A Hybrid Algorithm for Monotone Variational Inequalities

Reza Rahimi Baghbadorani[a], Peyman Mohajerin Esfahani[a,b], Sergio Grammatico[a]

***Abstract*— Inspired by the adaptive Golden Ratio Algorithm (aGRAAL), we propose new methods for solving variational inequalities. We show that by selecting the momentum parameter beyond the golden ratio, the convergence speed can be improved, which motivates us to study the switching between small and large momentum parameters to accelerate convergence. We validate the performance of our proposed algorithms on several classes of variational inequality problems studied in the machine learning and control literature, including Nash equilibrium, composite minimization, Markov decision processes, and zero-sum games, and compare them to that of existing methods.**

I. Introduction

The variational inequality problem has recently emerged from several multi-agent control and machine learning problems [1], e.g., in generative adversarial networks, robust optimization, and optimal control [2]. In this paper, we consider the following variational inequality (VI) problem:

$$\text{find } x^* \in \mathcal{V} \text{ s.t. } \inf_{x\in\mathcal{V}} \langle F(x^*), x-x^*\rangle + g(x) - g(x^*) \geq 0, \quad (1)$$

where $\mathcal{V}$ is a finite-dimensional vector space. We assume that the operator F is continuous, monotone, and Lipschitz continuous, that the solution set of (1) is nonempty, and $g(x)$ is a proper lower semicontinuous (lsc) convex function. Problem (1) can be written more traditionally as follows:

$$\text{find } x^* \in \mathcal{A} \quad \text{s.t.} \quad \inf_{x\in\mathcal{A}} \langle F(x^*), x - x^*\rangle \geq 0, \qquad (2)$$

where g in (1) would be the indicator function of the set $\mathcal{A}$ in (2). The VI in (1) can be considered as a general form of problems in convex optimization. As an example, consider the composite minimization problem $\min_{x\in\mathbb{R}^n} f(x) + g(x)$, where f is a convex and smooth function and g is a proper lsc convex (and possibly nonsmooth) function. Via the KKT conditions, this problem can be written as (1) with $F = \nabla f$ and the same g in (1) [3]. Another common problem in optimization and control theory is the min-max problem. For example, consider the convex-concave saddle point problem $\min_{y\in\mathbb{R}^n} \max_{z\in\mathbb{R}^m} g_1(y) + f(y,z) - g_2(z)$, where g_1 and g_2 are proper lsc convex functions and $f(y,z)$ is a smooth convex-concave function in y and z, respectively. By using first-order optimality conditions, we can rewrite this problem as in (1) with the following variables:

$$x = \begin{pmatrix} y \\ z \end{pmatrix}, \quad F = \begin{pmatrix} \nabla_y f(y,z) \\ -\nabla_z f(y,z) \end{pmatrix}, \quad g(x) = g_1(y) + g_2(z).$$

The authors are with (a) Delft University of Technology and (b) University of Toronto. E-mail addresses: {r.rahimibaghbadorani, p.mohajerinesfahani, s.grammatico}@tudelft.nl.
This work was supported by the ERC grant TRUST-949796 and the NSERC Discovery grant RGPIN-2025-06544.

Furthermore, in many applications of reinforcement learning and game theory, we need to solve a fixed-point problem. For instance, Markov decision processes (MDPs) are a powerful modeling framework in reinforcement learning, where we should solve a fixed point problem, $Tx = x$, for some finite dimensional operator T, that is (1) with $F = \mathrm{Id} - T$ and $g(x) = 0$ [4].
Several iterative algorithms have been introduced to address VI problems (1). For comparison purposes, let us review some recent and closely related existing methods. For simplicity, let us consider the VI problem formulation in (2).
Projected Gradient descent (PGD) [5]:

$$x^{k+1} = \pi_{\mathcal{A}}(x^k - \lambda F(x^k)),$$

where λ is the stepsize. The convergence of this method is guaranteed for strongly monotone (with a strongly monotone constant μ) and Lipschitz (with a Lipschitz constant L) operator with $\lambda \in \left(0, 2\mu/L^2\right)$. This algorithm is used for equilibrium seeking in aggregative games [6] (IEEE TAC, 2021), optimal consensus and resource allocation [7] (IEEE TAC, 2023), open-loop Nash equilibrium in linear quadratic dynamic games [1], and game theoretical approach for generative adversarial network [8] (IEEE CDC, 2020).
Extragradient descent [9]:

$$y^k = \pi_{\mathcal{A}}(x^k - \lambda F(x^k)),$$
$$x^{k+1} = \pi_{\mathcal{A}}(x^k - \lambda F(y^k)),$$

where λ is the stepsize, and unlike the previous method, the convergence is guaranteed for a Lipschitz and monotone operator (with a Lipschitz constant L) with $\lambda \in (0, 1/L)$. The authors in [10] (IEEE CDC, 2014) implement this method to design robust stochastic extragradient algorithms for solving monotone VIs. Similarly, [7] (IEEE TAC, 2023) uses the extragradient method to solve the variational inequality problem in optimal consensus and resource allocation problems. The same algorithm is adopted in [11] (IEEE TPS, 2022) to design an algorithm for the variational inequality problem associated with a collaborative pricing scheme for a power-transportation coupled network.
Projected Reflected Gradient descent (PrjRef) [12]:

$$x^{k+1} = \pi_{\mathcal{A}}(x^k - \lambda F(2x^k - x^{k-1})),$$

where λ is the stepsize, and the convergence of this method is guaranteed for Lipschitz and monotone operator (with a Lipschitz constant L) with $\lambda \in \left(0, (\sqrt{2}-1)/L\right)$. Unlike the extragradient method, PrjRef needs only one projection per iteration. The authors in [13] (IEEE CDC, 2016) implement this method to design a stochastic monotone VI algorithm under some weak sharpness assumptions. Likewise, this method is adopted in [14] (IEEE CDC, 2021) to design an algorithm for solving Bayesian regression game, which is a special class of two-player general-sum Bayesian game. The authors in [15] (ECC, 2021) also use the projected-reflected method to design an algorithm for stochastic generalized Nash equilibrium problems.

Golden RAtio ALgorithm (GRAAL) [16]:

$$y^k = (1-\beta)x^k + \beta y^{k-1},$$
$$x^{k+1} = \pi_{\mathcal{A}}(y^k - \lambda F(x^k)),$$

where λ is the stepsize and $\beta \in \big(0, (\sqrt{5}-1)/2\big]$. The convergence of this method is guaranteed for Lipschitz and monotone operator (with a Lipschitz constant L) with $\lambda \in (0, 1/(2\beta L))$, and it requires one projection per iteration. The stepsize in this method can be chosen adaptively as follows, leading to the Adaptive Golden RAtio ALgorithm (aGRAAL) [16]:

$$\lambda_k = \min\left\{(\beta+\beta^2)\lambda_{k-1}, \frac{\|x^k - x^{k-1}\|^2}{4\beta^2\lambda_{k-2}\|F(x^k) - F(x^{k-1})\|^2}\right\}.$$

This algorithm is applied in [17] (IEEE TAC, 2021) and [18] (IEEE CDC, 2021) to design a method for (stochastic) generalized Nash equilibrium in monotone games. The authors in [19] also use this method in a stochastic portfolio allocation game as a case study for Nash equilibrium seeking in quadratic-bilinear Wasserstein distributionally robust games.

Contribution. In this paper, we propose two algorithms for solving the monotone variational inequality problem in (1) that do not require knowledge of a global Lipschitz constant. Both algorithms are inspired by stability in switched and hybrid systems, where a switched system is asymptotically stable if each subsystem has a strictly decreasing Lyapunov function and switching does not increase it [20]. Then, based on this context and the application of hybrid system stability in optimization and extremum seeking [21]–[23], we switch between two algorithms adopted for solving VIs. Our technical contribution is to show convergence for the first algorithm and the ergodic $\mathcal{O}(k^{-1})$ convergence rate for the second one. The proposed algorithms reduce dependency on the negative momentum term, previously used in [16] to ensure boundedness and convergence of iterates, by increasing the momentum parameter in some iterations with the potential to switch the momentum parameter between small (used in aGRAAL) and large values. Using a large momentum parameter in our proposed algorithms (Algorithms 1 and 2) brings the iterations closer to the most recent one, allowing us to estimate the local Lipschitz constant of F more accurately and reducing the frequent use of the negative momentum term which repetitively affects convergence speed [24]. Briefly speaking, if F is a Lipschitz and monotone operator, our proposed methods switch between PGD (method without the negative momentum term) and aGRAAL (method with the negative momentum term) based on certain conditions, along with an adaptive stepsize. Finally, we provide several numerical experiments in which the proposed algorithms consistently outperform the existing state of the art. We note that our method rarely requires additional computations for operator evaluation and projection. However, in the worst case, we may need to perform these computations twice compared to the aGRAAL method.

Notation. Let $\mathcal{V}$ be the finite-dimensional real vector space with the standard inner product $\langle \cdot, \cdot \rangle$ and ℓ_p-norm $\|\cdot\|_p$ (by $\|\cdot\|$, we mean the Euclidean standard 2-norm). We also denote the $\pi_{\mathcal{A}}$ for the metric projection onto set $\mathcal{A}$ ($\pi_{\mathcal{A}}(x) = \arg\min_{y\in\mathcal{A}} \|x-y\|$), $\delta_{\mathcal{A}}$ the indicator function of set $\mathcal{A}$, $\operatorname{dist}(x, \mathcal{A})$ the distance from x to set $\mathcal{A}$ ($\operatorname{dist}(x,\mathcal{A}) = \|\pi_{\mathcal{A}}(x) - x\|$), and $\mathbb{B}(\tilde{x}, r)$ a closed ball with center $\tilde{x}$ and radius $r > 0$. The operator F is L-Lipschitz, if there is $L > 0$ such that for all $x, y \in \mathcal{V}$ we have $\|F(x) - F(y)\| \le L\|x-y\|$. Furthermore, F is locally Lipschitz, if it is Lipschitz over any compact set of its domain. The operator F is monotone if $\langle F(x) - F(y), x - y\rangle \ge 0$ for all $x, y \in \mathcal{V}$ and it is called strongly monotone with constant $\mu > 0$ if $\langle F(x) - F(y), x-y\rangle \ge \mu\|x-y\|^2$ for all $x, y \in \mathcal{V}$. The prox operator of a function $g\colon \mathcal{V} \to \mathbb{R}$ is defined as $\operatorname{prox}_g(x) = \arg\min_u\{g(u) + \|u-x\|^2/2\}$. A function is "prox-friendly" if the prox operator is available (computationally or explicitly). The following equations are useful and commonly used in the proofs [25]:

$$y = \operatorname{prox}_g x \iff \langle y - x, z - y\rangle \ge g(y) - g(z), \quad \forall z \in \mathcal{V} \tag{3a}$$
$$\|ax + (1-a)y\|^2 = a\|x\|^2 + (1-a)\|y\|^2 - a(1-a)\|x-y\|^2. \quad \forall x, y \in \mathcal{V}, \forall a \in \mathbb{R} \tag{3b}$$

II. Preliminaries and First Algorithm

In this section, we first present the main theorem, which helps establish the boundedness and convergence of the iterations with a variable momentum parameter for the

Algorithm 1 Adaptive algorithm for VI (Method 1)

Require: Choose x^0, x^1, $\lambda_0 > 0$, $\phi \in (1, \frac{1+\sqrt{5}}{2}]$, $\theta_0 = 1$, $\rho = \dfrac{1}{\phi} + \dfrac{1}{\phi^2}$, flg $= 0$, $\bar{k} = 1$

For $k = 1, 2, \ldots$ **do**

Find the stepsize:
$$\lambda_k = \min\left\{\rho\lambda_{k-1}, \frac{\phi\theta_{k-1}}{4\lambda_{k-1}} \frac{\|x^k - x^{k-1}\|^2}{\|F(x^k) - F(x^{k-1})\|^2}\right\}$$

if $(J(x^k) - J(x^{k-1}) > 0 \;\wedge\;$ flg $= 1) \;\vee\; \min\{J_i\}_{i=0}^{k-1} < J_k + 1/\bar{k}$ **then**

$$\bar{x}^k = \frac{(\phi-1)x^k + \bar{x}^{k-1}}{\phi}, \text{ flg} = 0$$

else

$$\bar{x}^k = x^k, \text{ flg} = 1, \; \bar{k} = \bar{k} + 1$$

end if

Update the next iteration:
$$x^{k+1} = \operatorname{prox}_{\lambda_k g}(\bar{x}^k - \lambda_k F(x^k))$$

Update: $$\theta_k = \frac{\phi\lambda_k}{\lambda_{k-1}}$$

Residual computation: $J_{k+1} = x^k - \operatorname{prox}_g(x^k - F(x^k))$

Algorithm 2 Adaptive algorithm for VI (Method 2)

Require: Choose x^0, x^1, $\lambda_0 > 0$, $\alpha \in (1, \frac{1+\sqrt{5}}{2}]$, $\theta_0 = 1$, $\rho = \dfrac{1}{\alpha} + \dfrac{1}{\alpha^2}$, $\bar{\phi} \gg \frac{1+\sqrt{5}}{2}$, $\text{sum}_0^1 = 0$, $\text{sum}_0^2 = 0$, flg = 1, $\phi_0 = \bar{\phi}$.

For $k = 1, 2, \dots$ **do**
  Find the stepsize:
$$\lambda_k = \min\left\{\rho\lambda_{k-1}, \frac{\alpha\theta_{k-1}}{4\lambda_{k-1}} \frac{\|x^k - x^{k-1}\|^2}{\|F(x^k) - F(x^{k-1})\|^2}\right\}$$
  $\bar{x}^k = \dfrac{(\phi_k - 1)x^k + \bar{x}^{k-1}}{\phi_k}$
  Update the next iteration:
$$x^{k+1} = \text{prox}_{\lambda_k g}(\bar{x}^k - \lambda_k F(x^k))$$
  Update: $\theta_k = \dfrac{\alpha\lambda_k}{\lambda_{k-1}}$
  compute the following summations with $\phi_{k+1} = \bar{\phi}$:
    $\text{sum}_{k+1}^1 = \text{sum}_k^1$ + (13)
    $\text{sum}_{k+1}^2 = \text{sum}_k^2$ + (14)
  if $(\text{sum}_{k+1}^1 \le 0 \ \wedge\ \text{flg} = 1) \ \vee\ (\text{sum}_{k+1}^2 \le 0 \ \wedge\ \text{flg} = 0)$ **then**
    $\phi_{k+1} = \bar{\phi}$, $\text{flg} = 1$
  else
    if $\text{flg} = 1$ **then**
      $x^{k+1} = x^k$, $x^k = x^{k-1}$, $\bar{x}^k = \bar{x}^{k-1}$
      $\phi_{k+1} = \alpha$, $\theta_k = \theta_{k-1}$, $\lambda_k = \lambda_{k-1}$
      $\text{sum}_{k+1}^1 = 0$, $\text{sum}_{k+1}^2 = 0$, $\text{flg} = 0$
    else
      $\phi_{k+1} = \alpha$
      $\text{sum}_{k+1}^2 = \text{sum}_k^2$ + ((14) with $\phi_{k+1} = \alpha$)
      $\text{sum}_{k+1}^1 = 0$
    end if
  end if

algorithms whose general forms are given in Algorithms 1 and 2. We then describe our first algorithm, which follows the PGD and aGRAAL frameworks, differing only in the choice of the momentum parameter determined by conditions ensuring a sufficient decrease in the error bound.

Before proceeding with the theorem, let us define the merit function $\Psi(x, y) := \langle F(x), y - x\rangle + g(y) - g(x)$, which is convex with respect to y. It can be easily seen that (1) is equivalent to finding $x^* \in \mathcal{V}$ such that $\Psi(x^*, x) \ge 0$, $\forall x \in \mathcal{V}$.

Theorem II.1 (Variable momentum in aGRAAL)**.** *Let $F\colon \operatorname{dom} g \to \mathcal{V}$ be locally Lipschitz and monotone operator. Then $\left(x^k\right)_{k\in\mathbb{N}}$ and $\left(\bar{x}^k\right)_{k\in\mathbb{N}}$, generated by Algorithms 1-2, satisfy the following inequality:*

$$\begin{aligned}
&\frac{\phi_{k+1}}{\phi_{k+1}-1}\|\bar{x}^{k+1} - x\|^2 + \frac{\theta_k}{2}\|x^{k+1} - x^k\|^2 + 2\lambda_k\Psi(x, x^k) \\
&\le \frac{\phi_{k+1}}{\phi_{k+1}-1}\|\bar{x}^k - x\|^2 + \frac{\theta_{k-1}}{2}\|x^k - x^{k-1}\|^2 \\
&- \frac{\lambda_k}{\lambda_{k-1}}\phi_k\|x^k - \bar{x}^k\|^2 + \big(\frac{\lambda_k}{\lambda_{k-1}}\phi_k - 1 - \frac{1}{\phi_{k+1}}\big)\|x^{k+1} - \bar{x}^k\|^2 \\
&- \big(\frac{\lambda_k}{\lambda_{k-1}}\phi_k - \theta_k\big)\|x^{k+1} - x^k\|^2.
\end{aligned} \tag{4}$$

Proof: Let $x \in \mathcal{V}$ be arbitrary. Now consider Algorithms 1 and 2, where x^k and $\bar{x}^k$ are updated as follows:
$$\bar{x}^k = \frac{(\phi_k - 1)x^k + \bar{x}^{k-1}}{\phi_k}, \quad x^{k+1} = \text{prox}_{\lambda_k g}(\bar{x}^k - \lambda_k F(x^k)).$$
Then by using (3a), we have
$$\begin{aligned}
\langle x^{k+1} - \bar{x}^k + \lambda_k F(x^k), x - x^{k+1}\rangle &\ge \\
\lambda_k\left(g(x^{k+1}) - g(x)\right), \quad & (5)\\
\langle x^k - \bar{x}^{k-1} + \lambda_{k-1}F(x^{k-1}), x^{k+1} - x^k\rangle &\ge \\
\lambda_{k-1}\left(g(x^k) - g(x^{k+1})\right). \quad & (6)
\end{aligned}$$
Multiplying (6) by $\frac{\lambda_k}{\lambda_{k-1}} \ge 0$ and using that $x^k - \bar{x}^{k-1} = \phi_k(x^k - \bar{x}^k)$, we obtain
$$\begin{aligned}
&\langle \frac{\lambda_k}{\lambda_{k-1}}\phi_k(x^k - \bar{x}^k) + \lambda_k F(x^{k-1}), x^{k+1} - x^k\rangle \\
&\qquad \ge \lambda_k\left(g(x^k) - g(x^{k+1})\right).
\end{aligned} \tag{7}$$
The summation of (5) and (7) gives us
$$\begin{aligned}
&\langle x^{k+1} - \bar{x}^k, x - x^{k+1}\rangle + \frac{\lambda_k\phi_k}{\lambda_{k-1}}\langle x^k - \bar{x}^k, x^{k+1} - x^k\rangle \\
&+ \lambda_k\langle F(x^k) - F(x^{k-1}), x^k - x^{k+1}\rangle \ge \\
&\lambda_k\langle F(x^k), x^k - x\rangle + \lambda_k\left(g(x^k) - g(x)\right) \ge \\
&\lambda_k\Big[\langle F(x), x^k - x\rangle + g(x^k) - g(x)\Big] = \lambda_k\Psi(x, x^k).
\end{aligned} \tag{8}$$
Expressing the first two terms in (8) through norms leads to
$$\begin{aligned}
\|x^{k+1} - x\|^2 &\le \|\bar{x}^k - x\|^2 - \|x^{k+1} - \bar{x}^k\|^2 \\
&\quad + 2\lambda_k\langle F(x^k) - F(x^{k-1}), x^k - x^{k+1}\rangle \\
&+ \frac{\lambda_k}{\lambda_{k-1}}\phi_k\left(\|x^{k+1} - \bar{x}^k\|^2 - \|x^{k+1} - x^k\|^2 - \|x^k - \bar{x}^k\|^2\right) \\
&\quad - 2\lambda_k\Psi(x, x^k).
\end{aligned} \tag{9}$$
Similarly to (3b), we have
$$\begin{aligned}
\|x^{k+1} - x\|^2 &= \frac{\phi_{k+1}}{\phi_{k+1}-1}\|\bar{x}^{k+1} - x\|^2 \\
&- \frac{1}{\phi_{k+1}-1}\|\bar{x}^k - x\|^2 + \frac{1}{\phi_{k+1}}\|x^{k+1} - \bar{x}^k\|^2.
\end{aligned} \tag{10}$$
By combining this with (9), we obtain
$$\begin{aligned}
&\frac{\phi_{k+1}}{\phi_{k+1}-1}\|\bar{x}^{k+1} - x\|^2 \le \frac{\phi_{k+1}}{\phi_{k+1}-1}\|\bar{x}^k - x\|^2 + \\
&\big(\frac{\lambda_k}{\lambda_{k-1}}\phi_k - 1 - \frac{1}{\phi_{k+1}}\big)\|x^{k+1} - \bar{x}^k\|^2 - 2\lambda_k\Psi(x, x^k) \\
&- \frac{\lambda_k}{\lambda_{k-1}}\phi_k\left(\|x^{k+1} - x^k\|^2 + \|x^k - \bar{x}^k\|^2\right) \\
&+ 2\lambda_k\langle F(x^k) - F(x^{k-1}), x^k - x^{k+1}\rangle.
\end{aligned} \tag{11}$$
Using the stepsize update rule and Holder's inequality, the last term on the right-hand side of (11) can be upper bounded by
$$\begin{aligned}
2\lambda_k\langle F(x^k) - F(x^{k-1}), x^k - x^{k+1}\rangle &\le \\
2\lambda_k\|F(x^k) - F(x^{k-1})\|\|x^k - x^{k+1}\| &\le \\
\sqrt{\theta_k\theta_{k-1}}\|x^k - x^{k-1}\|\|x^k - x^{k+1}\| &\le
\end{aligned}$$

$$\frac{\theta_k}{2}\|x^{k+1}-x^k\|^2+\frac{\theta_{k-1}}{2}\|x^k-x^{k-1}\|^2. \quad (12)$$

Finally, by applying the obtained estimate to (11), we conclude the result of the theorem.

By controlling the right-hand-side of (4), we can prove the boundedness and convergence of the sequence $\left(x^k\right)_{k\in\mathbb{N}}$. Next, we aim to maintain the negativity of the last three terms of the right-hand-side of (4) while ensuring that ϕ_k attains a sufficiently large value which makes $\bar{x}^k$ closer to the current iterate x^k instead of $\bar{x}^{k-1}$. Subsequently, we elaborate on two methods devised for achieving this objective.

Algorithm 1. In this method, we alternate between the algorithm with a small $\phi \in \left(1, \frac{1+\sqrt{5}}{2}\right]$ and the one without momentum (or equivalently $\phi = \infty$) based on the residual evaluation, used as a measure of performance in VI [25, Proposition 1.5.8]. The use of small ϕ ultimately leads to the convergence of the residual to zero due to the negativity of the three rightmost terms in (4) [16, Theorem 2]. We initiate the algorithm without the momentum term, and by computing the residual, $J_k = \|x^k - \mathrm{prox}_g(x^k - F(x^k))\|$, in each iteration, we continue without ϕ if the residual is decreasing. Conversely, if the residual is not decreasing, we switch ϕ to the small value until the residual becomes smaller than the minimum residual achieved so far plus $\frac{1}{\bar{k}}$, where $\bar{k}$ denotes the number of times switching has occurred so far. It is noteworthy that the use of a small ϕ may result in non-monotone changes in the residual, and we refrain from altering ϕ until the residual becomes smaller than the minimum residual achieved so far plus the non-summable term. Figure 1 illustrates how Algorithm 1 operates: The alteration of ϕ is observed when the residual decreases sufficiently, ensuring convergence due to the fact that $\sum_k 1/k = \infty$.

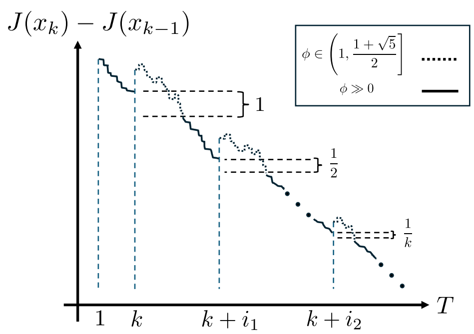


Fig. 1: Residual variation induced by Algorithm 1.

III. An efficient switching algorithm

In this section, we analyze the convergence of Algorithm 2 for solving (1), which follows the aGRAAL method. Differently from aGRAAL, the momentum parameter is not fixed, and in fact, in our numerical experience, it has a large value in many iterations, which supports the acceleration of the algorithms. Now, by employing (4), we use a simple analysis to control the right-hand side of (4) and aim to maintain the negativity of the right-hand side while ensuring that ϕ_k attains a sufficiently large value.

Algorithm 2. The algorithm is initiated with a large value for ϕ_k, and the summation of (13) is computed after each iteration (with large value of ϕ_{k+1}). If the resulting summation is negative, the algorithm proceeds with the initial large value of ϕ_k. Conversely, if the summation is not negative, the algorithm is reset (by restarting, we mean that x^{k+1} is generated by large ϕ_k and other parameters with indices k are not considered as a new iteration and variables, lines 11-13 of Algorithm 2), and ϕ_k is chosen from the interval $\left(1, \frac{1+\sqrt{5}}{2}\right]$.

$$\begin{aligned}
&\frac{\theta_{k-1}}{2}\|x^k-x^{k-1}\|^2-\frac{\lambda_k}{\lambda_{k-1}}\phi_k\|x^k-\bar{x}^k\|^2\\
&+\big(\frac{\lambda_k}{\lambda_{k-1}}\phi_k-1-\frac{1}{\phi_{k+1}}\big)\|x^{k+1}-\bar{x}^k\|^2\\
&-\big(\frac{\lambda_k}{\lambda_{k-1}}\phi_k-\theta_k\big)\|x^{k+1}-x^k\|^2-\frac{\theta_k}{2}\|x^{k+1}-x^k\|^2. \quad (13)
\end{aligned}$$

After the restarting, the following equation is examined in each iteration (with large value of ϕ_{k+1})

$$\begin{aligned}
-\frac{\lambda_k\phi_k}{\lambda_{k-1}}\|x^k-\bar{x}^k\|^2+\big(\frac{\lambda_k\phi_k}{\lambda_{k-1}}-1-\frac{1}{\phi_{k+1}}\big)\|x^{k+1}-\bar{x}^k\|^2\\
-\big(\frac{\lambda_k\phi_k}{\lambda_{k-1}}-\theta_k\big)\|x^{k+1}-x^k\|^2. \quad (14)
\end{aligned}$$

If the computed summation is negative, then the algorithm employs the large ϕ once more for the next iterations; conversely, if the summation is not negative, the algorithm persists with a small value of ϕ. In this context, three scenarios are contemplated for Algorithm 2

(i) **Always negative summation:** By telescoping (4) the summation of (13) is always negative; therefore, x^k are bounded and $x^k \to x^*$ if $k \to \infty$.

(ii) **Always positive summation:** We always have $\phi \in (1, \frac{1+\sqrt{5}}{2}]$, thus we obtain the same algorithm as in [16, Algorithm 1].

(iii) **Switching between small and large ϕ :** If ϕ_k is small and by modifying ϕ_{k+1} to a larger value, (14) becomes negative, we adjust ϕ_{k+1} to a larger value in the subsequent step. Then, the inequality $\frac{\phi_{k+1}}{\phi_{k+1}-1} \le \frac{\phi_k}{\phi_k-1}$ holds, and (4) in two steps is as follows:

$$\begin{aligned}
&\big(\frac{\phi_k}{\phi_k-1}-\frac{\phi_{k+1}}{\phi_{k+1}-1}\big)\|\bar{x}^k-x\|^2+\frac{\phi_{k+1}}{\phi_{k+1}-1}\|\bar{x}^k-x\|^2\\
&+\frac{\theta_{k-1}}{2}\|x^k-x^{k-1}\|^2+2\lambda_{k-1}\Psi(x,x^{k-1})\\
&\le\frac{\phi_k}{\phi_k-1}\|\bar{x}^{k-1}-x\|^2+\frac{\theta_{k-2}}{2}\|x^{k-1}-x^{k-2}\|^2\\
&+\big(\frac{\lambda_{k-1}}{\lambda_{k-2}}\phi_{k-1}-1-\frac{1}{\phi_k}\big)\|x^k-\bar{x}^{k-1}\|^2\\
&-\frac{\lambda_{k-1}}{\lambda_{k-2}}\phi_{k-1}\|x^{k-1}-\bar{x}^{k-1}\|^2\\
&-\big(\frac{\lambda_{k-1}}{\lambda_{k-2}}\phi_{k-1}-\theta_{k-1}\big)\|x^k-x^{k-1}\|^2. \quad (15)
\end{aligned}$$

$$
\begin{aligned}
&\frac{\phi_{k+1}}{\phi_{k+1}-1}\|\bar{x}^{k+1}-x\|^2+\frac{\theta_k}{2}\|x^{k+1}-x^k\|^2+2\lambda_k\Psi(x,x^k)\\
&\leq\frac{\phi_{k+1}}{\phi_{k+1}-1}\|\bar{x}^k-x\|^2+\frac{\theta_{k-1}}{2}\|x^k-x^{k-1}\|^2\\
&-\frac{\lambda_k}{\lambda_{k-1}}\phi_k\|x^k-\bar{x}^k\|^2-\big(\frac{\lambda_k}{\lambda_{k-1}}\phi_k-\theta_k\big)\|x^{k+1}-x^k\|^2\\
&+\big(\frac{\lambda_k}{\lambda_{k-1}}\phi_k-1-\frac{1}{\phi_{k+1}}\big)\|x^{k+1}-\bar{x}^k\|^2, \qquad (16)
\end{aligned}
$$

where in the first line of (15) we add and subtract $\frac{\phi_{k+1}}{\phi_{k+1}-1}\|\bar{x}^k-x\|^2$. However, if ϕ_k is large and the summations of (13) is not negative, the algorithm should be reset with a smaller ϕ_k.

Let us assume we switch to the large ϕ in the k^{th} iteration and after $i+1$ steps, we change ϕ to a small value. In this case, the condition "$\text{sum}^1_{k+1}\leq 0$" in Algorithm 2 (negative summation of (13)) ensures that $\|\bar{x}^k-x\|^2\geq\|\bar{x}^{k+i}-x\|^2$ while $\frac{\phi_k}{\phi_k-1}-\frac{\phi_{k+1}}{\phi_{k+1}-1}=-(\frac{\phi_{k+i}}{\phi_{k+i}-1}-\frac{\phi_{k+i+1}}{\phi_{k+i+1}-1})$. Therefore, (4) in two steps can be expressed as follows:

$$
\begin{aligned}
&(\frac{\phi_{k+i}}{\phi_{k+i}-1}-\frac{\phi_{k+i+1}}{\phi_{k+i+1}-1})\|\bar{x}^{k+i}-x\|^2\\
&+\frac{\phi_{k+i+1}}{\phi_{k+i+1}-1}\|\bar{x}^{k+i}-x\|^2+\frac{\theta_{k+i-1}}{2}\|x^k-x^{k+i-1}\|^2\\
&+2\lambda_{k+i-1}\Psi(x,x^{k+i-1})\leq\frac{\phi_{k+i}}{\phi_{k+i}-1}\|\bar{x}^{k+i-1}-x\|^2\\
&+\frac{\theta_{k+i-2}}{2}\|x^{k+i-1}-x^{k+i-2}\|^2\\
&-\frac{\lambda_{k+i-1}}{\lambda_{k+i-2}}\phi_{k+i-1}\|x^{k+i-1}-\bar{x}^{k+i-1}\|^2\\
&+\big(\frac{\lambda_{k+i-1}}{\lambda_{k+i-2}}\phi_{k+i-1}-1-\frac{1}{\phi_{k+i}}\big)\|x^{k+i}-\bar{x}^{k+i-1}\|^2\\
&-\big(\frac{\lambda_{k+i-1}}{\lambda_{k+i-2}}\phi_{k+i-1}-\theta_{k+i-1}\big)\|x^{k+i}-x^{k+i-1}\|^2. \qquad (17)
\end{aligned}
$$

$$
\begin{aligned}
&\frac{\phi_{k+i+1}}{\phi_{k+i+1}-1}\|\bar{x}^{k+i+1}-x\|^2+\frac{\theta_{k+i}}{2}\|x^{k+i+1}-x^{k+i}\|^2\\
&+2\lambda_{k+i}\Psi(x,x^{k+i})\leq\frac{\phi_{k+i+1}}{\phi_{k+i+1}-1}\|\bar{x}^{k+i}-x\|^2\\
&+\frac{\theta_{k+i-1}}{2}\|x^{k+i}-x^{k+i-1}\|^2\\
&-\frac{\lambda_{k+i}}{\lambda_{k+i-1}}\phi_{k+i}\|x^{k+i}-\bar{x}^{k+i}\|^2\\
&+\big(\frac{\lambda_{k+i}}{\lambda_{k+i-1}}\phi_{k+i}-1-\frac{1}{\phi_{k+i+1}}\big)\|x^{k+i+1}-\bar{x}^{k+i}\|^2\\
&-\big(\frac{\lambda_{k+i}}{\lambda_{k+i-1}}\phi_{k+i}-\theta_{k+i}\big)\|x^{k+i+1}-x^{k+i}\|^2, \qquad (18)
\end{aligned}
$$

where in the first line of (17) we add and subtract $\frac{\phi_{k+i+1}}{\phi_{k+i+1}-1}\|\bar{x}^{k+i}-x\|^2$. Then by telescoping (4) (in both cases, whether switching from a small ϕ to a large one (15) and (16), or switching from a large ϕ to a small one (17) and (18)), we drive (19). More precisely, the conditions in line 7 of Algorithm 2 ensure that, by telescoping (4), we obtain similar terms on the right and left-hand side of successive lines of (4) (e.g., the leftmost terms in (15) and (17) can be removed by telescoping the inequalities, and we have similar terms on the right and left-hand sides of two successive inequalities) which allows us to *point-wise* remove the similar terms and obtain the following inequality:

$$
\begin{aligned}
&\frac{\phi_T}{\phi_T-1}\|\bar{x}^T-x\|^2+\frac{\theta_{T-1}}{2}\|x^T-x^{T-1}\|^2\\
&\quad+2\sum_{i=1}^{T}\lambda_i\Psi(x,x^i)\leq\frac{\phi_2}{\phi_2-1}\|\bar{x}^1-x\|^2\\
&\quad+\frac{\theta_0}{2}\|x^1-x^0\|^2+D, \qquad (19)
\end{aligned}
$$

where D is a non-positive constant which is summation of the three negative rightmost terms in (4) for T iterations. Note that T in (19) is not exactly the number of projections or operator evaluations in Algorithm 2. In more detail, if we are in case (i) and always continue with large ϕ, then the number of projections and operator evaluations is exactly T. In the worst-case scenario, we have case (ii), where the current sequence should regenerate with small ϕ. In this situation, the number of projections and operator evaluations is $2T$. Finally, if the sequence is generated by switching between large and small ϕ (case (iii)), then the number of projections and operator evaluations is between T and $2T$. It is also worth noting that, in practice, the number of projections and operator evaluations is close to T (see Section IV).

Similarly to [16], we can prove the ergodic convergence rate based on (19). The following lemmas indicate the convergence properties of Algorithm 2.

Lemma III.1 (Ergodic convergence)**.** *Let X_k be the ergodic sequence $X_k=\sum_{i=1}^k\lambda_i x^i/\sum_{i=1}^k\lambda_i$ and $e_r(y)=\max_{x\in\mathcal{U}}\Psi(x,y)\quad\forall y\in\mathcal{V}$, where $\mathcal{U}=\operatorname{dom} g\cap\mathbb{B}(\hat{x},r)$ and $\hat{x}\in\operatorname{dom} g$. Then, we obtain the $\mathcal{O}(k^{-1})$ convergence rate for the ergodic sequence X_k, where $M>0$ is some constant that dominates the right-hand side of* (19) *for all $x\in\mathcal{U}$, in particular $\sum_{i=1}\lambda_i\Psi(x,x^i)\leq M$. More precisely we have*

$$
e_r(X_k)=\max_{x\in\mathcal{U}}\Psi(x,X_k)\leq\frac{M}{k}.
$$

Proof: See Appendix.

Algorithm 2 consists of three parts. Line 8 handles either the case of (i) or modifies ϕ_{k+1} to a large value (case (iii)). In lines 11-13, the algorithm adjusts ϕ_{k+1} to a small value (cases (ii) or (iii)). Note that in lines 11–13, the generated x^{k+1} is removed, and we go one step back to reset the setup. We then use x^k, x^{k-1}, and $\bar{x}^{k-1}$ with an updated setup to generate a new x^{k+1}. Finally, lines 15-17 correspond to case (ii), where we continue by updating sum^2_{k+1} with a small ϕ_{k+1} if sum^2_{k+1} is non-negative with a large ϕ_k and $\text{flg}=0$.

IV. Numerical simulations

We demonstrate the performance of Algorithm 1 and 2 on several classes of VI problems studied in the literature: (1) Nash–Cournot equilibrium, (2) feasibility problem (finding a point in the intersection of balls) (3) sparse logistic regression

(4) skew symmetric operator (5) Two-player Zero Sum Game (6) Markov decision processes (7) strongly monotone operator with equality constraint and (8) VI problem with non-monotone operator. To evaluate the performance of our proposed algorithms, we compare their residual, used as a measure of performance in VI, with the residuals of the following methods from the literature throughout the iterations: (i) Projected Gradient descent (PrGD), (ii) projected reflected Gradient descent (PrRefGD), and (iii) adaptive Golden ratio (aGRAAL), a relatively recent method for monotone variational inequality and the closest in spirit to our proposed method. To be more fair, we plot the residual (y-axis) against the fixed number of *operator evaluation* (x-axis) in all our figures. We set $\phi = 1.5$ and $\lambda_0 = 1$ in Algorithm 1 and aGRAAL, and in Algorithm 2, $\bar{\phi} = 10^6$, $\alpha = 1.5$, and $\lambda_0 = 1$. The stepsizes for PrGD and PrRefGD are chosen for each problem as the maximum values they can take without diverging. We also note that projection operators in all examples are evaluated using the solver `OSQP solver`[1].

(1) **Nash–Cournot equilibrium problem [25].** A variational inequality that corresponds to the Nash–Cournot equilibrium is find $x^* = (x_1^*, \dots, x_n^*) \in \mathbb{R}^n_+$

$$\text{s.t.} \quad \langle F(x^*), x - x^* \rangle \geq 0, \quad \forall x \in \mathbb{R}^n_+,$$

where $F(x^*) = (F_1(x^*), \dots, F_n(x^*))$ and

$$F_i(x^*) = f_i'(x_i^*) - p\Big(\sum_{j=1}^n x_j^*\Big) - x_i^* p'\Big(\sum_{j=1}^n x_j^*\Big).$$

We assume that the function p and f_i are written as $p(Q) = 5000^{1/\gamma} Q^{-1/\gamma}$ and $f_i(x_i) = c_i x_i + \frac{\beta_i}{\beta_i+1} L_i^{\frac{1}{\beta_i}} x_i^{\frac{\beta_i+1}{\beta_i}}$. We set $n = 1000$ and generate our data randomly. Furthermore, we consider two scenarios for each entry of β, c, and L, which are drawn independently from the uniform distributions as follows:

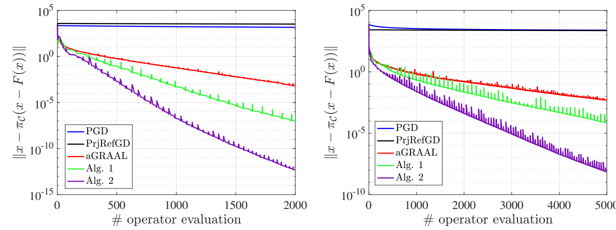
(i) $\gamma = 1.1$, $\beta_i \sim \mathcal{U}(0.5, 2)$, $c_i \sim \mathcal{U}(1, 100)$, $L_i \sim \mathcal{U}(0.5, 5)$;

(ii) $\gamma = 1.5$, $\beta_i \sim \mathcal{U}(0.3, 4)$ and c_i, L_i as above.

These parameters control the level of smoothness of f_i and p; therefore, they can affect the convergence speed. Figure 2 reports the results where all algorithms are initialized at the same point chosen randomly: Our proposed algorithms exhibits faster convergence speed and outperforms other algorithms.

(2) **Feasibility problem (finding a point in the intersection of balls) [26].** In this problem we have to find $x \in \cap_{i=1}^m C_i$, where $C_i = \mathbb{B}(c_i, r_i)$. The projection onto C_i is simple: $P_{C_i} x = \frac{x - c_i}{\|x - c_i\|} r_i$ if $\|x - c_i\| > r_i$ and x otherwise. Therefore, due to the non-expensiveness of the projection, one can use successive projections (the Krasnoselskii-Mann method) to find such a point (KM method $x^{k+1} = T x^k = \frac{1}{m} \sum_{i=1}^m P_{C_i} x^k$) [27], [28]. On the other hand, by choosing $F = \mathrm{Id} - T$, we can rewrite this problem as a VI (1). We set $n = 1000$, $m = 2000$, the center of each ball c_i is chosen randomly from $\mathcal{N}(0, 100)$,

[1] https://github.com/osqp/osqp



(a) Case (i). (b) Case (ii).

Fig. 2: Nash-Cournot equilibrium (1).

and the corresponding radius is $r_i = -\|c_i\| + 1$. The results are provided in Figure 3. The starting point of all methods are chosen as the average of all centers c_i.

(3) **Sparse logistic regression [29].** The sparse logistic regression can be written as follows

$$\min_x f(x) := \sum_{i=1}^m \log(1+\exp(-b_i \langle a_i, x \rangle)) + \gamma \|x\|_1, \tag{20}$$

where $x \in \mathbb{R}^n$, $a_i \in \mathbb{R}^n$, and $b_i \in \{-1, 1\}$, $\gamma > 0$. This problem can be found in several machine learning applications, where one attempts to find a linear classifier for points a_i. The objective function in (20) is $f(x) = s(x) + g(x)$ with $g(x) = \gamma \|x\|_1$ and $s(x) = h(Dx)$, where matrix $D \in \mathbb{R}^{m \times n}$ as $D_{ij} = -b_i a_{ij}$ and set $h(y) = \sum_{i=1}^m \log(1 + \exp(y_i))$. It is easy to see that $s(x)$ is smooth with Lipschits constant gradient with $L_{\nabla s} = \frac{1}{4} \|D^\top D\|$. We used this constant as a stepsize in PrGD and PrRefGD. In our experiments the test data a_i and b_i are generated randomly using the standard Gaussian distribution, $\gamma = 0.005 \|A^\top b\|_\infty$, where $A = [a_1 | a_2 | \cdots | a_m] \in \mathbb{R}^{n \times m}$, $n = 500$, and $m = 200$. The results are presented in Figure 4, where our methods demonstrates superior efficacy compared to other algorithms. We also plot the result of solving (20) using accelerated PrGD (FISTA) [30].

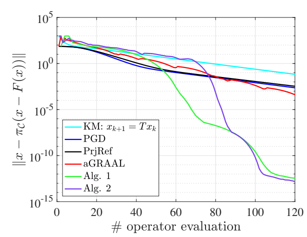


Fig. 3: Feasibility problem (2).

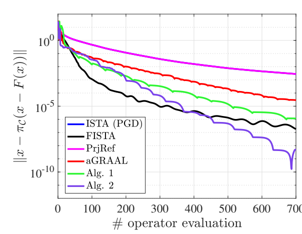


Fig. 4: Logistic regression (3).

(4) **Skew symmetric operator [31, Ex. 20.35].** An example of a monotone operator that cannot satisfy even locally strong monotonicity is the skew-symmetric operator. This operator is quite simple where we have m blocks of $n \times n$ skew symmetric matrices (SK), which is defined as

$$S = \mathrm{diag}(\mathrm{SK}_1, \mathrm{SK}_2, \cdots, \mathrm{SK}_m), \; F(x) = Sx. \tag{21}$$

In our simulation results we set $n = 10$, $m = 20$ and $\mathrm{SK}_i = \mathrm{tril}(A_i) - \mathrm{triu}(A_i)$, where A_i are symmetric positive definite matrices generated randomly ($B_i =$

randn(n,n), $A_i = B_i^\top B_i$). Figure 5 compares the convergence rates of different solving methods for the skew symmetric operator. As we have seen, our methods exhibit similar behavior to aGRAAL.

(5) **Two-player Zero Sum Game [32].** Generative adversarial networks (GANs) are a powerful class of neural networks that are used for unsupervised learning. The training of GANs can be considered a two-player zero sum game [33]. For solving a two-player zero sum game, we need to solve the following bilinear saddle point problem,

$$\min_{x\in\Delta^m} \max_{y\in\Delta^n} \Phi(x,y) := x^\top A y, \tag{22}$$

where $A \in \mathbb{R}^{m\times n}$ is a pay-off matrix and $\Delta^d = \{v \in \mathbb{R}_+^d \mid \sum_{i=1}^d v_i = 1\}$ denotes the d-dimensional simplex. The solution of (22) is given by a saddle point $(x^*, y^*) \in \Delta^m \times \Delta^n$ satisfying $\Phi(x^*, y) \le \Phi(x^*, y^*) \le \Phi(x, y^*)$ for all $(x,y) \in \Delta^m \times \Delta^n$, which can be written by problem (1) with $\mathcal{A} = \Delta^m \times \Delta^n$ and

$$F(x,y) = \begin{pmatrix} Ay \\ -A^\top x \end{pmatrix} = \begin{pmatrix} 0 & A \\ -A^\top & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}. \tag{23}$$

For the experiments, we set $d = m = n = 50$, A is generated with a uniform distribution on $[0,1)$. A comparison of methods is reported in Figure 6. As we can see from (23), this example is similar to the skew-symmetric operator (21); thus, we expect similar results.

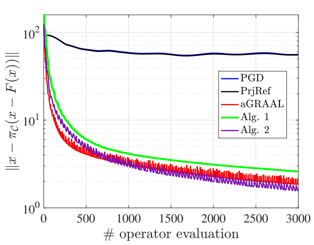


Fig. 5: Skew symmetric operator (4).

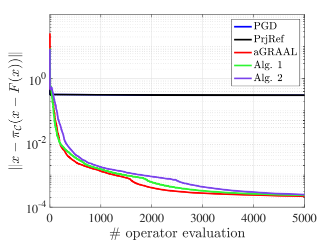


Fig. 6: Zero Sum Game (5).

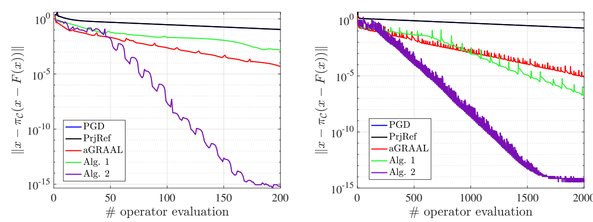
(6) **Markov decision processes (MDPs) [34].** An MDP is a pair of $(\mathcal{S}, \mathcal{A}, \mathbb{P}, c, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are the state space and action space, respectively. The transition kernel $\mathbb{P}$ describes how the system moves between states: given a state s and an action a, it shows the probability of transitioning to another state s^+. The cost function $c : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, bounded from below, assigns a cost to each action-state pair. The discount factor $\gamma \in (0,1)$ can be seen as a trade-off parameter between short- and long-term costs. We take $\mathcal{S} = \{1, 2, \dots, n\}$ and $\mathcal{A} = \{1, 2, \dots, m\}$. MDPs provide a robust modeling framework for stochastic environments, offering control mechanisms to minimize cost measures. By accessing to the transition kernel and the cost function, the problem is usually characterized by the fixed-point problem $v^\star = T(v^\star)$, i.e.,

$$v^\star(s) = [T(v^\star)](s), \quad \forall s \in \mathcal{S}, \tag{24}$$

where $T : \mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$ is the *Bellman operator* given by

$$[T(v)](s) = \min_{a\in\mathcal{A}} \left\{ c(s,a) + \gamma \mathbb{E}_{s^+ \sim \mathbb{P}(\cdot|s,a)} \left[v(s^+) \right] \right\}.$$

The optimal value function $v^\star$ is the unique fixed-point of the Bellman operator T. Therefore, we can reformulate this problem with (1) by $F = \mathrm{Id} - T$ and $g(x) = 0$. The comparison of the proposed algorithms in solving 50 instances of the optimal control problems of randomly generated Garnet MDPs with $n = 50$ states and $m = 5$ actions with two different values of discount factor γ is reported in Figure 7.



(a) $\gamma = 0.9$. (b) $\gamma = 0.99$.

Fig. 7: Performance in MDP for different values of γ (6).

(7) **Strongly monotone operator [35].** One popular VI problem with a strongly monotone operator is (1) with linear operator $F(x) = Mx + q$, where M generated randomly as $M = AA^\top + B + D$, where each entry of the $n \times n$ matrix A and the skew-symmetric matrix B is uniformly sampled from the interval $(-5, 5)$, and every entry of diagonal matrix D is uniformly sampled from the interval $(0, 0.3)$ (ensuring M is positive definite), with each entry of q uniformly sampled from $(-500, 0)$. The feasible set is $\mathcal{A} = \{x \in \mathbb{R}_+^n | \, x^1 + x^2 + \cdots + x^n = n\}$. For simulation experiments, we consider $n = 100$ and $L = |M|$ as the Lipschitz continuity of F, which is used in the stepsize of PrGD and PrRefGD methods. Figure 8 illustrates the results with initial point $x^0 = (1, 1, \dots, 1)$.

(8) **Non-monotone operator.** As a last example, we test our proposed algorithms on a non-monotone operator mentioned in [16], where we aim to find a non-zero solution of $F(x) := M(x)x = 0$. Here, $M \colon \mathbb{R}^n \to \mathbb{R}^{n\times n}$ is a matrix-valued function, which can be considered as a VI (1) with $g = 0$. For the experiment, we define M as $M(x) := t_1 t_1^\top + t_2 t_2^\top$, with $t_1 = A \sin x$, $t_2 = B \exp(x)$, where $x \in \mathbb{R}^n$, A and $B \in \mathbb{R}^{n\times n}$. For the experiment, we choose $n = 500$, and the matrices A and B are independently and randomly generated from the normal distribution $\mathcal{N}(0,1)$. The results of solving VI with the non-monotone operator M are reported in Figure 9, where the proposed algorithms outperform other methods.

V. Appendix

In this section, we provide the proof of Lemma III.1 and additional supporting material.

Definition V.1 (Cluster point)**.** *A point x is called a* ***cluster point*** *of the sequence $\{x^k\}$ if there exists a subsequence $\{x^{k_j}\}$ such that $\lim_{j\to\infty} x^{k_j} = x$.*

Lemma V.2 (Bolzano–Weierstrass theorem)**.** *If $x^k \in \mathcal{V}$ is a bounded sequence, and $\lim_{k\to\infty}(x^k - x)$ exists, where x is a cluster point of the sequence x^k, then x^k is convergent.*

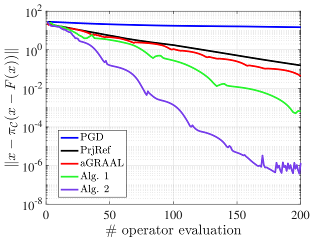


Fig. 8: Strongly monotone operator (7).

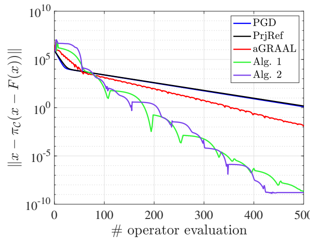


Fig. 9: Non-monotone operator (8).

Proof: [Proof of Lemma III.1] We use same proof technique as in [16]. If $x = x^* \in \mathcal{S}$ in (19), where $\mathcal{S}$ is the solution set of (1), the sequences x^k and $\bar{x}^k$ are bounded, with $\theta_k|x^k - \bar{x}^k| \to 0$. By [16, Lemma 2], λ_k and θ_k remain bounded away from zero, implying $x^k - \bar{x}^{k-1} \to 0$ and $x^{k+1} - x^k \to 0$. For any cluster point of (x^k) and $(\bar{x}^k)$, let (k_i) be a subsequence such that $x^{k_i} \to \hat{x}$ and $\lambda_{k_i} \to \lambda > 0$. Then, $x^{k_i+1} \to \hat{x}$ and $\bar{x}^{k_i} \to \hat{x}$. Taking the limit of (5) for this subsequence gives: $\lambda\langle F(\hat{x}), x - \hat{x}\rangle \geq \lambda(g(\hat{x}) - g(x)), \quad \forall x \in \mathcal{V}$. Thus, $\hat{x} \in \mathcal{S}$. Finally, by Lemma V.2, the sequence (x^k) converges to some point in $\mathcal{S}$. Now, by defining a merit function $e_r(y) := \max_{x\in\mathcal{U}} \Psi(x, y)$ for all $y \in \mathcal{V}$, where $\mathcal{U} = \operatorname{dom} g \cap \mathbb{B}(\hat{x}, r)$ and $\hat{x} \in \operatorname{dom} g$, as shown in [16, Lemma 3], we know that $e_r(y)$ is a positive convex function for all $y \in \mathcal{U}$. Furthermore, if $e_r(\tilde{x}) = 0$ for some $\tilde{x}$ where $\|\tilde{x} - \hat{x}\| \leq r$, then $\tilde{x}$ is a solution of (1). Since we assume that F is a continuous operator and g is lsc, there exists a constant $M > 0$ that bounds the right-hand-side of (19) for all $x \in \mathcal{U}$. Consequently, $\sum_{i=1}^{k} \lambda_i \Psi(x, x^i)$ can be bounded above by the constant M for all $x \in \mathcal{U}$. Finally, let X^k be the ergodic sequence defined in Lemma III.1. Using the convexity of $\Psi(x, \cdot)$, we obtain $e_r(X^k) = \max_{x\in U} \Psi(x, X^k) \leq \frac{\max_{x\in U}\left(\sum_{i=1}^{k} \Psi(x,x^i)\right)}{\sum_{i=1}^{k}\lambda_i} \leq \frac{M}{\sum_{i=1}^{k}\lambda_i}$. This implies an ergodic convergence rate of $\mathcal{O}(k^{-1})$ as λ_k is separated from zero.

References

[1] E. Benenati and S. Grammatico, "Linear-quadratic dynamic games as receding-horizon variational inequalities," *arXiv preprint arXiv:2408.15703*, 2024.

[2] J. C. De los Reyes, "Optimal control of a class of variational inequalities of the second kind," *SIAM Journal on Control and Optimization*, vol. 49, no. 4, pp. 1629–1658, 2011.

[3] F. Facchinei, A. Fischer, and C. Kanzow, "Regularity properties of a semismooth reformulation of variational inequalities," *SIAM Journal on Optimization*, vol. 8, no. 3, pp. 850–869, 1998.

[4] F. E. Browder, "A new generalization of the schauder fixed point theorem," *Mathematische Annalen*, vol. 174, no. 4, pp. 285–290, 1967.

[5] A. S. Nemirovskij and D. B. Yudin, "Problem complexity and method efficiency in optimization," 1983.

[6] G. Belgioioso and S. Grammatico, "Semi-decentralized generalized Nash equilibrium seeking in monotone aggregative games," *IEEE Transactions on Automatic Control*, vol. 68, no. 1, pp. 140–155, 2021.

[7] Y. Huang, Z. Meng, J. Sun, and W. Ren, "A unified distributed method for constrained networked optimization via saddle-point dynamics," *IEEE Transactions on Automatic Control*, pp. 1818–1825, 2023.

[8] B. Franci and S. Grammatico, "A game–theoretic approach for generative adversarial networks," in *2020 59th IEEE conference on decision and control (CDC)*. IEEE, 2020, pp. 1646–1651.

[9] G. Korpelevich, "Extragradient method for finding saddle points and other problems," *Matekon*, vol. 13, no. 4, pp. 35–49, 1977.

[10] F. Yousefian, A. Nedić, and U. V. Shanbhag, "Optimal robust smoothing extragradient algorithms for stochastic variational inequality problems," in *53rd IEEE conference on decision and control*. IEEE, 2014.

[11] S. Xie, Q. Wu, N. D. Hatziargyriou, M. Zhang, Y. Zhang, and Y. Xu, "Collaborative pricing in a power-transportation coupled network: A variational inequality approach," *IEEE Transactions on Power Systems*, vol. 38, no. 1, pp. 783–795, 2022.

[12] Y. Malitsky, "Projected reflected gradient methods for monotone variational inequalities," *SIAM Journal on Optimization*, vol. 25, no. 1, pp. 502–520, 2015.

[13] S. Cui and U. V. Shanbhag, "On the analysis of reflected gradient and splitting methods for monotone stochastic variational inequality problems," in *2016 IEEE 55th conference on decision and control (CDC)*. IEEE, 2016, pp. 4510–4515.

[14] W. Guo, M. I. Jordan, and T. Lin, "A variational inequality approach to bayesian regression games," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 795–802.

[15] B. Franci and S. Grammatico, "Distributed projected–reflected–gradient algorithms for stochastic generalized Nash equilibrium problems," in *2021 European Control Conference (ECC)*. IEEE, 2021, pp. 369–374.

[16] Y. Malitsky, "Golden ratio algorithms for variational inequalities," *Mathematical Programming*, vol. 184, no. 1-2, pp. 383–410, 2020.

[17] B. Franci and S. Grammatico, "Stochastic generalized Nash equilibrium-seeking in merely monotone games," *IEEE Transactions on Automatic Control*, vol. 67, no. 8, pp. 3905–3919, 2021.

[18] S. Krilašević and S. Grammatico, "An extremum seeking algorithm for monotone Nash equilibrium problems," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 1232–1237.

[19] G. Pantazis, R. R. Bahbadorani, and S. Grammatico, "Nash equilibrium seeking for a class of quadratic-bilinear wasserstein distributionally robust games," *arXiv preprint arXiv:2411.09636*, 2024.

[20] D. Liberzon, *Switching in systems and control*. Springer, 2003.

[21] R. Goebel, R. G. Sanfelice, and A. R. Teel, "Hybrid dynamical systems," *IEEE control systems magazine*, vol. 29, no. 2, pp. 28–93, 2009.

[22] J. I. Poveda and A. R. Teel, "A hybrid systems approach for distributed nonsmooth optimization in asynchronous multi-agent sampled-data systems," *IFAC-PapersOnLine*, vol. 49, no. 18, pp. 152–157, 2016.

[23] J. I. Poveda, R. Kutadinata, C. Manzie, D. Nešić, A. R. Teel, and C.-K. Liao, "Hybrid extremum seeking for black-box optimization in hybrid plants: An analytical framework," in *2018 IEEE Conference on Decision and Control (CDC)*. IEEE, 2018, pp. 2235–2240.

[24] A. Alacaoglu, A. Böhm, and Y. Malitsky, "Beyond the golden ratio for variational inequality algorithms," *Journal of Machine Learning Research*, vol. 24, no. 172, pp. 1–33, 2023.

[25] F. Facchinei and J.-S. Pang, *Finite-dimensional variational inequalities and complementarity problems*. Springer, 2003.

[26] H. H. Bauschke and J. M. Borwein, "On projection algorithms for solving convex feasibility problems," *SIAM review*, pp. 367–426, 1996.

[27] M. A. Krasnosel'skii, "Two remarks on the method of successive approximations," *Uspekhi matematicheskikh nauk*, pp. 123–127, 1955.

[28] W. R. Mann, "Mean value methods in iteration," *Proceedings of the American Mathematical Society*, vol. 4, no. 3, pp. 506–510, 1953.

[29] K. Mishchenko, "Regularized newton method with global convergence," *SIAM Journal on Optimization*, vol. 33, no. 3, pp. 1440–1462, 2023.

[30] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM journal on imaging sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[31] H. H. Bauschke and P. L. Combettes, *Convex analysis and monotone operator theory in Hilbert spaces*. Springer, 2017.

[32] C. E. Lemke and J. T. Howson, Jr, "Equilibrium points of bimatrix games," *Journal of the Society for industrial and Applied Mathematics*, vol. 12, no. 2, pp. 413–423, 1964.

[33] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[34] M. A. S. Kolarijani and P. M. Esfahani, "From optimization to control: Quasi policy iteration," *arXiv preprint arXiv:2311.11166*, 2023.

[35] P. T. Harker and J.-S. Pang, "For the linear complementarity problem," *Lectures in Applied Mathematics*, vol. 26, 1990.