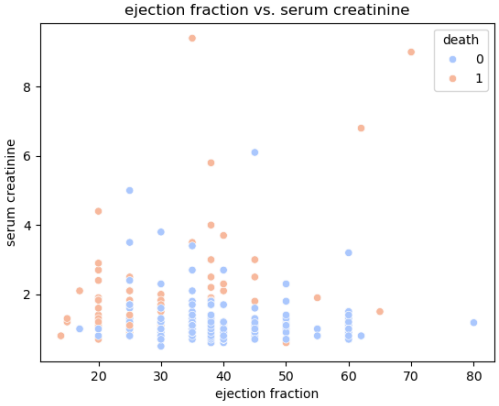

Figure 81. Scatter plot of ejection fraction vs serum creatinine.

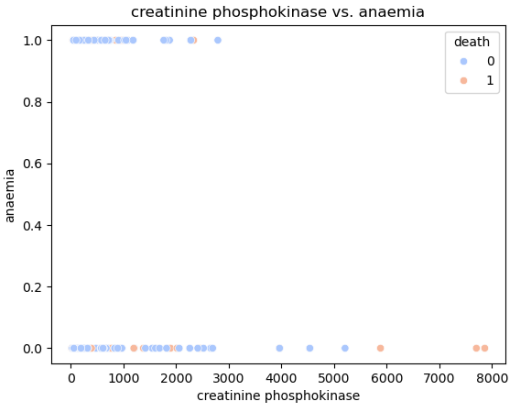

Figure 82. Scatter plot of creatinine phosphokinase vs anaemia.

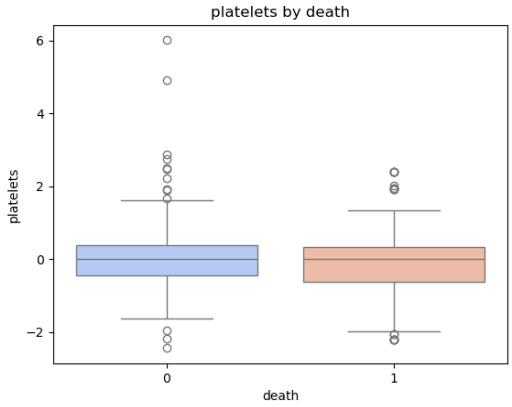

Figure 83. Box plot of platelets vs death.

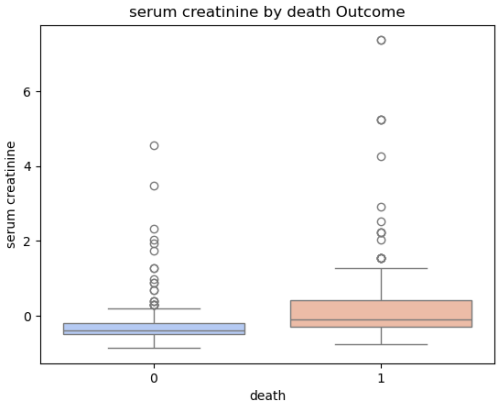

Figure 84. Box plot of serum creatinine by death outcome.

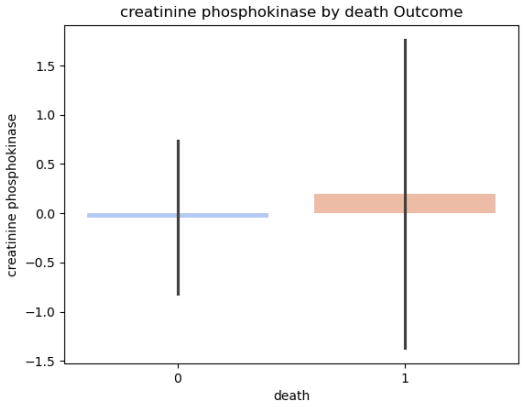

Figure 85. Bar plot of creatinine phosphokinase by death outcome.

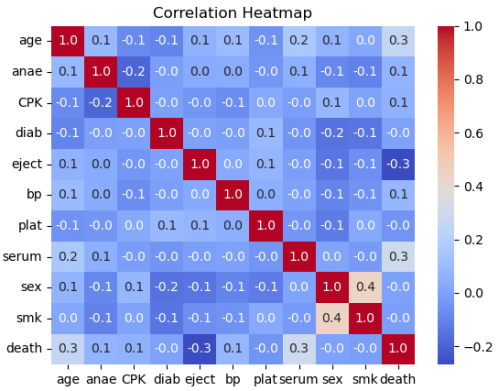

Figure 86. Heart failure imbalanced data correlation heatmap.

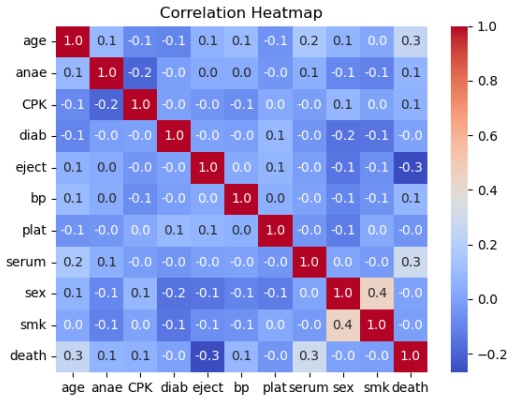

Figure 87. Heart failure balanced data correlation heatmap.

Table 28. Comparison results for the train and test dataset (heart failure).

	Train Data		Test Data			
	Iter.	CPU Time	Iter.	CPU Time	%△ in MSE	%△ in RMSE
Algorithm 3.1	2.0	2.4000×10^{-4}	2.0	2.4000×10^{-4}	10.95441	5.33490
Algorithm 3.2	2.0	3.1700×10^{-3}	2.0	3.1700×10^{-3}	10.75699	5.24115
Algorithm 3.3	2.0	1.2700×10^{-3}	2.0	1.2700×10^{-3}	10.74104	5.23357
FISTA	> 10000	1.33395	> 10000	1.33395	23.68819	11.21519
IREM	692.4	0.12118	692.4	0.12118	20.73761	9.88067
REM	692.6	0.11635	692.6	0.11635	20.73361	9.87885

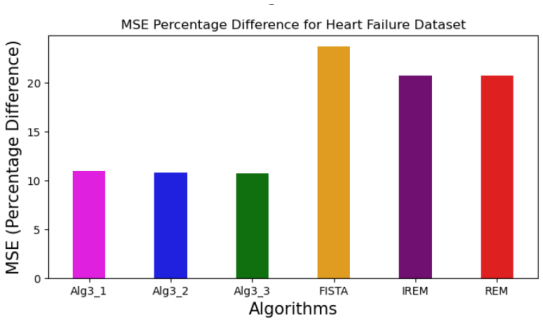

Figure 88. %△ in mean square error for heart failure train data.

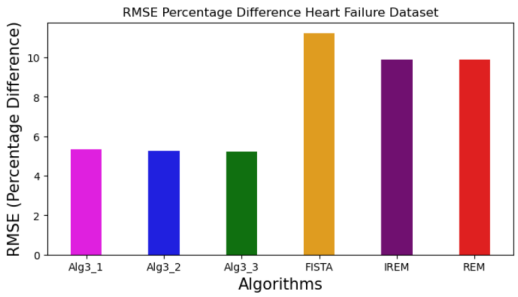

Figure 89. %△ in root mean square error for heart failure train data.

5.3.4. Experiment 5: prostate cancer classification

Prostate cancer represents a significant global public health challenge. In 2022, an estimated 1.5 million new cases and 397,000 deaths were reported worldwide, underscoring the growing burden of this disease. These figures highlight the urgent need to strengthen prostate cancer screening and treatment efforts to mitigate rising incidence and mortality.

We apply Algorithms 3.1, 3.2, and 3.3, which enhances the predictive performance of the ELM by accurately computing the optimal output weights using prostate cancer data set. The prostate cancer dataset is sourced from Kaggle and comprises diagnostic measurements derived from medical imaging and histological analysis. This dataset comprises 38 patients without prostate cancer and 62 patients diagnosed with prostate cancer, as illustrated in the bar chart below.

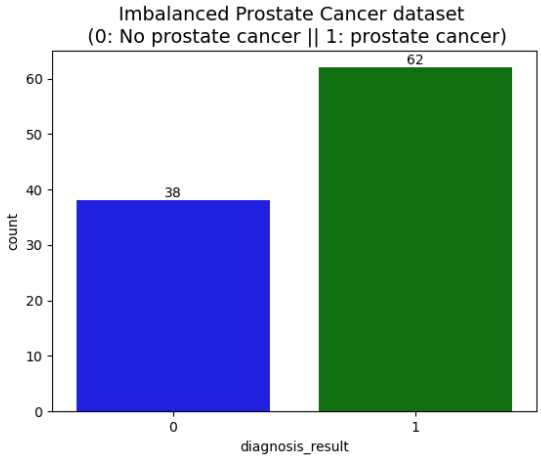

Figure 90. Bar chart of prostate cancer imbalanced data.

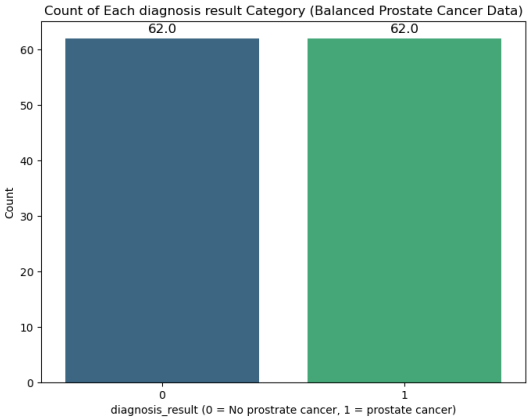

Figure 91. Bar chart of prostate cancer balanced data.

The statistics description of the features for prostate cancer dataset is shown in Table 29.

Table 29. Summary statistics of the prostate Cancer dataset.

Feature	Mean	Std Dev	Min	25%	50%	75%	Max
Radius	16.850	4.879	9	12	17	21	25
Texture	18.23	5.193	11	14	17.5	22.25	27
Perimeter	96.78	23.676	52	82.5	94	114.25	172
Area	702.880	319.711	202	476.750	644	917	1878
Smoothness	0.10273	0.01464	0.07	0.0935	0.102	0.112	0.143
Compactness	0.1267	0.06114	0.038	0.0805	0.1185	0.157	0.345
Symmetry	0.19317	0.03079	0.135	0.172	0.19	0.209	0.304
Fractal Dimension	0.06469	0.00815	0.053	0.059	0.063	0.069	0.097
Diagnosis Result	0.62	0.498	0	0	1	1	1

The relationship between each feature and prostate cancer diagnosis is shown below using some statistical tools.

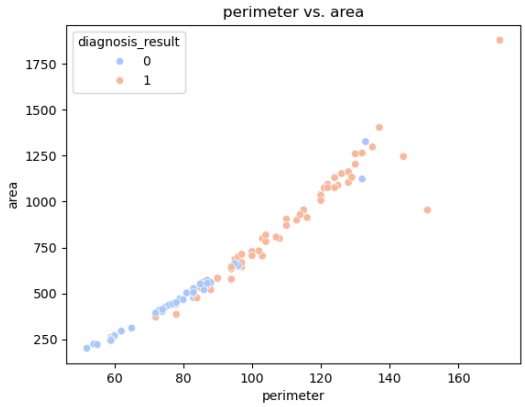

Figure 92. Scatter plot of perimeter vs area.

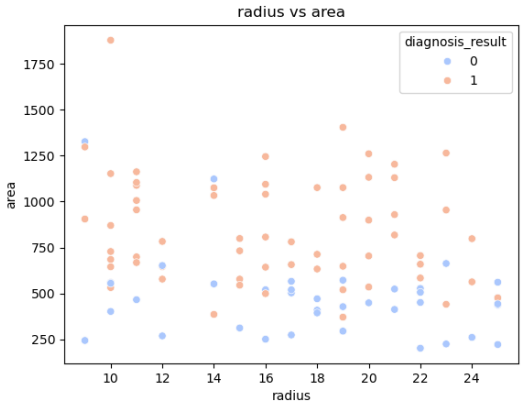

Figure 93. Scatter plot of radius vs area.

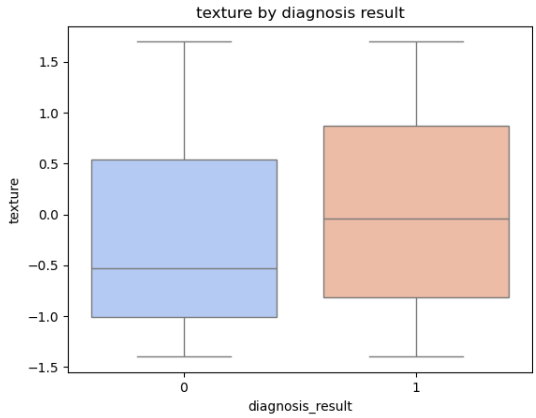

Figure 94. Box plot of texture by diagnosis result.

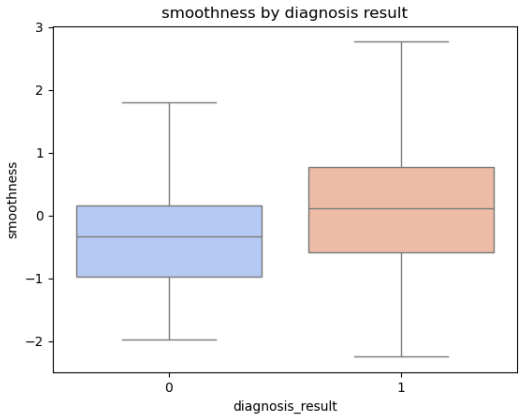

Figure 95. Box plot of smoothness by diagnosis result.

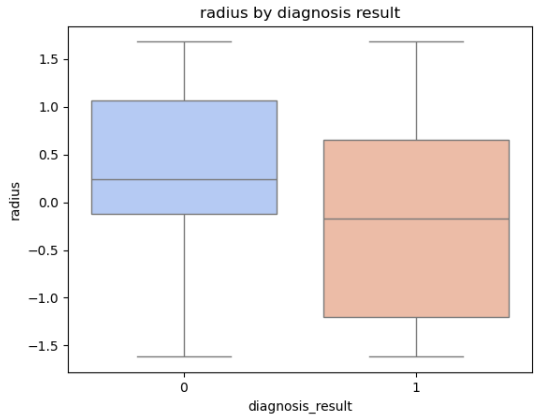

Figure 96. Box plot of radius by diagnosis result.

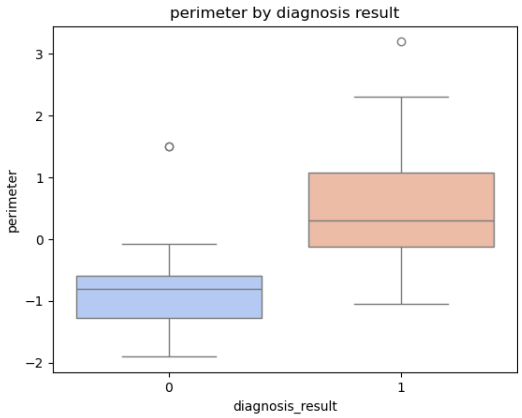

Figure 97. Box plot of p by diagnosis result.

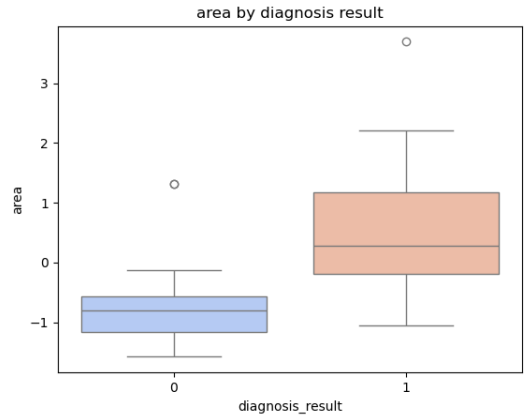

Figure 98. Box plot of area by diagnosis result.

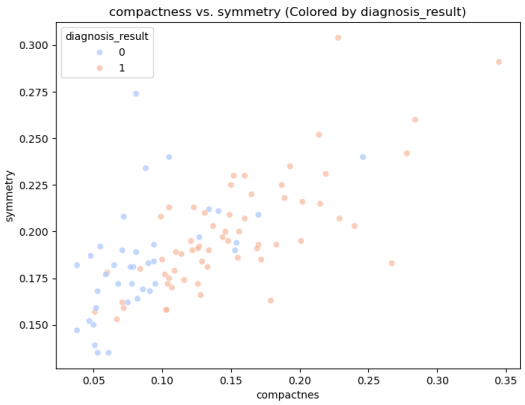

Figure 99. Scatter plot of compactness vs symmetry.

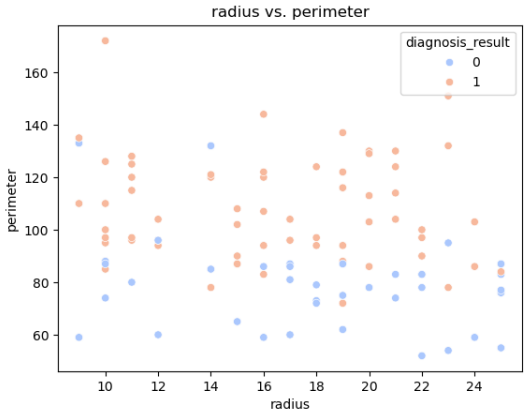

Figure 100. Scatter plot of radius vs perimeter.

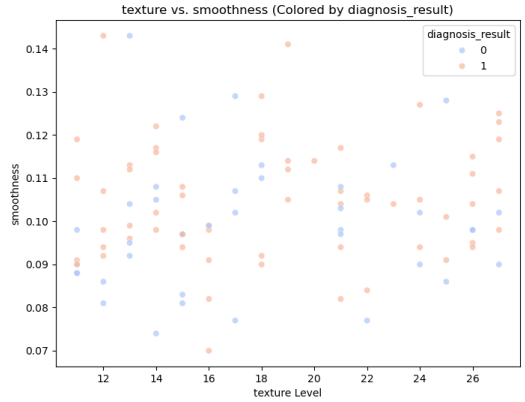

Figure 101. Scatter plot of texture vs smoothness.

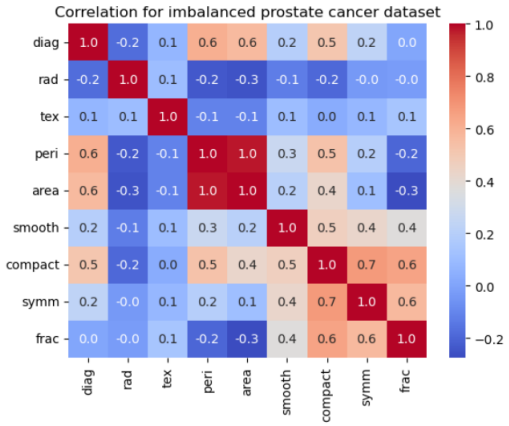

Figure 102. Prostate cancer imbalanced data correlation heatmap.

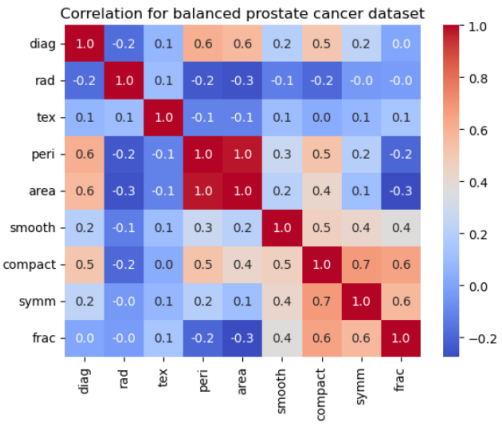

Figure 103. Prostate cancer balanced data correlation heatmap.

Table 30. Comparison results for the train and test dataset (prostate cancer).

	Train Data		Test Data			
	Iter.	CPU Time	Iter.	CPU Time	%△ in MSE	%△ in RMSE
Algorithm 3.1	2	1.9970×10^{-4}	2	1.9970×10^{-4}	3.989159	2.014878
Algorithm 3.2	2	7.6800×10^{-4}	2	7.6800×10^{-4}	3.662432	1.848297
Algorithm 3.3	2	6.4620×10^{-4}	2	6.4620×10^{-4}	3.654993	1.844507
FISTA	> 10000	1.099744	> 10000	1.099744	14.835759	7.715526
IREM	539.2	0.105547	539.2	0.105547	5.156433	2.612338
REM	539.8	0.131158	539.8	0.131158	5.153856	2.611015

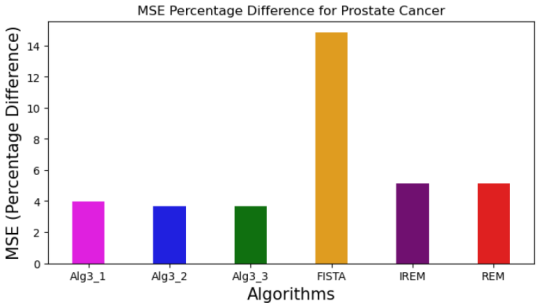

Figure 104. %△ in mean square error for prostate cancer data.

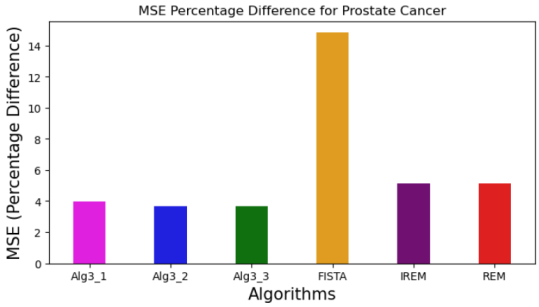

Figure 105. %△ in root mean square error for prostate cancer train data.

Remark 5.1. 1. Algorithms 3.1, 3.2, and 3.3 demonstrate strong performance and applicability for a range of problems, including those in finite- and infinite-dimensional spaces, as well as classification challenges. These algorithms have been effective in addressing problems such as heart disease classification, lung cancer classification, heart failure classification, and prostate cancer classification, utilizing the datasets outlined in Tables 22, 25, 27, and 29 and Figures 44, 45, 58, 59, 76, 77, 90, and 91.

2. Our proposed Algorithms (3.1, 3.2, and 3.3) demonstrate greater efficiency than existing algorithms such as FISTA, IREM, and REM, particularly in terms of the number of iterations and CPU time needed to achieve the desired outcomes, as shown in Tables 24, 26, 28, and 30 and Figures 56, 57, 74, 75, 88, 89, 104, and 105.
3. In addressing classification challenges using the ELM, our proposed Algorithms 3.1, 3.2, and 3.3 exhibit minimal percentage differences in RMSE and MSE, suggesting a low variance between training and testing outcomes. Conversely, FISTA, IREM, and REM demonstrate higher RMSE percentage differences, indicating greater variance and a potential risk of overfitting, indicating that our models possess strong generalization capabilities and are well suited to the data.
4. Overall, the proposed algorithms, especially Algorithm 3.2, stand out due to their speed, stability, and adaptability as optimization tools for classifying lung cancer. Their ability to quickly converge with low error variance makes them highly suitable for clinical use, outperforming traditional

methods in terms of computational efficiency and predictive reliability. Their swift convergence and low error variance render them particularly apt for clinical settings, excelling over conventional approaches in speed and generalization.

6. Conclusions

Our results of this work demonstrate that the proposed Mann-type inertial algorithms with adaptive step-size selection provide an effective and versatile framework for solving split variational inequality problems (SVIPs) arising in theoretical and data-driven settings. Motivated by the broad applicability of SVIPs in optimization, inverse problems, signal processing, and machine learning, we develop three algorithms that relax classical restrictions on inertial parameters and eliminate the need for on-line rules commonly imposed in the literature. The convergence analysis establishes weak convergence under mild assumptions, even when the inertial parameters extend beyond the traditional interval $[0, 1)$. The main limitation of the proposed methods is the cocoercivity (inverse strong monotonicity) assumption, which can later be relaxed. Extensive numerical experiments confirm the efficiency, stability, and robustness of the proposed methods. In particular, the algorithms exhibit strong performance on large scale nonlinear models and maintain reliable behavior. When applied to biomedical classification tasks through the ELM framework, the algorithms significantly enhance numerical stability and predictive accuracy, yielding improved performance on heart disease, lung cancer, heart failure, and prostate cancer datasets. Comparative studies further show that the proposed algorithms outperform existing methods, including those of Censor et al., Cholamjiak et al., Xu et al., FISTA, IREM, and REM, in terms of iteration count, CPU time, and generalization capability. Among the three methods, Algorithm 3.2 consistently demonstrates superior speed, adaptability, and convergence behavior, making it particularly suitable for high-dimensional learning and clinical decision-support applications. Overall, the combination of relaxed inertial dynamics, adaptive step-size rules, and Mann-type structures provides a powerful and flexible approach for solving SVIPs and advancing modern machine learning pipelines. The theoretical and numerical findings highlight the potential of these algorithms as reliable optimization tools for complex coupled systems and data driven diagnostic models.

Use of Generative-AI Tools Declaration:

The authors declare that no generative-AI tools were used in the research, analysis, or editing of this manuscript.

Author Contributions

All authors contributed equally to the research methodology, analysis, numerical experiments, and preparation of the manuscript.

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. A. Gibali, Y. Shehu, An efficient iterative method for finding common fixed points and variational inequalities in Hilbert spaces, *Optim.,* **68** (2019), 13–32. https://doi.org/10.1080/02331934.2018.1490417
2. G. M. Korpelevich, The extragradient method for finding saddle points and other problems, *Ekon. Mat. Metody,* **12** (1976), 747–756.
3. J. Fan, L. Liu, X. Qin, A subgradient extragradient algorithm with inertial effects for solving strongly pseudomonotone variational inequalities, *Optim.,* **69** (2020), 2199–2215. https://doi.org/10.1080/02331934.2019.1625355
4. D. V. Thong, D. V. Hieu, Inertial extragradient algorithms for strongly pseudomonotone variational inequalities, *J. Comput. Appl. Math.,* 341 (2018), 80–98. https://doi.org/10.1016/j.cam.2018.03.019
5. Y. Shehu, O. S. Iyiola, S. Reich, A modified inertial subgradient extragradient method for solving variational inequalities, *Optim. Eng.,* **23** (2021), 421–449. https://doi.org/10.1007/s11081-020-09593-w
6. X. Chang, S. Liu, Z. Deng, S. Li, An inertial subgradient extragradient algorithm with adaptive stepsizes for variational inequality problems, *Optim. Methods Softw.,* **37** (2021), 1507–1526. https://doi.org/10.1080/10556788.2021.1910946
7. L. C. Ceng, Q. H. Ansari, J. C. Yao, Relaxed extragradient methods for finding minimum-norm solutions of the split feasibility problem, *Nonlinear Anal.,* **75** (2012), 2116–2125. https://doi.org/10.1016/j.na.2011.10.012
8. L. C. Ceng, A. Petrusel, X. Qin, J. C. Yao, Two inertial subgradient extragradient algorithms for variational inequalities with fixed-point constraints, *Optim.*, **70** (2021), 1337–1358. https://doi.org/10.1080/02331934.2020.1858832
9. Y. Yao, O. S. Iyiola, Y. Shehu, Subgradient extragradient method with double inertial steps for variational inequalities, *J. Sci. Comput.,* **90** (2022), 71. https://doi.org/10.1007/s10915-021-01751-1
10. I. O. Ibiam, L. O. Madu, E. U. Ofoedu, C. E. Onyi, H. Zegeye, Hybrid Algorithm for Nonlinear Equilibrium, Variational Inequality, and Fixed Point Problems, *Basic Phys. Res.,* **7** (2017), 1–17. https://doi.org/10.5281/zenodo.4249212
11. Y. Censor, T. Elfving, A multiprojection algorithm using Bregman projections in product space, *Numer. Algor.*, **8** (1994), 117722. https://doi.org/10.1007/BF02142692
12. A. R. Eze, M. U. Wickramasinghe, T O. Alakoya, O. T. Mewomo, O. S. Iyiola, An enhanced Mann-type algorithm for split equality monotone variational inclusions and fixed point problems: Applications in image recovery, *J. Comput. Appl. Math.,* **487** (2026), 0377–0427. https://doi.org/10.1016/j.cam.2026.117722
13. C. Byrne, A unified treatment for some iterative algorithms in signal processing and image reconstruction, *Inverse Probl.,* **20** (2004), 103–120. https://doi.org/10.1088/0266-5611/20/1/006

14. Y. Censor, A. Gibali, S. Reich, Algorithms for the split variational inequality problem, *Numer. Algor.,* **59** (2012), 301–323. https://doi.org/10.1007/s11075-011-9490-5

15. O. S. Iyiola, M. U. Wickramasinghe, T. O. Alakoya, O. T. Mewomo, O. Ogunleye, A New Popov's Method for Solving Quasimonotone Variational Inequalities with Applications in Optimal Control Problems and Machine Learning Algorithms, *Netw. Spat. Econ.,* 2026. https://doi.org/10.1007/s11067-026-09738-x

16. C. Byrne, Iterative oblique projection onto convex sets and the split feasibility problem, *Inverse Probl.,* **18** (2002), 441–453. https://doi.org/10.1088/0266-5611/18/2/310

17. J. M. Hendrickx, A. Olshevsky, Matrix p-norms are NP-hard to approximate if $p \neq 1, 2, \infty$, *SIAM J. Matrix Anal. Appl.,* **31** (2010), 2802–2812. https://doi.org/10.1137/09076773X

18. G. López, V. Martín-Márquez, F. Wang, H.-K. Xu, Solving the split feasibility problem without prior knowledge of matrix norms, *Inverse Probl.,* **28** (2012), 085004. https://doi.org/10.1088/0266-5611/28/8/085004

19. J.-Z. Chen, L.-C. Ceng, Y.-Q. Qiu, Z.-R. Kong, Extragradient methods for solving split feasibility and fixed point problems, *Fixed Point Theory Appl.*, **2015** (2015), 192. https://doi.org/10.1186/s13663-015-0441-z

20. Q. L. Dong, Y. C. Tang, Y. J. Cho, Th. M. Rassias, "Optimal" choice of the step length of the projection and contraction methods for solving the split feasibility problem, *J. Glob. Optim.,* **71** (2018), 341–360 https://doi.org/10.1007/s10898-018-0628-z

21. A. Gibali, L.-W. Liu, Y.-C. Tang, Note on the modified relaxation CQ algorithm for the split feasibility problem, *Optim. Lett.,* **12** (2018), 817–830. https://doi.org/10.1007/s11590-017-1148-3

22. R. Kraikaew, S. Saejung, A simple look at the method for solving split feasibility problems in Hilbert spaces, *RACSAM,* **114** (2020), 117. https://doi.org/10.1007/s13398-020-00851-1

23. B. Qu, N. Xiu, A note on the CQ algorithm for the split feasibility problem, *Inverse Probl.,* **21** (2005), 1655–1665. https://doi.org/10.1088/0266-5611/21/5/009

24. X. Qin, L. Wang, A fixed point method for solving a split feasibility problem in Hilbert spaces, *RACSAM*, **113** (2019), 315–325. https://doi.org/10.1007/s13398-017-0476-6

25. Y. Tang, A. Gibali, Several inertial methods for solving split convex feasibilities and related problems, *RACSAM,* **114** (2020), 121. https://doi.org/10.1007/s13398-020-00857-9

26. H. K. Xu, Iterative methods for the split feasibility problem in infinite-dimensional Hilbert spaces, *Inverse Probl.,* **26** (2010), 105018. https://doi.org/10.1088/0266-5611/26/10/105018

27. Y. Dang, J. Sun, H. Xu, Inertial accelerated algorithms for solving a split feasibility problem, *J. Ind. Manag. Optim.,* **13** (2017), 1383–1394. https://doi.org/ 10.3934/jimo.2016078

28. W. Cholamjiak, Y. Shehu, J. Yao, Prediction of breast cancer through fast optimization techniques applied to machine learning, *Optim.*, **74** (2025), 3629–3657. https://doi.org/10.1080/02331934.2024.2385646

29. Q. L. Dong, H. B. Yuan, Y. J. Cho, Th. M. Rassias, Modified inertial Mann algorithm and inertial CQ-algorithm for nonexpansive mappings, *Optim. Lett.*, **12** (2018), 87–102. https://doi.org/10.1007/s11590-016-1102-9

30. D. A. Lorenz, T. Pock, An inertial forward–backward algorithm for monotone inclusions, *J. Math. Imaging Vis.,* **51** (2015), 311–325. https://doi.org/10.1007/s10851-014-0523-2

31. A. Moudafi, M. Oliny, Convergence of a splitting inertial proximal method for monotone operators, *J. Comput. Appl. Math.,* **155** (2003), 447–454. https://doi.org/10.1016/S0377-0427(02)00906-8

32. P. E. Maingé, Inertial iterative process for fixed points of certain quasi-nonexpansive mappings, *Set-Valued Anal.,* **15** (2007), 67–79. https://doi.org/10.1007/s11228-006-0027-3

33. Y. Nesterov, A method for solving the convex programming problem with convergence rate $O(1/k^2)$, *Dokl. Akad. Nauk SSSR*, **269** (1983), 543–547.

34. B. T. Polyak, Some methods of speeding up the convergence of iteration methods, *U.S.S.R Comput. Math. Phys.*, **4** (1964), 1–17. https://doi.org/10.1016/0041-5553(64)90137-5

35. T. O. Alakoya, A. Eze, O. S. Iyiola, O. Ugwu, An efficient single projection scheme for solving bilevel variational inequality problems: applications to fractional programming, *Results Math.,* **80** (2025), 214. https://doi.org/10.1007/s00025-025-02528-w

36. O. T. Mewomo, T. O. Alakoya, A. Eze, O. S. Iyiola, An inertial-like Tseng's extragradient method for solving pseudomonotone variational inequalities in reflexive Banach spaces, *Math. Meth. Appl. Sci.,* **47** (2024), 9637–9668. https://doi.org/10.1002/mma.10087

37. O. Ogunleye, O. T. Mewomo, T. O. Alakoya, O. S. Iyiola, On fixed point algorithms for solving split inverse problems with applications, *Appl. Set-Valued Anal. Optim.,* **5** (2023), 451–476. https://doi.org/10.23952/asvao.5.2023.3.08

38. M. U. Wickramasinghe, O. T. Mewomo, T. O. Alakoya, O. S. Iyiola, Mann-type approximation scheme for solving a new class of split inverse problems in Hilbert spaces, *Appl. Anal.,* **103** (2024), 1118–1148. https://doi.org/10.1080/00036811.2023.2233977

39. H.-Y. Xu, H.-Y. Lan, Y. Zhao, Y. Ye, Novel extended Halpern-type convergence algorithms for the split common fixed point problem involving α-demicontractive operators, *Optim.,* **74** (2025),343–364. http://doi.org/10.1080/02331934.2023.2256968

40. H. He, C. Ling, H. K. Xu, A relaxed projection method for split variational inequalities, *J. Optim. Theory Appl.,* **166** (2015), 213–233. https://doi.org/10.1007/s10957-014-0598-3

41. B. Tan, X. Qin, J. C. Yao, Strong convergence of self-adaptive inertial algorithms for solving split variational inclusion problems with applications, *J. Sci. Comput.,* **87** (2021), 20. https://doi.org/10.1007/s10915-021-01428-9

42. Z. Zhou, B. Tan, S. Li, Inertial algorithms with adaptive stepsizes for split variational inclusion problems and their applications to signal recovery problem, *Math. Meth. Appl. Sci.*, **47** (2024), 9431–9449. https://doi.org/ 10.1002/mma.9436

43. A. E. Ofem, A. A. Mebawondu, G. C. Ugwunnadi, P. Cholamjiak, O. K. Narain, A novel method for solving split variational inequality and fixed point problems, *Appl. Anal.,* **104** (2025), 3717–3747. https://doi.org/ 10.1080/00036811.2025.2505615

44. W. A. Kirk, Fixed point theorems for nonexpansive mappings, *Proc. Am. Math. Soc.,* **16** (1965), 763–766.

45. H. H. Bauschke, P. L. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, 2 Eds. Springer, New York, 2011.

46. Kinderlehrer, G. Stampacchia, *An Introduction to Variational Inequalities and Their Applications*, SIAM, Philadelphia, 1980. Available from: https://epubs.siam.org/doi/book/10.1137/1.9780898719451.
47. K. Goebel, W. A. Kirk, *Topics in Metric Fixed Point Theory*, Cambridge University Press, Cambridge, 1990.
48. W. Rudin, *Functional Analysis*, 2 Eds., McGraw-Hill, New York, 1991.
49. Z. Opial, Weak convergence of the sequence of successive approximations for nonexpansive mappings, *Bull. Am. Math. Soc.*, **73** (1967), 591–597.
50. J. Wang, Y. Hu, C. Li, J.-C. Yao, Linear convergence of CQ algorithms and applications in gene regulatory network inference, *Inverse Probl.,* **33** (2017), 055017. https://doi.org/10.1088/1361-6420/aa6699
51. A. Gibali, D. Mai, N. T. The Vinh, A new relaxed CQ algorithm for solving split feasibility problems in Hilbert spaces and its applications, *J. Ind. Manag. Optim.*, **15** (2019), 963–984. https://doi.org/10.3934/jimo.2018080
52. S. S. Tai, N. Pholasa, P. Cholamjiak, The modified inertial relaxed CQ algorithm for solving split feasibility problems, *J. Ind. Manag. Optim.*, **14** (2018), 1595–1615. https://doi.org/10.3934/jimo.2018023
53. S. S. Tai, N. Pholasa, P. Cholamjiak, Relaxed CQ algorithms involving the inertial technique for multiple-sets split feasibility problems, *RACSAM,* **113** (2019), 1081-–1099. https://doi.org/10.1007/s13398-018-0535-7
54. F. Alvarez, 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. https://doi.org/10.1023/A:1011253113155
55. G. N. Ogwo, C. Izuchukwu, O. T. Mewomo, Relaxed inertial methods for solving split variational inequality problems without product space formulation, *Acta Math. Sci.,* **42** (2022), 1701–1733. https://doi.org/10.1007/s10473-022-0501-5
56. W. Sun, G. Lu, Y. Jin, Z. Peng, Strong convergence theorems for split variational inequality problems in Hilbert spaces, *AIMS Math.,* **8** (2023), 27291–27308. https://doi.org/10.3934/math.20231396
57. D. R. Han, H. J. He, H. Yang, X. M. Yuan, A customized Douglas–Rachford splitting algorithm for separable convex minimization with linear constraints, *Numer. Math.,* **127** (2014), 167–200. https://doi.org/ 10.1007/s00211-013-0580-2
58. G. B. Huang, Q. Y. Zhu, C. K. Siew, Extreme learning machine: theory and applications, *Neurocomputing,* **70** (2006), 489–501. https://doi.org/10.1016/j.neucom.2005.12.126
59. T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2 Eds., Springer, New York, 2009.

AIMS Press