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 two new methods for solving monotone variational inequalities. We show that by selecting the momentum parameter beyond the golden ratio in aGRAAL, 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 seeking, composite minimization, Markov decision processes, and zero-sum games, and compare them to that of existing methods.**

I. Introduction

The variational inequality (VI) 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].

Motivating example (Linear-Quadratic Dynamic Game [1]). Consider a finite-horizon open-loop linear–quadratic (LQ) dynamic game with N agents and shared system state $x_t \in \mathbb{R}^n$ evolving as

$$x_{t+1} = A_t x_t + \sum_{i=1}^{N} B_t^{(i)} u_t^i, \quad t \in \{0, \dots, T-1\},$$

where $u_t^i \in \mathbb{R}^{m_i}$ is the control input of agent i at time t. Each agent i seeks to minimize the quadratic cost

$$J_i(u^1, \dots, u^N) = \frac{1}{2} \sum_{t=0}^{T-1} \Big(x_t^\top Q_t^{(i)} x_t + u_t^{i\top} R_t^{(i)} u_t^i + \sum_{j \neq i} u_t^{j\top} S_t^{(i,j)} u_t^i \Big) + \frac{1}{2} x_T^\top Q_T^{(i)} x_T,$$

with $Q_t^{(i)}, Q_T^{(i)} \succeq 0$ weighting the state deviation, $R_t^{(i)} \succ 0$ penalizing the individual control effort, and $S_t^{(i,j)}$ capturing the *interaction or coupling* between the control actions of agents i and j. We impose *local constraints* on each agent's control input $u^i \in \mathcal{V}^i$ and *global constraints* on the state trajectory $x_t \in \mathcal{X}$. By recursively substituting the dynamics, the state trajectory can be expressed as an affine function of the stacked control vector $u = \mathrm{col}(u^1, \dots, u^N)$. The resulting Nash equilibrium conditions can then be formulated as an affine variational inequality:

$$\text{find } u^* \in \mathcal{V} := \mathcal{V}^1 \times \cdots \times \mathcal{V}^N \text{ such that}$$

$$\langle F(u^*), v - u^* \rangle \geq 0, \quad \forall v \in \mathcal{V},$$

where F is the pseudogradient operator of cost J_i defined as

$$F(u) = \begin{bmatrix} \nabla_{u^1} J_1(u) \\ \vdots \\ \nabla_{u^N} J_N(u) \end{bmatrix}.$$

This framework illustrates how shared dynamics and interactions among agents naturally lead to a VI problem, providing a tractable benchmark for developing fast and scalable algorithms for constrained multi-agent dynamic games in various applications, such as distributed automatic generation control, vehicle platooning, and the control of autonomous vehicles navigating a crossroad [1], [3].

This motivates fast algorithms for the following 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, Lipschitz continuous, 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). Note that VI problem 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) [4]. 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:

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

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

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.

$g(x) = 0$ [5].

Several iterative algorithms have been introduced to address VI problems (1). For comparison purposes, let us review some recent and closely related existing methods used in system and control applications. For simplicity, let us consider the VI problem formulation in (2).

Projected Gradient descent (PGD) [6]:

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

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 $\lambda \in \big(0, 2\mu/L^2\big)$. This algorithm is used for equilibrium seeking in aggregative games [7] (IEEE TAC, 2021), optimal consensus and resource allocation [8] (IEEE TAC, 2023), open-loop Nash equilibrium in linear quadratic dynamic games [1], and game theoretical approach for generative adversarial network [9] (IEEE CDC, 2020).

Extragradient descent [10]:

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

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 [11] (IEEE CDC, 2014) implement this method to design robust stochastic extragradient algorithms for solving monotone VIs. Similarly, [8] (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 [12] (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) [13]:

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

where λ is the stepsize, and the convergence of this method is guaranteed for Lipschitz and monotone operator (with a Lipschitz constant L) with $\lambda \in \big(0, (\sqrt{2}-1)/L\big)$. Unlike the extragradient method, PrjRef needs only one projection per iteration. The authors in [14] (IEEE CDC, 2016) implement this method to design a stochastic monotone VI algorithm under some weak sharpness assumptions. Likewise, this method is adopted in [15] (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 [16] (ECC, 2021) also use the projected-reflected method to design an algorithm for stochastic generalized Nash equilibrium problems.

Golden RAtio ALgorithm (GRAAL) [17]:

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

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) [17]:

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

This algorithm is applied in [18] (IEEE TAC, 2021) and [19] (IEEE CDC, 2021) to design a method for (stochastic) generalized Nash equilibrium in monotone games. The authors in [20] 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. We also refer interested readers to [21] for additional algorithms for applications of monotone VI, as well as a ready-to-use Python toolbox.

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 [22]. Then, based on this context and the application of hybrid system stability in optimization and extremum seeking [23]–[25], 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 [17] 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 [26]. Briefly speaking, if F is a Lipschitz and monotone operator, our proposed methods switch between PGD (method without momentum) and aGRAAL (method with the negative momentum) 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 and projection evaluations compared to aGRAAL. However, in the worst case, these computations may need to be performed twice.

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 $\mathcal{A}$, $\text{dist}(x,\mathcal{A})$ the distance from x to set $\mathcal{A}$ ($\text{dist}(x,\mathcal{A}) = \|\pi_{\mathcal{A}}(x) - x\|$), 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 $\|F(x) - F(y)\| \leq L\|x - y\|$. Furthermore, F is locally Lipschitz, if it is Lipschitz over any compact set of its domain. The operator F is monotone if $\langle F(x) - F(y), x - y\rangle \geq 0$ for all $x, y \in \mathcal{V}$ and it is called strongly monotone with constant $\mu > 0$ if $\langle F(x) - F(y), x - y\rangle \geq \mu\|x - y\|^2$ for all $x, y \in \mathcal{V}$. The prox operator of a function $g\colon \mathcal{V} \to \mathbb{R}$ is defined as $\text{prox}_g(x) = \arg\min_u\{g(u) + \|u - x\|^2/2\}$. 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 [27]:

$$y = \text{prox}_g x \iff \langle y - x, z - y\rangle \geq g(y) - g(z), \quad \forall z \in \mathcal{V} \tag{3a}$$

$$\|ax + (1-a)y\|^2 = a\|x\|^2 + (1-a)\|y\|^2 - a(1-a)\|x - y\|^2. \quad \forall x, y \in \mathcal{V}, \forall a \in \mathbb{R} \tag{3b}$$

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 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 $\Psi(x^*, x) \geq 0$, $\forall x \in \mathcal{V}$.

Algorithm 1 Adaptive algorithm for VI (Method 1)

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

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

Find the stepsize:

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

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:

$$x^{k+1} = \text{prox}_{\lambda_k g}(\bar{x}^k - \lambda_k F(x^k))$$

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

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

Algorithm 2 Adaptive algorithm for VI (Method 2)

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

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

Find the stepsize:

$$\lambda_k = \min\left\{\rho\lambda_{k-1}, \frac{\alpha\theta_{k-1}}{4\lambda_{k-1}}\frac{\|x^k - x^{k-1}\|^2}{\|F(x^k) - F(x^{k-1})\|^2}, \bar{\lambda}\right\}$$

$$\bar{x}^k = \frac{(\phi_k - 1)x^k + \bar{x}^{k-1}}{\phi_k}$$

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 $\phi_{k+1} = \bar{\phi}$:

$$\text{sum}_{k+1}^1 = \text{sum}_k^1 + (5)$$
$$\text{sum}_{k+1}^2 = \text{sum}_k^2 + (6)$$

if $(\text{sum}_{k+1}^1 \leq 0 \ \wedge$ flg $= 1) \ \vee \ (\text{sum}_{k+1}^2 \leq 0 \ \wedge$ flg $= 0)$ **then**

$\phi_{k+1} = \bar{\phi}$, flg $= 1$

else

if flg $= 1$ **then**

$x^{k+1} = x^k$, $x^k = x^{k-1}$, $\bar{x}^k = \bar{x}^{k-1}$

$\phi_{k+1} = \alpha$, $\theta_k = \theta_{k-1}$, $\lambda_k = \lambda_{k-1}$

$\text{sum}_{k+1}^1 = 0$, $\text{sum}_{k+1}^2 = 0$, flg $= 0$

else

$\phi_{k+1} = \alpha$

$\text{sum}_{k+1}^2 = \text{sum}_k^2 + ((6)$ with $\phi_{k+1} = \alpha)$

$\text{sum}_{k+1}^1 = 0$

end if

end if

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

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

Proof: Our proof is based on inequalities from monotone operator theory and convex analysis, as well as the classical Cauchy–Schwarz inequality. For brevity, the complete proof is provided in the extended version []. ■

By controlling the right-hand-side of (4), we can prove the boundedness and convergence of the sequence $(x^k)_{k\in\mathbb{N}}$. 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. The idea of Algorithm 1 is to 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 [27, 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) [17, Theorem 2]. We initiate the algorithm without the momentum term, and by computing the residual, $J_k = \|x^k - \text{prox}_g(x^k - F(x^k))\|$, 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 $\frac{1}{k}$, 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-

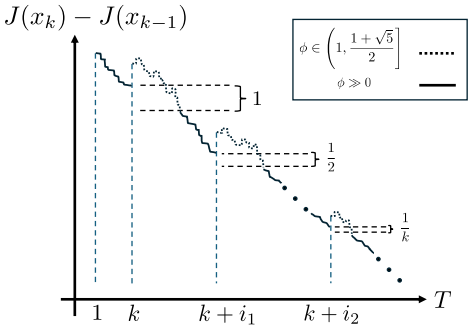


Fig. 1: Residual variation induced by Algorithm 1.

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 $\sum_k 1/\bar{k} \to \infty$.

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 (5) 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, x^{k+1} that 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 $\left(1, \frac{1+\sqrt{5}}{2}\right]$.

$$\begin{aligned}&\frac{\theta_{k-1}}{2}\|x^k - x^{k-1}\|^2 - \frac{\lambda_k}{\lambda_{k-1}}\phi_k\|x^k - \bar{x}^k\|^2\\ &+\big(\frac{\lambda_k}{\lambda_{k-1}}\phi_k - 1 - \frac{1}{\phi_{k+1}}\big)\|x^{k+1} - \bar{x}^k\|^2\\ &-\big(\frac{\lambda_k}{\lambda_{k-1}}\phi_k - \theta_k\big)\|x^{k+1} - x^k\|^2 - \frac{\theta_k}{2}\|x^{k+1} - x^k\|^2. \quad (5)\end{aligned}$$

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

$$\begin{aligned}-\frac{\lambda_k\phi_k}{\lambda_{k-1}}\|x^k - \bar{x}^k\|^2 + \big(\frac{\lambda_k\phi_k}{\lambda_{k-1}} - 1 - \frac{1}{\phi_{k+1}}\big)\|x^{k+1} - \bar{x}^k\|^2\\ - \big(\frac{\lambda_k\phi_k}{\lambda_{k-1}} - \theta_k\big)\|x^{k+1} - x^k\|^2. \quad (6)\end{aligned}$$

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 (5) 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 [17, Algorithm 1].

(iii) **Switching between small and large ϕ :** If ϕ_k is small and by modifying ϕ_{k+1} to a larger value, (6) becomes negative, we adjust ϕ_{k+1} to a larger value in the subsequent step. Then, the inequality $\frac{\phi_{k+1}}{\phi_{k+1}-1} \leq \frac{\phi_k}{\phi_k-1}$ holds, and (4) in two steps is as follows:

$$\begin{aligned}&\big(\frac{\phi_k}{\phi_k - 1} - \frac{\phi_{k+1}}{\phi_{k+1} - 1}\big)\|\bar{x}^k - x\|^2 + \frac{\phi_{k+1}}{\phi_{k+1} - 1}\|\bar{x}^k - x\|^2\\ &+ \frac{\theta_{k-1}}{2}\|x^k - x^{k-1}\|^2 + 2\lambda_{k-1}\Psi(x, x^{k-1})\\ &\leq \frac{\phi_k}{\phi_k - 1}\|\bar{x}^{k-1} - x\|^2 + \frac{\theta_{k-2}}{2}\|x^{k-1} - x^{k-2}\|^2\\ &+ \big(\frac{\lambda_{k-1}}{\lambda_{k-2}}\phi_{k-1} - 1 - \frac{1}{\phi_k}\big)\|x^k - \bar{x}^{k-1}\|^2\\ &- \frac{\lambda_{k-1}}{\lambda_{k-2}}\phi_{k-1}\|x^{k-1} - \bar{x}^{k-1}\|^2\\ &- \big(\frac{\lambda_{k-1}}{\lambda_{k-2}}\phi_{k-1} - \theta_{k-1}\big)\|x^k - x^{k-1}\|^2. \quad (7)\end{aligned}$$

$$\begin{aligned}&\frac{\phi_{k+1}}{\phi_{k+1} - 1}\|\bar{x}^{k+1} - x\|^2 + \frac{\theta_k}{2}\|x^{k+1} - x^k\|^2 + 2\lambda_k\Psi(x, x^k)\\ &\leq \frac{\phi_{k+1}}{\phi_{k+1} - 1}\|\bar{x}^k - x\|^2 + \frac{\theta_{k-1}}{2}\|x^k - x^{k-1}\|^2\\ &- \frac{\lambda_k}{\lambda_{k-1}}\phi_k\|x^k - \bar{x}^k\|^2 - \big(\frac{\lambda_k}{\lambda_{k-1}}\phi_k - \theta_k\big)\|x^{k+1} - x^k\|^2\\ &+ \big(\frac{\lambda_k}{\lambda_{k-1}}\phi_k - 1 - \frac{1}{\phi_{k+1}}\big)\|x^{k+1} - \bar{x}^k\|^2, \quad (8)\end{aligned}$$

where in the first line of (7) 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 (5) 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 "$\text{sum}_{k+1}^1 \leq 0$" in Algorithm 2 (negative summation of (5)) 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:

$$
\begin{aligned}
&(\frac{\phi_{k+i}}{\phi_{k+i}-1} - \frac{\phi_{k+i+1}}{\phi_{k+i+1}-1})\|\bar{x}^{k+i} - x\|^2 \\
&+ \frac{\phi_{k+i+1}}{\phi_{k+i+1}-1}\|\bar{x}^{k+i} - x\|^2 + \frac{\theta_{k+i-1}}{2}\|x^k - x^{k+i-1}\|^2 \\
&+ 2\lambda_{k+i-1}\Psi(x, x^{k+i-1}) \leq \frac{\phi_{k+i}}{\phi_{k+i}-1}\|\bar{x}^{k+i-1} - x\|^2 \\
&+ \frac{\theta_{k+i-2}}{2}\|x^{k+i-1} - x^{k+i-2}\|^2 \\
&- \frac{\lambda_{k+i-1}}{\lambda_{k+i-2}}\phi_{k+i-1}\|x^{k+i-1} - \bar{x}^{k+i-1}\|^2 \\
&+ \big(\frac{\lambda_{k+i-1}}{\lambda_{k+i-2}}\phi_{k+i-1} - 1 - \frac{1}{\phi_{k+i}}\big)\|x^{k+i} - \bar{x}^{k+i-1}\|^2 \\
&- \big(\frac{\lambda_{k+i-1}}{\lambda_{k+i-2}}\phi_{k+i-1} - \theta_{k+i-1}\big)\|x^{k+i} - x^{k+i-1}\|^2. \qquad (9)
\end{aligned}
$$

$$
\begin{aligned}
&\frac{\phi_{k+i+1}}{\phi_{k+i+1}-1}\|\bar{x}^{k+i+1} - x\|^2 + \frac{\theta_{k+i}}{2}\|x^{k+i+1} - x^{k+i}\|^2 \\
&+ 2\lambda_{k+i}\Psi(x, x^{k+i}) \leq \frac{\phi_{k+i+1}}{\phi_{k+i+1}-1}\|\bar{x}^{k+i} - x\|^2 \\
&+ \frac{\theta_{k+i-1}}{2}\|x^{k+i} - x^{k+i-1}\|^2 \\
&- \frac{\lambda_{k+i}}{\lambda_{k+i-1}}\phi_{k+i}\|x^{k+i} - \bar{x}^{k+i}\|^2 \\
&+ \big(\frac{\lambda_{k+i}}{\lambda_{k+i-1}}\phi_{k+i} - 1 - \frac{1}{\phi_{k+i+1}}\big)\|x^{k+i+1} - \bar{x}^{k+i}\|^2 \\
&- \big(\frac{\lambda_{k+i}}{\lambda_{k+i-1}}\phi_{k+i} - \theta_{k+i}\big)\|x^{k+i+1} - x^{k+i}\|^2, \qquad (10)
\end{aligned}
$$

where in the first line of (9) 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 (7) and (8), or switching from a large ϕ to a small one (9) and (10)), we drive (11). More precisely, the conditions in line 7 of Algorithm 2 ensure that, by telescoping (4), we obtain similar coefficient terms on the right and left-hand side of successive lines of (4) (e.g., the leftmost terms in (7) and (9) 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 inequality (11) after T iterations, as follows:

$$
\begin{aligned}
&\frac{\phi_T}{\phi_T - 1}\|\bar{x}^T - x\|^2 + \frac{\theta_{T-1}}{2}\|x^T - x^{T-1}\|^2 + 2\sum_{i=1}^{T}\lambda_i\Psi(x, x^i) \\
&\quad \leq \frac{\phi_2}{\phi_2 - 1}\|\bar{x}^1 - x\|^2 + \frac{\theta_0}{2}\|x^1 - x^0\|^2 + D, \qquad (11)
\end{aligned}
$$

where D is a non-positive constant which is summation of the three negative rightmost terms in (4). Note that T in (11) 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 [17], we can prove the ergodic convergence rate based on (11). The following theorem indicate the convergence properties of Algorithm 2.

Theorem 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. More precisely we have*

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

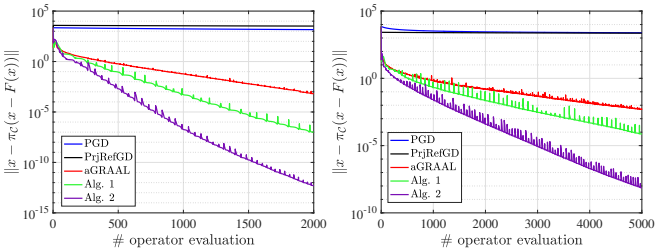
where $M > 0$ is some constant dominates the right-hand side of (11) for all $x \in \mathcal{U}$, in particular $\sum_{i=1} \lambda_i \Psi(x, x^i) \leq M$.

Proof: We use the same proof technique as in [17]; the interested reader can find the complete proof in the extended version []. ■

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_{k+1}^2 with a small ϕ_{k+1} if sum_{k+1}^2 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 six classes of VI problems studied in the literature: (1) Nash–Cournot equilibrium, (2) sparse logistic regression, (3) Two-player Zero Sum Game, (4) Markov decision processes, (5) strongly affine monotone operator, and (6) VI problem with non-monotone operator. Due to space limitations, we present only the simulation results for the Nash–Cournot equilibrium here and refer the interested reader to the extended version of this paper [] for the remaining simulations. 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



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

Fig. 2: Nash-Cournot equilibrium.

relatively recent method for monotone variational inequality and the closest in spirit to our proposed methods. 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 $\lambda_0 = \bar{\lambda} = 1$ in Algorithm 1 and aGRAAL, and in Algorithm 2, $\bar{\phi} = 1$, $\alpha = 1.5$, and $\lambda_0 = \bar{\lambda} = 1$. The stepsizes for PrGD and PrRefGD are selected in each problem based on the Lipschitz constant of the underlying operator or the largest values that prevent divergence. Note that projection operators in all examples are evaluated using the solver `OSQP solver`[1].

Nash–Cournot equilibrium problem [27]. 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.

References

[1] E. Benenati and S. Grammatico, "Linear-quadratic dynamic games as receding-horizon variational inequalities," *IEEE Transactions on Automatic Control*, 2025, to appear.

[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] R. R. Baghbadorani, E. Benenati, and S. Grammatico, "A douglas-rachford splitting method for solving monotone variational inequalities in linear-quadratic dynamic games," *preprint available at arXiv:2504.05757*, 2025.

[4] 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.

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

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

[7] 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.

[8] 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.

[9] 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.

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

[11] 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.

[12] 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.

[13] Y. Malitsky, "Projected reflected gradient methods for monotone variational inequalities," *SIAM Journal on Optimization*, 2015.

[14] 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.

[15] 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.

[16] 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.

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

[18] 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.

[19] 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.

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

[21] N. Mignoni, R. R. Baghbadorani, R. Carli, P. M. Esfahani, M. Dotoli, and S. Grammatico, "monviso: A python package for solving monotone variational inequalities," in *European Control Conference.*, 2025.

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

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

[24] 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.

[25] 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.

[26] 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.

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

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