OptiML Group Meeting, 2022-2023 Winter

A computer-aided convergence analysis of algorithms for smooth monotone game

'Finite-Time Last-Iterate Convergence for Learning in Multi-Player Games

(Cai et al., NeurIPS 2022)'

Presenter: Hanseul Cho

March 17th, 2023

OptiML Optimization and Machine Learning Laboratory

Overview

Game class	Setting	Step size	Convergence rate
Strongly monotone	general	constant	$O(c^{-T})$ (see e.g., [Tse95] [LS19, MOP19, ZMM$^+$21])
Cocoercive	unconstrained	constant	$O(\frac{1}{\sqrt{T}})$ [LZMJ20]
Monotone	general	constant	Asymptotic [Pop80, HIMM19]
	general	decreasing	Asymptotic * (see e.g., [ZMM$^+$17] [ZMA$^+$18, MZ19, HAM21])
	unconstrained	constant	$O(\frac{1}{\sqrt{T}})^\dagger$ [GPD20]
	general	constant	$O(\frac{1}{\sqrt{T}})$ **[This paper]**

Table 1: Last-iterate convergence for no-regret learning in smooth monotone games with perfect gradient feedback. (*) The results hold for variationally stable games. (†) The result holds under an additional second-order smoothness assumption.

- Contribution: non-asymptotic last-iterate convergence results on both **ExtraGradient (EG) algorithm** and **Optimistic Gradient (OG) algorithm** of rate $\mathcal{O}(\frac{1}{\sqrt{T}})$ to a Nash equilibrium of a *constrained* multi-player smooth monotone game.
- The proofs heavily utilize **computer-aided** analyses based on **Sum-of-Square (SOS) Programming**.

Table of Contents

Continuous game with convex constraints

- Multi-player continuous game $\mathcal{G} = (\mathcal{N}, \{\mathcal{Z}^{(i)}\}_{i\in[N]}, \{f^{(i)}\}_{i\in[N]})$:
- N players $\mathcal{N} = \{1, \cdots, N\}$
- For each player $i \in \mathcal{N}$,
  - Constraint (*i.e.*, set of actions) $\mathcal{Z}^{(i)}$: closed convex in $\mathbb{R}^{n_i}$
  - Cost function $f^{(i)} : \mathcal{Z} \to \mathbb{R}$ where $\mathcal{Z} := \mathcal{Z}^{(1)} \times \cdots \times \mathcal{Z}^{(N)} \subset \mathbb{R}^n$

- Action profile $z = (z^{(1)}, \cdots, z^{(N)}) \in \mathcal{Z}$
  - Denote by $z^{(-i)}$ the vector of actions of all the other players than i
- Player i then receives gradient feedback $\nabla_{z^{(i)}} f^{(i)}(z)$

- **Nash equilibrium** (NE): an action profile $z^* \in \mathcal{Z}$ such that:
  - $f^{(i)}(z^*) \le f^{(i)}(z'^{(i)}, z^{*(-i)})$ for any feasible action $z'^{(i)} \in \mathcal{Z}^{(i)}$.
  - Meaning: where every single change of action is sub-optimal.

Smooth monotone game

- Consider an operator $F : \mathcal{Z} \subset \mathbb{R}^n \to \mathbb{R}^n$ defined by

$$F(\cdot) = \left(\nabla_{z^{(1)}} f^{(1)}(\cdot), \cdots, \nabla_{z^{(N)}} f^{(N)}(\cdot)\right)$$

- Game $\mathcal{G}$ is L-smooth and monotone (resp.) if the operator F is:
  1. L-**Lipschitz**: $\|F(z) - F(z')\| \leq L \|z - z'\|$,
  2. **monotone**: $\langle F(z) - F(z'), z - z' \rangle \geq 0$.
- Facts:
  - Monotone game $\implies$ concave game (every $f^{(i)}$ is convex).
  - For monotone game, z^* is NE $\iff$ $\langle F(z^*), z^* - z \rangle \leq 0$ $(\forall z \in \mathcal{Z})$.[1]
- Examples:
  - 1^{st}-order optimality of constrained convex minimization of $f(x)$: $\langle \nabla f(x^*), x^* - x \rangle \leq 0$ $(\forall x \in \mathcal{X})$.
  - Convex-concave minimax optimization: $f^{(1)}(x_1, x_2) = -f^{(2)}(x_1, x_2)$, where $f^{(i)}$ is convex in x_i.

[1] The problem of finding such a z^* is called *monotone VI (variational inequality)*, which has monotone game as a special case.

Failure of gradient descent

- Gradient descent (GD) : $z_{k+1} = z_k - \eta F(z_k)$: does it work?
- With only $N \geq 2$, it fails to have best-/last-iterate convergence
  - Consider $f^{(1)}(x, y) = -f^{(2)}(x, y) = x \cdot y$: $\min_x \max_y x \cdot y$.
  - GD can only have average-iterate convergence, of rate $\mathcal{O}(1/\sqrt{T})$.

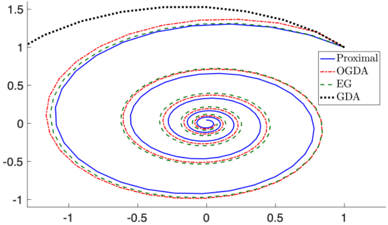


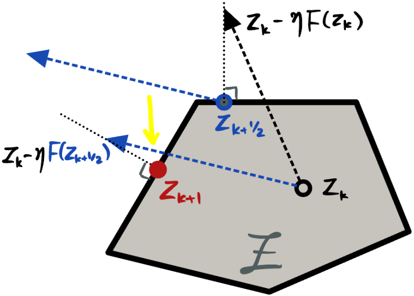
Figure 1: Convergence trajectories of proximal point (PP), extra-gradient (EG), optimistic gradient descent ascent (OGDA), and gradient descent ascent (GDA) for $\min_x \max_y xy$. The proximal point method has the fastest convergence. EG and OGDA approximate the trajectory of PP and both converge to the optimal solution. The GDA method is the only method that diverges.

* Figure from Mokhtari et al. [2020]

Extragradient (EG) & Optimistic Gradient (OG)

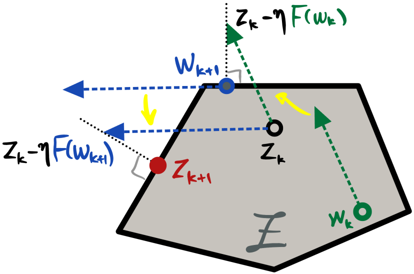
- EG [Korpelevich, 1976]:

$$z_{k+1/2} = \Pi_{\mathcal{Z}}[z_k - \eta F(z_k)]$$
$$z_{k+1} = \Pi_{\mathcal{Z}}[z_k - \eta F(z_{k+1/2})]$$



- OG [Popov, 1980]:

$$w_{k+1} = \Pi_{\mathcal{Z}}[z_k - \eta F(w_k)]$$
$$z_{k+1} = \Pi_{\mathcal{Z}}[z_k - \eta F(w_{k+1})]$$



- “Explore, then Update” structure in common
- Tractable approximations of 'proximal point (PP)' algorithm [Mokhtari et al., 2020, Yoon and Ryu, 2022]

Extragradient (EG) & Optimistic Gradient (OG)

- EG [Korpelevich, 1976] :

$$z_{k+1/2} = \Pi_{\mathcal{Z}}[z_k - \eta F(z_k)]$$
$$z_{k+1} = \Pi_{\mathcal{Z}}[z_k - \eta F(z_{k+1/2})]$$

- OG [Popov, 1980] :

$$w_{k+1} = \Pi_{\mathcal{Z}}[z_k - \eta F(w_k)]$$
$$z_{k+1} = \Pi_{\mathcal{Z}}[z_k - \eta F(w_{k+1})]$$

For smooth & (non-strongly-)monotone games,

- Average-iterate: $\mathcal{O}(1/T)$ [Nemirovski, 2004, Mokhtari et al., 2020]
  - Faster than GD.
- Best-/random-iterate: $\mathcal{O}(1/\sqrt{T})$ [Korpelevich, 1976, Gorbunov et al., 2022]
- Last-iterate:
  - EG: $\mathcal{O}(1/\sqrt{T})$ in *unconstrained* setting [Gorbunov et al., 2022]
  - OG: $\mathcal{O}(1/\sqrt{T})$ under additional second-order smoothness [Golowich et al., 2020]

Table of Contents

Proof Plan

- Three steps:
  1. Define a good potential function $\Phi(\cdot)$.
  2. Prove a **best-iterate** convergence rate for $\Phi(\cdot)$.
  3. Prove that $\Phi(\cdot)$ is **non-increasing**; the last-iterate is the best-iterate.
- Remarks:
  - Usually, 2 requires a proof that the average of potential values over iterations converges to 0 (*e.g.*, $\frac{1}{T}\sum_{t=1}^{T}\Phi(z_t) = \mathcal{O}(\frac{1}{\sqrt{T}})$).[2]
  - "best-iterate convergence $\not\Longrightarrow$ last-iterate convergence."
  - Discovery of $\Phi(\cdot)$ (1) & Monotonicity proof (3) are done with help of computer-aided analyses based on Sum-of-Square (SOS) Programming.

[2] It builds upon vast amount of existing literature; I will skip this today.

Monotonicity proof: EG, unconstrained

Theorem (Cai et al. [2022])

Let F be a monotone and L-Lipschitz operator. Then, EG algorithm with step size $\eta \in (0, \frac{1}{L})$ satisfies $\|F(z_k)\|^2 \geq \|F(z_{k+1})\|^2$.

Proof.

- EG update: $z_{k+1/2} = z_k - \eta F(z_k)$; $z_{k+1} = z_k - \eta F(z_{k+1/2})$
- monotone:
$$\begin{aligned} &\tfrac{1}{\eta}\langle F(z_{k+1}) - F(z_k), z_{k+1} - z_k\rangle \\ &= \langle F(z_{k+1}) - F(z_k), -F(z_{k+1/2})\rangle \geq 0. \end{aligned} \tag{1}$$
- L-Lipschitz & $\eta^2 L^2 < 1$:
$$\begin{aligned} L^2 \left\|z_{k+1/2} - z_{k+1}\right\|^2 - \left\|F(z_{k+1/2}) - F(z_{k+1})\right\|^2 &\geq 0, \\ \implies \left\|F(z_{k+1/2}) - F(z_k)\right\|^2 - \left\|F(z_{k+1/2}) - F(z_{k+1})\right\|^2 &\geq 0. \end{aligned} \tag{2}$$
- The proof concludes by showing
$$\|F(z_k)\|^2 - \|F(z_{k+1})\|^2 - 2 \cdot (\text{Equation } 1) - 1 \cdot (\text{Equation } 2) = 0.$$

□

Monotonicity proof: OG, unconstrained

Theorem (Cai et al. [2022])

Let F be a monotone and L-Lipschitz operator. Then, OG with step size $\eta \in (0, \frac{1}{2L})$ satisfies $\Phi(z_k, w_k) \geq \Phi(z_{k+1}, w_{k+1})$ where $\Phi(z, w) := \|F(z) - F(w)\|^2 + \|F(z)\|^2$.

Proof.

- OG update: $w_{k+1} = z_k - \eta F(w_k)$; $z_{k+1} = z_k - \eta F(w_{k+1})$
- monotone:

$$\begin{aligned} &\tfrac{1}{\eta}\langle F(z_{k+1}) - F(z_k), z_{k+1} - z_k\rangle \\ &= \langle F(z_{k+1}) - F(z_k), -F(w_{k+1})\rangle \geq 0. \end{aligned} \tag{3}$$

- L-Lipschitz & $\eta^2 L^2 < 1/4$:

$$\begin{aligned} &L^2 \|w_{k+1} - z_{k+1}\|^2 - \|F(w_{k+1}) - F(z_{k+1})\|^2 \geq 0, \\ \implies &\tfrac{1}{4}\|F(w_{k+1}) - F(w_k)\|^2 - \|F(w_{k+1}) - F(z_{k+1})\|^2 \geq 0. \end{aligned} \tag{4}$$

- The proof concludes by showing

$$\begin{aligned} &\Phi(z_k, w_k) - \Phi(z_{k+1}, w_{k+1}) - 2\cdot(\text{Equation 3}) - 2\cdot(\text{Equation 4}) \\ &= \frac{1}{2}\|F(w_k) + F(w_{k+1}) - 2F(z_k)\|^2 \geq 0. \end{aligned}$$

□

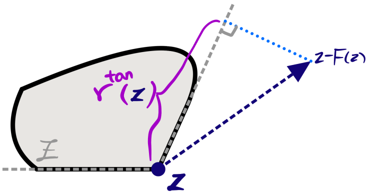
Potential functions for constrained cases

- **Tangent Residual** $r^{\tan}_{(F,\mathcal{Z})}(z)$ is defined as

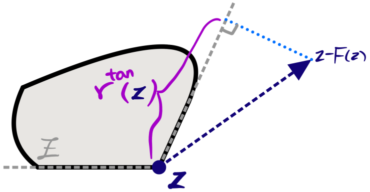
$$r^{\tan}_{(F,\mathcal{Z})}(z) := \left\| \Pi_{\{z\}+T_{\mathcal{Z}}(z)}[z - F(z)] - z \right\|$$
$$\equiv \sqrt{\|F(z)\|^2 - \max_{\substack{a \in N_{\mathcal{Z}}(z),\\ \langle a, F(z)\rangle \le 0}} \frac{\langle a, F(z)\rangle}{\|a\|^2}}$$

for an operator $F : \mathcal{Z} \to \mathbb{R}^n$ defined on a closed convex set $\mathcal{Z}$, where

$$N_{\mathcal{Z}}(z) = \{v : \langle v, z' - z\rangle \le 0, \forall z' \in \mathcal{Z}\}, \qquad \text{(normal cone)}$$
$$T_{\mathcal{Z}}(z) = \{z' : \langle z', a\rangle \le 0, \forall a \in N_{\mathcal{Z}}(z)\} \qquad \text{(tangent cone)}$$



Potential functions for constrained cases



- Potential for **EG**: $\Phi(z_k) = r^{\tan}_{(F,\mathcal{Z})}(z_k)^2$.
- Potential for **OG**: $\Phi(z_k, w_k) = \|F(z) - F(w)\|^2 + r^{\tan}_{(F,\mathcal{Z})}(z_k)^2$.
- Remarks:
  - If $\mathcal{Z} = \mathbb{R}^n$, then $r^{\tan}_{(F,\mathcal{Z})}(z) = \|F(z)\|$.
  - Any other well-studied performance measures are observed as *non-monotone* (*e.g.*, gap function ($\max_{z' \in \mathcal{Z}} \langle F(z), z - z' \rangle$), natural residual, distance from Nash eq., $\|z_{k+1} - z_k\|^2$, etc.)

Table of Contents

Computer-aided proofs for Convergence Analysis

1. Given: a problem of finding convergence rate or proving a certain inequality
2. Convert it to a constrained optimization/feasibility problem (which is possibly infinite-dimensional)
3. Apply relaxations to yield a tractable (finite-dimensional) semi-definite programming (SDP) (or other ways if possible)
4. Numerically solve the primal (or, dual) finite-dimensional program
    - Useful toolboxes in MATLAB, Mathematica, ...
5. The solution helps researchers to *come up with* a rigorous proof.

- Examples:
    - Performance Estimation Problem (PEP) [Drori and Teboulle, 2014, Taylor et al., 2017, 2018b, Taylor, 2017, Ryu et al., 2020, Zamani et al., 2022, Bousselmi et al., 2023]
    - Integrated Quadratic Constraints (IQC) [Lessard et al., 2016, Taylor et al., 2018a, Lessard and Seiler, 2020, Zhang et al., 2021]
    - **Sum-of-Square (SOS) Programming** [Fazlyab et al., 2018, Tan et al., 2021, Jarvis-Wloszek, 2003, Tan and Packard, 2008, Papp and Yildiz, 2019, Anderson and Papachristodoulou, 2015, Cai et al., 2022]

Sum-of-Square (SOS) Polynomial

- A polynomial $p(\boldsymbol{x}) \in \mathbb{R}[\boldsymbol{x}]$ is said to be **SOS polynomial** if it is a sum of squared polynomials:

$$SOS[\boldsymbol{x}] := \left\{ \sum_{i=1}^{m} \{q_i(\boldsymbol{x})\}^2 : m \in \mathbb{N}, q_1(\boldsymbol{x}), \ldots, q_m(\boldsymbol{x}) \in \mathbb{R}[\boldsymbol{x}] \right\}.$$

- Is every non-negative polynomial SOS?
  - NO: $x^4y^2 + x^2y^4 - 3x^2y^2 + 1$ (*Motzkin polynomial*) is non-negative over $\mathbb{R}^2$ (proof: AM-GM) but not SOS.
  - YES in some cases (Hilbert):
    - ⋆ Univariate & any (even) degree
    - ⋆ Any number of variables & quadratic (degree-2)
    - ⋆ Two-variable & quartic (degree-4)
- Any non-negative polynomial is a SOS of rational functions (Artin, 1927).

Sum-of-Square (SOS) Polynomial

Now concern the *constrained* non-negativity.

- Suppose we want to show $f(\boldsymbol{x}) \in \mathbb{R}[\boldsymbol{x}]$ is non-negative over a semialgebraic set $\mathcal{S} = \{\boldsymbol{x} \in \mathbb{R}^n : g_i(\boldsymbol{x}) \geq 0, h_j(\boldsymbol{x}) = 0 \, (\forall i, j)\}$, where each $g_i(\boldsymbol{x}), h_i(\boldsymbol{x}) \in \mathbb{R}[\boldsymbol{x}]$
- Sufficient conditions for $f(\boldsymbol{x}) \geq 0$ over $\mathcal{S}$:
  1. IF $f(\boldsymbol{x}) \in SOS[\boldsymbol{x}]$.
  2. IF $f(\boldsymbol{x}) - \sum_{i=1}^{m} p_i g_i(\boldsymbol{x}) - \sum_{j=1}^{m'} q_j h_j(\boldsymbol{x}) \in SOS[\boldsymbol{x}]$ for some $p_i \geq 0$ and $q_j \in \mathbb{R}$.
  3. IF $f(\boldsymbol{x}) - \sum_{i=1}^{m} p_i(\boldsymbol{x}) g_i(\boldsymbol{x}) - \sum_{j=1}^{m'} q_j(\boldsymbol{x}) h_j(\boldsymbol{x}) \in SOS[\boldsymbol{x}]$ for some $p_i(\boldsymbol{x}) \in SOS[\boldsymbol{x}]$ and $q_j(\boldsymbol{x}) \in \mathbb{R}[\boldsymbol{x}]$.
- Under certain assumption, 3 is indeed a necessary condition for non-negativity over $\mathcal{S}$ (*Putinar's Positivstellensatz*)[3]. Even without such assumption, 3 is practically useful.

[3] To see a more general result, refer to *Krivine–Stengle Positivstellensatz*

Sum-of-Square (SOS) Programming

There are two natural classes of problem in convex SOS programming. Given that $\deg f = 2d$,

- Feasibility problem: Find
  - $p_i(\boldsymbol{x}) \in SOS[\boldsymbol{x}]$ of degree $2d - \deg g_i$ and
  - $q_j(\boldsymbol{x}) \in \mathbb{R}[\boldsymbol{x}]$ of degree $2d - \deg h_j$ such that:
  - $f(\boldsymbol{x}) - \sum_{i=1}^{m} p_i(\boldsymbol{x}) g_i(\boldsymbol{x}) - \sum_{j=1}^{m'} q_j(\boldsymbol{x}) h_j(\boldsymbol{x}) \in SOS[\boldsymbol{x}]$.
- Optimization problem ($\boldsymbol{w}$ is known):
  - Find $\boldsymbol{c}$ that minimizes $\langle \boldsymbol{w}, \boldsymbol{c} \rangle$,
  - where $\boldsymbol{c}$ is a vector of (unknown) coefficients of
  - $p_i(\boldsymbol{x}) \in SOS[\boldsymbol{x}]$ of degree $2d - \deg g_i$ and
  - $q_j(\boldsymbol{x}) \in \mathbb{R}[\boldsymbol{x}]$ of degree $2d - \deg h_j$ such that:
  - $f(\boldsymbol{x}) - \sum_{i=1}^{m} p_i(\boldsymbol{x}) g_i(\boldsymbol{x}) - \sum_{j=1}^{m'} q_j(\boldsymbol{x}) h_j(\boldsymbol{x}) \in SOS[\boldsymbol{x}]$.

Applications of SOS: Monotonicity proof for EG

Theorem (Cai et al. [2022])

Let F be a monotone and L-Lipschitz operator. Then, EG algorithm with step size $\eta \in (0, \frac{1}{L})$ satisfies $\|F(z_k)\|^2 \geq \|F(z_{k+1})\|^2$.

- To prove that $\|F(z_k)\|^2 - \|F(z_{k+1})\|^2 \geq 0$, **represent all the functional and algorithmic constraints as polynomial (in-)equalities w.r.t. variables** $\{z_i[\ell], \eta F(z_i)[\ell]\}_{i \in \mathcal{K}, \ell \in [n]}$, where $\mathcal{K} = \{k, k + \frac{1}{2}, k + 1\}$.
- Due to the *symmetry across coordinates*, it suffices to use $\{z_i[1], \eta F(z_i)[1]\}_{i \in \mathcal{K}}$.

Applications of SOS: Monotonicity proof for EG

$$z_{k+\frac{1}{2}}[\ell] - z_k[\ell] + \eta F(z_k)[\ell] = 0, \quad z_{k+1}[\ell] - z_k[\ell] + \eta F(z_{k+\frac{1}{2}})[\ell] = 0, \ \forall \ell \in [n], \quad \text{(EG Update)}$$

$$\|\eta F(z_i) - \eta F(z_j)\|^2 - (\eta L)^2 \|z_i - z_j\|^2 \le 0, \quad \forall i,j \in \{k, k+\frac{1}{2}, k+1\}, \quad \text{(Lipschitzness)}$$

$$\langle \eta F(z_i) - \eta F(z_j), z_j - z_i \rangle \le 0, \quad \forall i,j \in \{k, k+\frac{1}{2}, k+1\}. \quad \text{(Monotonicity)}$$

Input Fixed Polynomials. We use x to denote (x_0, x_1, x_2) and y to denote (y_0, y_1, y_2). Interpret x_i as $z_{k+\frac{i}{2}}[\ell]$ and y_i as $\eta F(z_{k+\frac{i}{2}})[\ell]$ for $0 \le i \le 2$. Observe that $h_1(x,y)$ and $h_2(x,y)$ come from the EG update rule on coordinate ℓ. $g^L_{i,j}(x,y)$ and $g^m_{i,j}(x,y)$ come from the ℓ-th coordinate's contribution in the Lipschitzness and monotonicity constraints.

- $h_1(x,y) := x_1 - x_0 + y_0$ and $h_2(x,y) := x_2 - x_0 + y_1$.
- $g^L_{i,j}(x,y) := (y_i - y_j)^2 - C \cdot (x_i - x_j)^2$ for any $0 \le j < i \le 2$.[a]
- $g^m_{i,j}(x,y) := (y_i - y_j)(x_j - x_i)$ for any $0 \le j < i \le 2$.

Decision Variables of the SOS Program:

- $p^L_{i,j} \ge 0$, and $p^m_{i,j} \ge 0$, for all $0 \le j < i \le 2$.
- $q_1(x,y)$ and $q_2(x,y)$ are two degree 1 polynomials in $\mathbb{R}[x,y]$.

Constraints of the SOS Program:

$$\begin{aligned} \text{s.t.} \quad & y_0^2 - y_2^2 + \sum_{2 \ge i > j \ge 0} p^L_{i,j} \cdot g^L_{i,j}(x,y) + \sum_{2 \ge i > j \ge 0} p^m_{i,j} \cdot g^m_{i,j}(x,y) \\ & + q_1(x,y) \cdot h_1(x,y) + q_2(x,y) \cdot h_2(x,y) \in \mathrm{SOS}[x,y]. \end{aligned} \quad (11)$$

[a] C represents $(\eta L)^2$. Larger C corresponds to a larger step size and makes the SOS program harder to satisfy. Through binary search, we find that the largest possible value of C is 1 while maintaining the feasibility of the SOS program.

Figure 3: Our SOS program in the unconstrained setting.

Applications of SOS: Monotonicity proof for EG

- Implementation example using SOSTOOLS toolbox in MATLAB:

```
% define variables & constants; initialize SOSP
pvar x0 x1 x2 y0 y1 y2;
x = [x0;x1;x2]; y = [y0;y1;y2]; C = 1;
Program = sosprogram([x;y]);

% declare decision variables for SOSP
[Program,pL01] = sossosvar(Program, 1);
[Program,pL02] = sossosvar(Program, 1);
[Program,pL12] = sossosvar(Program, 1);
[Program,pm01] = sossosvar(Program, 1);
[Program,pm02] = sossosvar(Program, 1);
[Program,pm12] = sossosvar(Program, 1);
[Program,q1] = sospolyvar(Program, [1;x;y]);
[Program,q2] = sospolyvar(Program, [1;x;y]);
decvar = [pL01;pL02;pL12;pm01;pm02;pm12;q1;q2];

% declare constraints for SOSP
V = y0^2 - y2^2;  % obj
gL01 = C*(x0-x1)^2 - (y0-y1)^2; % Lipschitz (1)
gL02 = C*(x0-x2)^2 - (y0-y2)^2; % Lipschitz (2)
gL12 = C*(x1-x2)^2 - (y1-y2)^2; % Lipschitz (3)
gm01 = (y0-y1)*(x0-x1);         % monotone (1)
gm02 = (y0-y2)*(x0-x2);         % monotone (2)
gm12 = (y1-y2)*(x1-x2);         % monotone (3)
h1 = x1-x0+y0; h2 = x2-x0+y1;   % EG updates
cnstrnt = [gL01;gL02;gL12;gm01;gm02;gm12;h1;h2];
f = V - decvar'*cnstrnt;
Program = sosineq(Program,f);

% solve SOSP
solver_opt.solver = 'sedumi'; solver_opt.simplify = 'on';
[Program,info]=sossolve(Program,solver_opt); f_ans = sosgetsol(Program,f);
[Q,Z] = findsos(f_ans); [L,D] = eig(Q);
sqrt(D)*L*[x;y] % each row = polynomials to be squared & added
```

```
pL01    0.000000
pL02    0.000000
pL12    1.000000
pm01    0.000000
pm02    2.000000
pm12    0.000000
_q1_    0.2891*x0 - 1.7482*x1 + 1.4591*x2 + 0.2518*y0 - 0.5410*y1 + 7.6418e-06*y2
_q2_    0.9003*x0 + 0.7646*x1 - 1.6649*x2 + 0.7646*y0 + 0.3351*y1 - 2.0000*y2
```

Applications of SOS: Potential design for OG

- Tangent residual ($r^{\tan}(z_k)$) and Hamiltonian ($\|F(z_k)\|^2$) are both non-monotone for OG :(
- Potential function is directly discovered using SOS Programming
- Formulated by searching over linear combi. of terms:
  - $\|F(z_k)\|^2 - \|F(z_{k+1})\|^2$,
  - $\|F(w_k)\|^2 - \|F(w_{k+1})\|^2$,
  - $\langle F(z_k), F(w_k)\rangle - \langle F(z_{k+1}), F(w_{k+1})\rangle$,
  - $r^{\tan}(z_k)^2 - r^{\tan}(z_{k+1})^2$,
- under the constraints that the linear combi. is non-increasing
- (while maximizing the sum of coeff's of the linear combi., in order to avoid finding trivial linear combi. $\equiv 0$.)

Table of Contents

Additional Remarks

Why SOS?

- PEP & IQC is also a powerful computer-aided proof framework
- However, PEP requires the performance measures and constraints to be polynomials of degree 2 or less; SOS can handle polynomials of higher degrees
- IQC is fitted to prove linear-convergence guarantee, but not very helpful to find an explicit convergence rate (as far as I know)

Alternating GDA: Global convergence is not fully explored yet.

- Local convergence result: Alternating GDA is faster than Simultaneous GDA for SCSC smooth minimax optimization [Zhang et al., 2022]
- Global convergence is partially guaranteed for a specific structured problem ('bilinearly coupled SCSC minimax') using IQC framework
- Can we also have a tight global convergence of GDA with assistance of numerical analyses? Can we even generalize this to sequential multi-player smooth strongly-monotone games?

References I

J. Anderson and A. Papachristodoulou. Advances in computational lyapunov analysis using sum-of-squares programming. *Discrete & Continuous Dynamical Systems-Series B*, 20(8), 2015.

N. Bousselmi, J. M. Hendrickx, and F. Glineur. Interpolation conditions for linear operators and applications to performance estimation problems. *arXiv preprint arXiv:2302.08781*, 2023.

Y. Cai, A. Oikonomou, and W. Zheng. Finite-time last-iterate convergence for learning in multi-player games. In *Advances in Neural Information Processing Systems*, 2022.

Y. Drori and M. Teboulle. Performance of first-order methods for smooth convex minimization: a novel approach. *Mathematical Programming*, 145(1-2):451–482, 2014.

M. Fazlyab, M. Morari, and V. M. Preciado. Design of first-order optimization algorithms via sum-of-squares programming. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 4445–4452. IEEE, 2018.

N. Golowich, S. Pattathil, and C. Daskalakis. Tight last-iterate convergence rates for no-regret learning in multi-player games. *Advances in neural information processing systems*, 33:20766–20778, 2020.

E. Gorbunov, N. Loizou, and G. Gidel. Extragradient method: O (1/k) last-iterate convergence for monotone variational inequalities and connections with cocoercivity. In *International Conference on Artificial Intelligence and Statistics*, pages 366–402. PMLR, 2022.

Z. W. Jarvis-Wloszek. *Lyapunov based analysis and controller synthesis for polynomial systems using sum-of-squares optimization*. University of California, Berkeley, 2003.

G. M. Korpelevich. The extragradient method for finding saddle points and other problems. 1976.

L. Lessard and P. Seiler. Direct synthesis of iterative algorithms with bounds on achievable worst-case convergence rate. In *2020 American Control Conference (ACC)*, pages 119–125. IEEE, 2020.

L. Lessard, B. Recht, and A. Packard. Analysis and design of optimization algorithms via integral quadratic constraints. *SIAM Journal on Optimization*, 26(1):57–95, 2016.

A. Mokhtari, A. Ozdaglar, and S. Pattathil. A unified analysis of extra-gradient and optimistic gradient methods for saddle point problems: Proximal point approach. In *International Conference on Artificial Intelligence and Statistics*, pages 1497–1507. PMLR, 2020.

References II

A. Nemirovski. Prox-method with rate of convergence o(1/t) for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM Journal on Optimization*, 15(1):229–251, 2004. doi: 10.1137/S1052623403425629. URL https://doi.org/10.1137/S1052623403425629.

D. Papp and S. Yildiz. Sum-of-squares optimization without semidefinite programming. *SIAM Journal on Optimization*, 29(1): 822–851, 2019.

L. D. Popov. A modification of the arrow-hurwicz method for search of saddle points. *Mathematical notes of the Academy of Sciences of the USSR*, 28:845–848, 1980.

E. K. Ryu, A. B. Taylor, C. Bergeling, and P. Giselsson. Operator splitting performance estimation: Tight contraction factors and optimal parameter selection. *SIAM Journal on Optimization*, 30(3):2251–2271, 2020.

S. S. Tan, A. Varvitsiotis, and V. Y. Tan. Analysis of optimization algorithms via sum-of-squares. *Journal of Optimization Theory and Applications*, 190(1):56–81, 2021.

W. Tan and A. Packard. Stability region analysis using polynomial and composite polynomial lyapunov functions and sum-of-squares programming. *IEEE Transactions on Automatic Control*, 53(2):565–571, 2008.

A. Taylor, B. Van Scoy, and L. Lessard. Lyapunov functions for first-order methods: Tight automated convergence guarantees. In *International Conference on Machine Learning*, pages 4897–4906. PMLR, 2018a.

A. B. Taylor. *Convex interpolation and performance estimation of first-order methods for convex optimization.* PhD thesis, Catholic University of Louvain, Louvain-la-Neuve, Belgium, 2017.

A. B. Taylor, J. M. Hendrickx, and F. Glineur. Smooth strongly convex interpolation and exact worst-case performance of first-order methods. *Mathematical Programming*, 161:307–345, 2017.

A. B. Taylor, J. M. Hendrickx, and F. Glineur. Exact worst-case convergence rates of the proximal gradient method for composite convex minimization. *Journal of Optimization Theory and Applications*, 178:455–476, 2018b.

T. Yoon and E. K. Ryu. Accelerated minimax algorithms flock together. *arXiv preprint arXiv:2205.11093*, 2022.

M. Zamani, H. Abbaszadehpeivasti, and E. de Klerk. Convergence rate analysis of the gradient descent-ascent method for convex-concave saddle-point problems. *arXiv preprint arXiv:2209.01272*, 2022.

G. Zhang, X. Bao, L. Lessard, and R. Grosse. A unified analysis of first-order methods for smooth games via integral quadratic constraints. *The Journal of Machine Learning Research*, 22(1):4648–4686, 2021.

G. Zhang, Y. Wang, L. Lessard, and R. B. Grosse. Near-optimal local convergence of alternating gradient descent-ascent for minimax optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 7659–7679. PMLR, 2022.