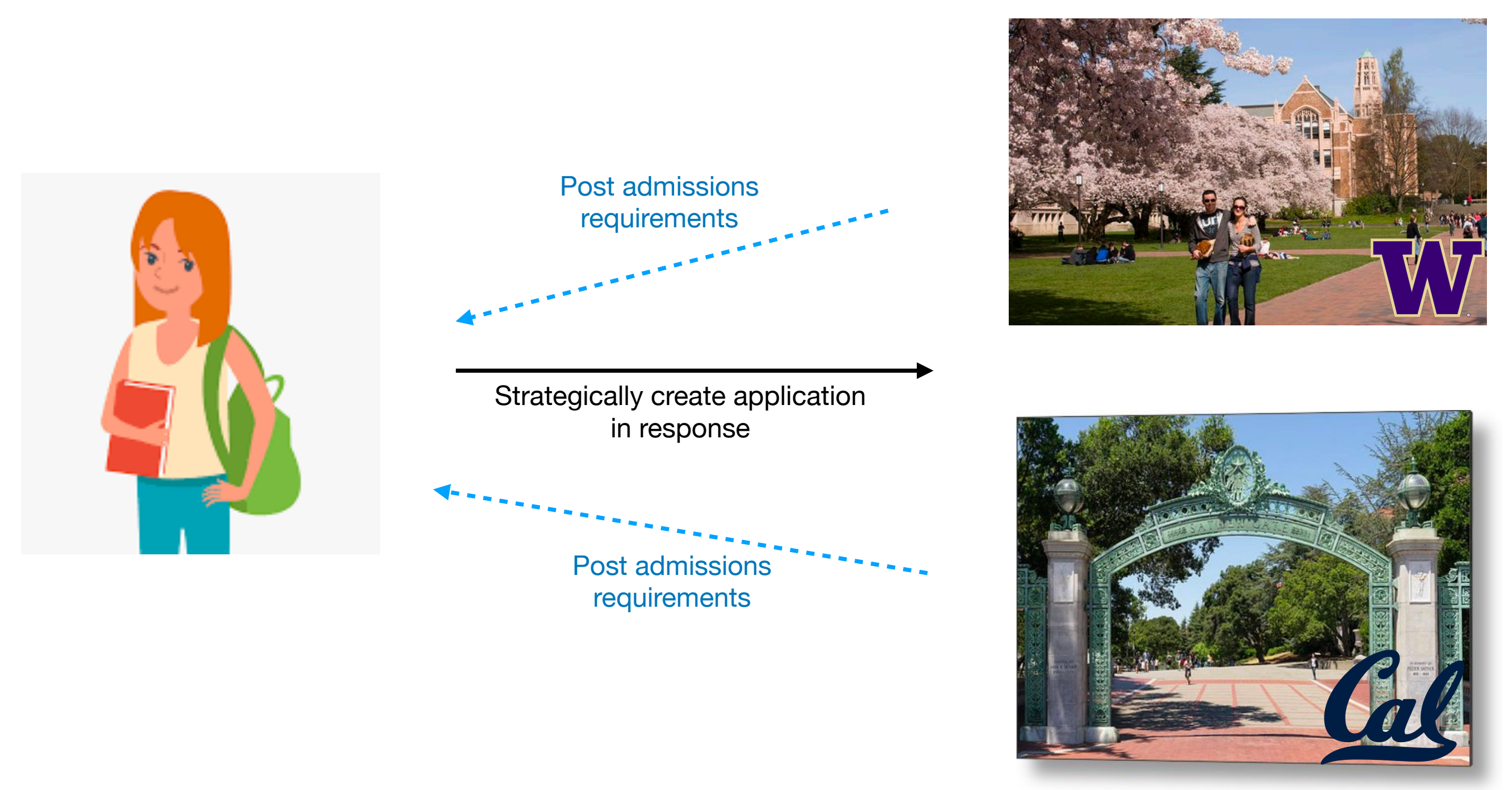
Learning in Stochastic Monotone Games with Decision-Dependent Data

Adhyyan Narang, Evan Faulkner, Dmitriy Drusvyatskiy, Lillian Ratliff, and Maryam Fazel
University of Washington

MOTIVATION AND GOAL

A big assumption of learning theory: Training and test data are drawn from a fixed distribution $\mathcal{D}$.

However, in practice, ML models are deployed in ecosystems with many other agents, and they impact each other's data.


Post admissions requirements
Strategically create application in response
Post admissions requirements

In the above, Cal and UW affect each other's data because their posted admission requirements influence joint applications.

Problem formulation

- Model problem as N-player game. Each player $i \in \{1, \ldots, N\}$ solves

$$\min_{x_i \in \mathcal{X}_i} \mathcal{L}_i(x_i, x_{-i}) \text{ where } \mathcal{L}_i(x) := \mathop{\mathbb{E}}_{z_i \sim \mathcal{D}_i(x)} \ell_i(x, z_i). \tag{1}$$

- Data observed by any agent depends on choices made by *all* agents.
- A tuple of learned parameters $(x_1^*, \ldots, x_n^*) \in \mathcal{X}$ is a Nash Equilibrium for $\mathcal{G} = (\mathcal{L}_1, \ldots, \mathcal{L}_N)$ if, for each $i \in [N]$,

$$x_i^* = \arg\min_{x_i} \mathcal{L}_i(x_i, x_{-i}^*) \tag{2}$$

Goal Design algorithms that can reliably find the Nash strategy for any player with few iterations.

SETUP AND ASSUMPTIONS

Static game Any vector $y \in \mathcal{X}$ induces a static game (without performative effects) $\mathcal{G}(y)$ wherein the distribution for player i is fixed at $\mathcal{D}_i(y)$:

$$\min_{x_i \in \mathcal{X}_i} \mathcal{L}_i^y(x_i, x_{-i}) \qquad \text{where} \qquad \mathcal{L}_i^y(x_i, x_{-i}) := \mathop{\mathbb{E}}_{z_i \sim \mathcal{D}_i(y)} \ell_i(x_i, x_{-i}, z_i). \tag{3}$$

Assumption 1 (Convexity and smoothness). *There exist $\alpha > 0$ and $\beta_i > 0$ such that for each $i \in [n]$: (i) $\forall y \in \mathcal{X}$, the game $\mathcal{G}(y)$ is α-strongly monotone. (ii) Each loss $\ell_i(x_i, x_{-i}, z_i)$ is C^1-smooth in x_i and the map $z_i \mapsto \nabla_i \ell_i(x, z_i)$ is β_i-Lipschitz continuous for any $x \in \mathcal{X}$.*

Assumption 2 (Lipschitz distributions). *For each $i \in [n]$, there exists $\gamma_i > 0$ satisfying*

$$W_1(\mathcal{D}_i(x), \mathcal{D}_i(y)) \le \gamma_i \cdot \|x - y\| \qquad \textit{for all } x, y \in \mathcal{X}.$$

In this case, we define the constant $\rho := \sqrt{\sum_{i=1}^n (\frac{\beta_i \gamma_i}{\alpha})^2}$.

MONOTONICITY

Assumption 3 (Smoothness of the distribution). *For each index $i \in [n]$ and point $x \in \mathcal{X}$, the map $u_i \mapsto \mathbb{E}_{z_i \sim \mathcal{D}(u_i, x_{-i})} \ell_i(x, z_i)$ is differentiable at $u_i = x_i$ and its derivative is continuous in x.*

Theorem 1 (Monotonicity). *Suppose that Assumptions 1–3 hold, and that we are in the regime $\rho < \frac{1}{2}$ and the map $x \mapsto H_x(y)$ is monotone for any y. The game* (1) *is strongly monotone with parameter $(1 - 2\rho)\,\alpha$.*

ALGORITHMS AND GUARANTEES

Derivative Free Method

$$\left\{ \begin{array}{l} \text{Sample } v_i^t \in \mathbb{S}_i \\ \text{Sample } z_i^t \sim \mathcal{D}_i(x^t + \delta v^t) \\ \text{Set } x_i^{t+1} = \operatorname{proj}_{(1-\delta)\mathcal{X}_i} \left(x_i^t - \eta_t \frac{d_i}{\delta} \ell_i(x^t + \delta v^t, z_i^t) v_i^t \right) \end{array} \right\}. \tag{4}$$

Here, $\mathbb{S}_i$ denotes the unit sphere with dimension d_i.

Proposition 1 (Informal). *Under reasonable smoothness and bounded variance assumptions, algorithm* (4) *with appropriately chosen parameters δ and η_t will find a point x satisfying $\mathbb{E}[\|x - x^\star\|^2] \le \varepsilon$ after at most $O(\frac{d^2}{\varepsilon^2})$ iterations.*

The above rate is usually prohibitively slow in practice.

Stochastic Gradient Method

Parametric assumption on distribution family Let $\zeta_i \sim \mathcal{P}_i$ be sampled from a fixed distribution with mean μ_i and covariance Σ_i for each i. $A_i \in \mathbb{R}^{m_i \times d_i}$ and $A_{-i} \in \mathbb{R}^{m_i \times d - d_i}$ are matrices with $d = \sum_{i=1}^N d_i$ and $m = \sum_{i=1}^n m_i$.

$$x_i \sim \mathcal{D}_i(x_i, x_{-i}) \iff z_i = \zeta_i + A_i x_i + A_{-i} x_{-i}.$$

$$\left\{ \begin{array}{l} \text{Sample } z_i^t \sim \mathcal{D}_i(x^t) \\ \text{Set } x_i^{t+1} = \operatorname{proj}_{\mathcal{X}_i} \left(x_i^t - \eta_t \cdot w_i(x^t, z_i^t) \right) \end{array} \right\}. \tag{5}$$

Theorem 2 (Informal). *Under α-strong monotonicity, the above parametric assumption, and simple smoothness and noise assumptions on $\mathcal{D}$, a single step of the stochastic gradient method* (5) *with any constant $\eta \le \frac{\alpha}{2L^2}$ satisfies*

$$\mathbb{E}[\|x^{t+1} - x^\star\|^2] \le \frac{1}{1 + \alpha\eta} \mathbb{E}[\|x^t - x^\star\|^2] + \frac{2\eta^2\sigma^2}{1 + \eta\alpha}, \tag{6}$$

Adaptive Gradient Method

1. Model environment: Estimating $\hat{A}_i$ and $\hat{A}_{-i}$ to obtain $\hat{\mathcal{D}}_i$.
2. Choose actions: Find the "approximate Nash" using gradient play, assuming $\mathcal{D}_i = \hat{\mathcal{D}}_i$.

Theorem 3 (Informal). *Adaptive algorithm with appropriately chosen parameters δ and η_t will find a point x satisfying $\mathbb{E}[\|x - x^\star\|^2] \le \varepsilon$ after at most $O(\frac{d}{\varepsilon})$ iterations.*

SIMULATIONS

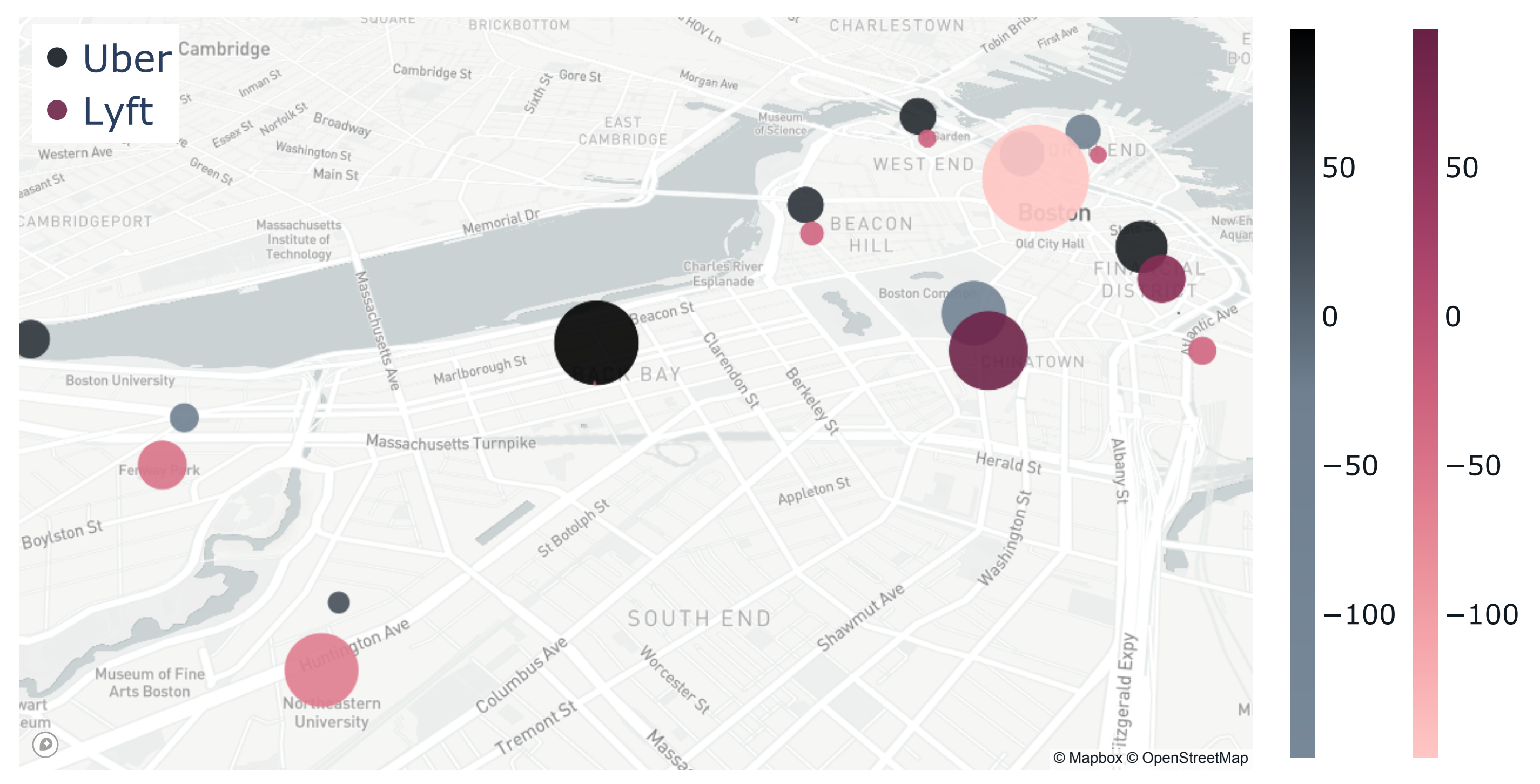


Fig. 1: Changes in revenue at Nash for both companies across 11 locations in Boston

- Simulate game between Uber and Lyft based on real data collected in Boston, MA.
- Consider 11 locations across city where rides originate. Companies adjust prices at each location,
- This affects the demand that they both experience. Demand z_i seen by company i is given by

$$z_i = \zeta_i + A_i x_i + A_{-i} x_{-i}$$

ζ_i is a fixed empirical distribution of demands from the dataset, x_i and x_{-i} are the price adjustments from a nominal value by each company. $A_i \preceq 0$ and $A_{-i} \succeq 0$ are matrices that represent the price elasticities of the rides.

- Company i's loss is

$$\ell_i(x_i, z_i) = -\frac{1}{2} z_i^\top x_i + \frac{\lambda_i}{2}\|x_i\|^2,$$

- Changes in revenue from the real-world revenue at Nash are shown in Figure 2.
- Distance from Nash of the iterates is shown in Figure 3. As expected, DFO performs worse than adaptive or stochastic gradient methods, which are comparable.

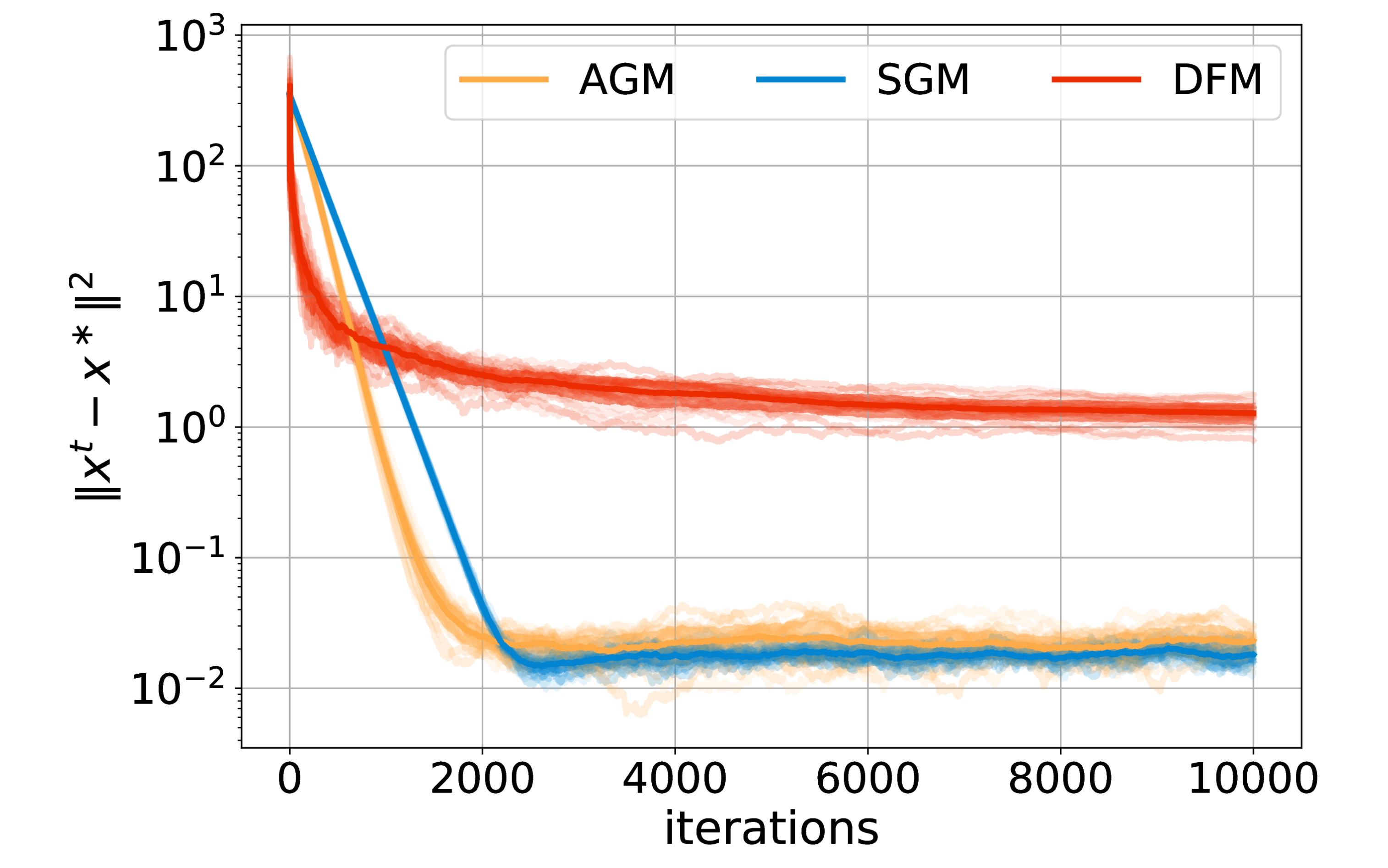


Fig. 2: Distance from iterates to Nash for each of the three algorithms