

Two-Timescale Learning in Stochastic Games: Convergence, Equilibrium Selection, and Algorithmic Collusion

Nan Chen

Systems Engineering & Engineering Management
The Chinese University of Hong Kong

World Congress of Bachelier Finance Society
Bologna, Italy
July 1, 2026

Based on a joint work with Ruixun Zhang (Peking University), Yumin Xu (Peking University), Mingyue Zhong (CUHK)

Outline

- 1 Algorithmic Collusion
- 2 Multiagent Reinforcement Learning (MARL)
- 3 A Two Timescale Evolutionary Game Approach
- 4 Convergence Analysis
- 5 Numerical Experiments

Outline

- 1 Algorithmic Collusion
- 2 Multiagent Reinforcement Learning (MARL)
- 3 A Two Timescale Evolutionary Game Approach
- 4 Convergence Analysis
- 5 Numerical Experiments

Algorithmic Pricing and Collusion

Although the use of pricing algorithms has a long history, the current generation of **AI-powered algorithms** is different in the following two aspects:

- **Autonomous learning**

Algorithms had once been based on pricing rules set by programmers, but now often rely on autonomous learning through active experimentation.

- **Real-time observation**

The wide availability of competitors' prices online enables AI-powered algorithms to quickly detect price changes and thus incorporate them in decision making.

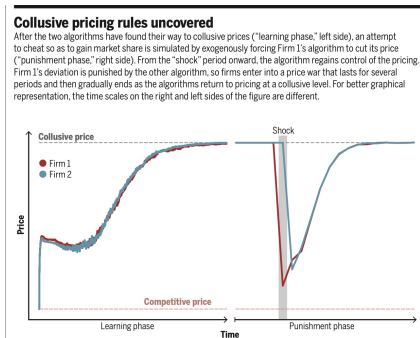
Tacit interaction through price becomes a reality.

It is Already in the Market: RealPage

- David Topkins (2015) and E-Turas (2016)
- RealPage (2024 to Now)
 - **Background** RealPage Inc., a Texas based revenue-management platform supplier, sells property pricing software that recommends rents to landlords. The software allows to use competitors' **nonpublic pricing and occupancy data**. Its services are used in pricing 24 million housing units worldwide.
 - **Accusation** The Department of Justice (DOJ), several states, and renter plaintiffs allege this let competing landlords coordinate rents through a common algorithm. "**Hub-and-spoke conspiracy**"
 - **Settlements & DOJ Remedy** DOJ settled (Nov 2025): RealPage must stop using nonpublic competitor data under a court monitor. In addition, a group of landlords and property management companies that use the RealPage service have paid roughly \$360 million to settle class-actions.

A Deeper Puzzle: The Study of Calvano et al. (2020, AER)

- RealPage is a shared algorithm. It is supposed to coordinate pricing in the center.
- But, what if two independent learning algorithms competing in the same market?
- Calvano et al. (Science 2020, AER 2020) show that they can learn to collude



A Deeper Puzzle: The Study of Calvano et al. (2020, AER)

Classical oligopoly theory predicts

- One-shot game: Bertrand–Nash is the unique equilibrium — competitive prices.
- Collusion survives only in the **repeated** game.
- Folk-theorem logic presumes **fully rational** and **forward-looking** players.

Friedman (1971, RES), Fudenberg and Maskin (1986, Econometrica), Abreu (1986, JET), Abreu (1988, Econometrica), Green and Porter (1984, Econometrica)

diverge

Algorithms do

- **Simple, myopic** learners.
- Independent — no shared platform/algorithms
- No communication, no instruction in advance.

⇒ yet they reach **supracompetitive, reward–punishment** outcomes.

Calvano et al. (2020, AER), Klein (2021, RAND Journal)

Where does the coordination come from?

Belief: Focus of Our Paper

- **What is a “belief” in an AI system?**

- An internal assumption, probability weight, or representation of data that an AI system treats as true to guide its actions and reasoning.

- **Beliefs drive the outcome of AI agent interaction.**

- The principled response under a belief system:

form beliefs about the environment and rivals $\Rightarrow$ best-response
 $\Rightarrow$ observe consequences $\Rightarrow$ update beliefs

- *Example.*

- The Belief–Desire–Intention model (Rao and Georgeff, 1991) combines *beliefs* (a model of the world and itself), *desires* (goals), and *intentions* (committed plans) to produce actions.
- Pricing systems in the German retail gasoline market are largely built on the BDI architecture (Assad et al., 2024, *JPE*).

Belief: What the Literature Misses — and What We Do

Calvano et al. (2020) leave the belief channel implicit.

- Tabular Q-learning stores *action values*; it never models the opponent.
- The state includes the rival's past choices, so an agent *can* condition on rival behavior — yet beliefs are never made explicit.
- Pure **simulation**: shows *that* collusion happens, not *why*; Q-learning convergence theory **breaks** once rivals also learn.

This paper: make the belief channel explicit

- Each firm holds a **posterior** over its rival's strategy and **updates it recursively** as prices arrive .
- Coordination **emerges** from these coupled belief dynamics — not from communication.

Regulatory Responses

- **US:** Federal Trade Commission (FTC) conducted a series of hearings from 2018 to 2019 on competition and consumer protection issues associated with the use of AI-powered algorithms.
- **OECD:** The Competition Committee OECD organized roundtable meetings to investigate the risk of algorithm tacit collusion in 2017.
- **EU:** Digital Service Act (2024) require algorithmic audits to ensure compliance with competition rules.
- **China:** State Council released an anti-monopoly guideline to regulate platform competition in 2021.

- **Foundational simulation evidence.** Calvano et al. (*AER* 2020; *Science* 2020); Klein (*RAND* 2021); Johnson et al. (*Econometrica* 2023).
- **Empirical / field evidence.** Assad et al. (*JPE* 2023).
- **Algorithm design and market dynamics.** Cartea et al. (*QF* 2022, 2023, 2024); Xiong and Cont (*ICAIF* 2021); Loots and den Boer (*POMS* 2022); Abada and Lambin (*MS* 2023); Dou, Goldstein, and Ji (2024); Colliard et al. (2025, RFS)
- **Collusion under large language models:** Fish et al. (2024); Keppo et al. (2025); Cartea et al. (2026)

Outline

- 1 Algorithmic Collusion
- 2 Multiagent Reinforcement Learning (MARL)
- 3 A Two Timescale Evolutionary Game Approach
- 4 Convergence Analysis
- 5 Numerical Experiments

MARL Setup

We can frame the study of algorithmic collusion as a **multi-agent reinforcement learning (MARL)** problem:

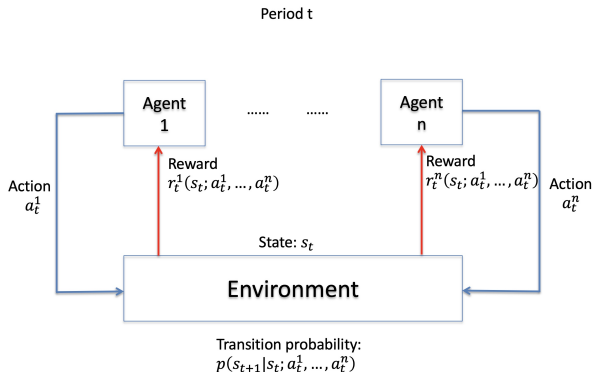


Figure 1: MARL Setup

MARL Setup (Continued)

- Time horizon: $t = 0, 1, \dots$
- Agents: $1, 2, \dots, n$
- Actions: each agent selects action $a_t^k \in \mathbb{A}^k$ at time t , $k = 1, \dots, n$
- System evolves from s_t to s_{t+1} according to the transition probability

$$p(s_{t+1} | s_t, a_t^1, \dots, a_t^n), \quad \forall t$$

- Rewards: Agent k obtains $r^k(s_t, a_t^1, \dots, a_t^n)$, $k = 1, \dots, n$, at time t .

Each agent's reward depends on *all* agents' actions $\Rightarrow$ inherently strategic.

Nash Equilibrium

- Agents play **mixed (stationary) policies**: agent k draws actions from $\mathbb{A}^k$ according to a distribution.
- Let $\pi_k(s)$ be agent k 's mixed policy at state s , and $\pi = (\pi_1, \dots, \pi_n)$. The expected discounted payoff of agent k is

$$v_{\pi}^k(s) = \mathbb{E}_{\pi} \left[\sum_{t=0}^{\infty} \beta^t r^k(s_t, a_t^1, \dots, a_t^n) \mid s_0 = s \right],$$

where the actions $a_t^k \sim \pi_k(s_t)$ and $\beta \in (0, 1)$ is the discount factor.

Nash Equilibrium (Continued)

Definition (Nash Equilibrium / Markov Perfect Equilibrium)

A Nash equilibrium is a collection of policies $\pi^* = (\pi_1^*, \dots, \pi_n^*)$ such that

$$v_{(\pi_k^*, \pi_{-k}^*)}^k(s) \geq v_{(\pi_k, \pi_{-k}^*)}^k(s), \quad \forall k \in \{1, \dots, n\}, s \in \mathbb{S}, \pi_k \in \Pi_k,$$

where $\pi_{-k}^* = (\pi_1^*, \dots, \pi_{k-1}^*, \pi_{k+1}^*, \dots, \pi_n^*)$.

No agent can improve by unilaterally deviating—our target benchmark for what algorithms might (or might not) converge to.

Q-learning: Single-Agent RL

- The algorithmic-collusion literature predominantly uses **Q-learning** to simulate pricing algorithms.
- Q-learning centers on the **Q-value** $Q(s, a)$: the estimated value of taking action a in state s . The optimal Q-value satisfies

$$Q^*(s, a) = \mathbb{E}[r(s, a)] + \beta \sum_{s'} v^*(s') p(s' \mid s, a),$$

where $v^*(s)$ is the agent's optimal value function.

Q-learning: Single-Agent RL (Continued)

We learn Q^* via the **Q-iteration**. At step l :

- 1 Choose an action greedily (with random perturbation for exploration):

$$a^l = \arg \max_a Q^l(s, a)$$

- 2 Take a^l ; observe the next state s' and reward $r_l(s, a)$.
- 3 Update the Q-value:

$$Q^{l+1}(s, a^l) = (1 - \alpha^l) \underbrace{Q^l(s, a)}_{\text{past experience}} + \alpha^l \underbrace{\left[r_l(s, a) + \beta \max_{a'} Q^l(s', a') \right]}_{\text{new observation}}$$

where α^l is the learning rate at iteration l .

Q-learning in the Multi-Agent Setup

- However, plain Q-learning **cannot capture strategic interaction** among agents:
 - Q-learners are unaware of the presence of opponents.
 - It is well known in the machine learning literature that Q-learning-based methods are numerically instable in multi-agent settings (Lowe et al. (2017))
 - Colliard, Foucault, and Lovo (2025, RFS) establish an **impossibility theorem** for the convergence of Q-matrices in the multi-agent setup.

Calvano et al. (2020, AER) likewise report that a fraction of their sample paths never converge.

Nash Q-learning

- **Nash Q-learning** extends Q-learning to explicitly model inter-agent interaction (Hu and Wellman (2003), Liu et al. (2021, ICML), Li et al. (2025, IEEE TNSE))
- It centers on the **Nash Q-value function**

$$Q_*^k(s, a_1, \dots, a_{k-1}, \textcolor{red}{a}_k, a_{k+1}, \dots, a_n).$$

- Update for agent k :

$$Q_{l+1}^k(s_l; a_1^l, \dots, a_n^l) = (1 - \alpha^l) Q_l^k(s_l; a_1^l, \dots, a_n^l) + \alpha^l [r_l^k + \beta \text{Nash} Q_l^k(s')]$$

- Computing $\text{Nash} Q_l^k(s')$ requires solving a **stage-game Nash equilibrium** at each step, using $\{Q_l^k(s, a^1, \dots, a^n) : k = 1, \dots, n\}$ as the payoff functions of agents.

Nash Q-learning: Obstacles for Collusion Studies

- Several obstacles for Nash Q-learning algorithm when applied for collusion studies:
 - **Computational cost:** solving a Nash equilibrium at every step is expensive.
 - **Observability:** requires observing rivals' Q-values — unsuitable for decentralized decision-making.
 - **No general convergence guarantee:** Hu and Wellman (2003) prove convergence only under restrictive conditions (**global optimum** or **saddle-point** stage games).

These gaps motivate our **two-timescale evolutionary game** approach.

Literature Review (An Incomplete List)

- Cooperative game:
 - Boutilier (1996), Lauer and Riedmiller (2000), Monderer and Shapley (1996), González-Sánchez and Hernández-Lerma (2013), ...
- Competing setup/Zero-sum game:
 - Robinson (1951), Hoffman and Karp (1966), Littman (1994), Hofbauer and Sandholm (2002), Zinkevich et al. (2008), Benaïm and Faure (2013), Hansen et al. (2013), Silver et al. (2017), ...
- **Non-zero sum game (with a fixed number of agents):**
 - Hu and Wellman (2003), Greenwald et al. (2003), ...
 - More suitable for algorithmic collusion studies
- Mean-field game:
 - Huang et al. (2003, 2006), Lasry and Lions (2007), Bensoussan et al. (2013), Fu and Horst (2017, 2020, 2023), Carmona and Delarue (2019), Guo et al. (2019, 2020), Angiuli et al. (2021), Barrasso and Touzi (2022), Talbi et al. (2023), Djete and Touzi (2023), Carmona, Cormier, and Soner (2023), ...

Outline

- 1 Algorithmic Collusion
- 2 Multiagent Reinforcement Learning (MARL)
- 3 A Two Timescale Evolutionary Game Approach
- 4 Convergence Analysis
- 5 Numerical Experiments

Contributions of Our Work

We propose a **decentralized two-timescale evolutionary game** approach. Our contributions are three-fold.

- **Modeling.** The learning dynamics enjoy several features appealing for algorithmic-collusion studies:
 - general-sum game;
 - requires *no* observability of rivals' Q-values;
 - heterogeneous updating rates;
 - heterogeneous levels of intelligence.
- **Convergence guarantee**, even with multiple equilibria.
- **Numerical experiments** show that collusion is contingent rather than universal, and we isolate several factors that govern it:
 - 1 Algorithm initialization
 - 2 Belief and sophistication of the algorithms.
 - 3 Market structure: demand volatility, the number of competitors, adverse selection, etc.

Objectives of the Algorithm

Each agent k ($k = 1, \dots, n$) maintains and learns:

- **Mixed policy** $X^k(s)$: the probabilities with which agent k plays each action at state $s \in \mathbb{S}$.
- **Belief** $\hat{X}^{k,j}(s)$: agent k 's estimate of the mixed policy played by agent j ($j \neq k$) at state s .
- **Q-value** $Q^k(s, a_1, \dots, a_n)$.

Superscript convention: $\hat{X}^{\text{owner, target}}$, so $\hat{X}^{k,j}$ is the belief *held by k about j* .

Key Components of Our Algorithm

- **Perturbed best-response** rule for policy learning.
- Each agent uses **fictitious play** to update its beliefs about opponents' strategies.
- **Two timescales** separating policy/belief learning (slow) from Q-value learning (fast).

The timescale separation is the key idea: from the fast Q-update's viewpoint, opponents' policies look essentially frozen — which tames the non-stationarity of MARL.

Evolutionary Learning Algorithm

Steps 0–3 — Action Sampling and Transition

Iteration l . At the start, each agent k observes the state s_l and holds

Q-value: $\{Q_l^k(s, a_1, \dots, a_n) : (a_1, \dots, a_n) \in \mathbb{A}\},$

policy: $\{X_l^k(s) : k = 1, \dots, n, s \in \mathbb{S}\},$

beliefs: $\{\hat{X}_l^{k,j}(s) : j \neq k, s \in \mathbb{S}\}.$

- 1 Each agent k samples an action a_l^k from $X_l^k(s_l)$.
- 2 Apply the joint action $(a_l^1, \dots, a_l^n)$; observe the next state s' and rewards $\{r_l^k : k = 1, \dots, n\}.$
- 3 Set $s_{l+1} \leftarrow s'.$

Evolutionary Learning Algorithm

Step 4 — Perturbed Best-Response (Policy)

Each agent k computes the belief-weighted value of playing b :

$$F_l^k(s_l, b) = \sum_{a_{-k}} \underbrace{\left(\prod_{j \neq k} \hat{X}_l^{k,j}(s_l, a_j) \right)}_{\text{belief on others}} \underbrace{Q_l^k(s_l, a_1, \dots, b, \dots, a_n)}_{\text{own Q-value, } b \text{ in slot } k}$$

for all $b \in \mathbb{A}^k$. Then choose

$$b_l^k = \arg \max_{b \in \mathbb{A}^k} \{F_l^k(s_l, b) + \varepsilon^k(b)\},$$

where $\{\varepsilon^k(b)\}$ are i.i.d. **Gumbel**:

$$\mathbb{P}(\varepsilon^k(b) \leq c) = \exp(-\exp(-\eta^{-1}c - \gamma)), \quad \forall c \in \mathbb{R},$$

for scale $\eta > 0$ ($\gamma = \text{Euler-Mascheroni constant}$).

Quantile response (Thanks to S. Jamungal for letting me know this term)

Evolutionary Learning Algorithm

Step 5 — Policy Weight Update (First Timescale)

Shift probability mass from the played action a_l^k toward the best response b_l^k :

$$X_{l+1}^k(s, a) = \begin{cases} X_l^k(s, a) + \frac{1}{N_l^k}, & s = s_l, a = b_l^k, \\ X_l^k(s, a) - \frac{1}{N_l^k}, & s = s_l, a = a_l^k, \\ X_l^k(s, a), & \text{otherwise.} \end{cases}$$

(First timescale) The policy learning rate is $\{1/N_l^k : l = 0, 1, \dots\}_{1 \leq k \leq n}$.

Evolutionary Learning Algorithm

Step 6 — Belief Update via Fictitious Play

Having observed a_l^k , every other agent j updates its belief $\hat{X}^{j,k}$ about agent k :

$$\hat{X}_{l+1}^{j,k}(s, \cdot) = \begin{cases} \hat{X}_l^{j,k}(s, \cdot) + \frac{1}{N_l^k} \left(\pi_{s_l}^{\text{pure}}(a_l^k) - \hat{X}_l^{j,k}(s, \cdot) \right), & s = s_l, \\ \hat{X}_l^{j,k}(s, \cdot), & \text{otherwise.} \end{cases}$$

Beliefs share the *same* (first-timescale) rate as the policy.

This is recursive **fictitious play** (Brown 1951; Robinson 1951): the empirical frequency

$$\hat{F}_{m+1}(a) = \frac{1}{m+1} \sum_{i=1}^{m+1} \mathbf{1}_{\{a_i=a\}} = \hat{F}_m(a) + \frac{1}{m+1} (\mathbf{1}_{\{a_{m+1}=a\}} - \hat{F}_m(a)).$$

we generalize the learning rate from $1/(m+1)$ to $1/N$.

Evolutionary Learning Algorithm

Steps 7–8 — Q-Value Update (Second Timescale)

- 7 Using the realized transition $(s_l, a_l^1, \dots, a_l^n) \rightarrow s_{l+1}$, each agent k forms the temporal-difference target

$$y_l^k = r_l^k + \beta \sum_{b \in \mathbb{A}^k} X_{l+1}^k(s_{l+1}, b) F_{l+1}^k(s_{l+1}, b),$$

where $\beta \in (0, 1)$ is the discount factor and F_{l+1}^k is the belief-weighted value from Step 4 (evaluated at the new state s_{l+1}).

- 8 Update the Q-value at the visited entry only:

$$Q_{l+1}^k(s, \mathbf{a}) = \begin{cases} Q_l^k(s, \mathbf{a}) + \alpha_l^k (y_l^k - Q_l^k(s, \mathbf{a})), & (s, \mathbf{a}) = (s_l, \mathbf{a}_l), \\ Q_l^k(s, \mathbf{a}), & \text{otherwise,} \end{cases}$$

with $\mathbf{a} = (a_1, \dots, a_n)$ and $\mathbf{a}_l = (a_l^1, \dots, a_l^n)$.

(Second timescale) The Q-learning rate $\{\alpha_l^k\}$ is *fast* relative to the policy rate:

$$\frac{1}{N_l^k} = o(\alpha_l^k) \quad \text{as } l \rightarrow \infty.$$

Features of the Learning Dynamics

- **Simplified Computation:** Unlike Nash Q-learning, our algorithm avoids the need for complex Nash equilibrium computations.
- **Decentralized.** Each agent uses only its *own* rewards and Q-values, plus observations of others' actions. No agent needs access to rivals' Q-values.
- **Two timescales.** Q-values learn fast; policies and beliefs learn slowly. From the fast process's viewpoint, opponents' policies appear *frozen* — which tames the non-stationarity that plagues MARL.

Outline

- 1 Algorithmic Collusion
- 2 Multiagent Reinforcement Learning (MARL)
- 3 A Two Timescale Evolutionary Game Approach
- 4 Convergence Analysis**
- 5 Numerical Experiments

ϵ -Nash Equilibrium

Exact Nash equilibria are computationally intractable (**PPAD-hard**, even for the one-shot game).

We therefore target to show the convergence to an *approximate* equilibrium.

Definition (ϵ -Nash equilibrium)

Fix $\epsilon > 0$. A joint policy profile $X_* = (X_*^1, \dots, X_*^n)$ is an ϵ -Nash equilibrium if no agent can gain more than ϵ by deviating:

$$v_{(X_*^k, X_*^{-k})}^k(s) \geq v_{(X^k, X_*^{-k})}^k(s) - \epsilon \quad \forall k, s, X^k.$$

Convergence: The Two-Timescale Idea

The learner couples three quantities: policies X , beliefs $\hat{X}$, and Q-values Q . We decouple them by speed.

- **Fast timescale — evaluation.** With policies essentially frozen, Q tracks the value of the current profile: $Q_t \approx \lambda(X_t, \hat{X}_t)$.
- **Slow timescale — adaptation.** Beliefs become consistent ($\|\hat{X}_t - X_t\| \rightarrow 0$) and the policy then follows a smooth ODE:

$$\dot{X} = f(X, X). \quad (9)$$

Takeaway

The stochastic algorithm *tracks the ODE* (9); its limit points are the rest points of f — the ϵ -Nash equilibria.

Tools: multi-timescale stochastic approximation with controlled Markov noise (Karmakar–Bhatnagar 2018, MOR) + quasi-Fejér analysis (Duvocelle et al. 2023, MOR; Mertikopoulos et al. 2024, Math Program). Full recursions. [▶ See derivation](#)

Assumptions

- **Regularity (standard).** Finite state/action spaces; bounded rewards; irreducible dynamics; two-timescale step-size conditions (Q updated faster than policies).
- **Local stability (key).** The target rest point X_* of (9) is *linearly stable* — the Jacobian of f at X_* has all eigenvalues with negative real part.

Convergence Theorem

Theorem (Chen et al. (2026))

*Under the above assumptions, for any $\epsilon > 0$, $p > 0$, there exist a small η , a large c , and a **neighborhood $\mathcal{W}$ of X_*** such that, whenever the algorithm starts in $\mathcal{W}$,*

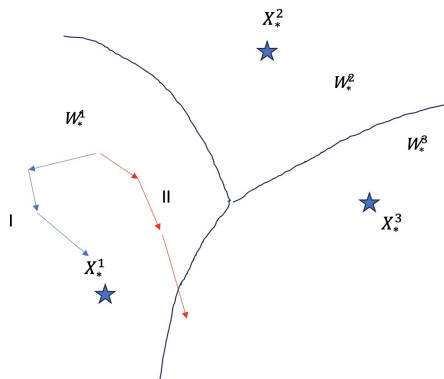
$$\mathbb{P}\left(\lim_{l \rightarrow \infty} (X_l^\eta, Q_l^\eta) = (X_*, Q_*)\right) > 1 - p,$$

where X_ is an ϵ -Nash equilibrium and Q_* its Nash Q-value.*

This is a local convergence guarantee: convergence to X_* holds for starting points near X_* , with high probability.

The Case of Multiple Equilibria

- Our result is about the local convergence property.
- When there exist a multiple stationary points for the system (5), the initial point largely determines which approximate Nash-equilibrium can be eventually found by the algorithm



Some Special Cases (Ongoing Work)

- In general, local stability of X_* must be assumed.
- For the following classes, we can establish the negative-eigenvalue condition directly,
 - Zero-sum games
 - Potential games
- Two-timescale framework in mean field type games
 - Chen, Lauriere and Zhong, "A two-time scale approach for mean field type games", Presented in BFS Session MS_426, June 30.

Theoretical Contributions of Our Work

- A comparison with recent literature on decentralized MARL

	Multi-timescale	Controlled Markov Noise	Game Type	Knowledge of the opponent
Ours	✓	✓	General-sum	Fictitious play
Sayin et al. (2022, SIAM J. Control Optim)	✓	×	Zero-sum	Fictitious play
Maheshwari et al. (2022)	✓	×	Potential	Local Q-function
Sayin et al. (2021, NeurIPS)	✓	✓	Zero-sum	Local Q-function
Yongacoglu et al. (2023)	✓	✓	Weakly Acyclic	Local Q-function

Outline

- 1 Algorithmic Collusion
- 2 Multiagent Reinforcement Learning (MARL)
- 3 A Two Timescale Evolutionary Game Approach
- 4 Convergence Analysis
- 5 Numerical Experiments**

Numerical Experiment I: Bertrand Oligopoly Competition

Setup

- n competing firms sell substitute products in a common market environment (see, e.g., Calvano et al. (2020), den Boer et al. (2023)).
 - Time is indexed by $t \in \mathbb{N}$. At the beginning of each time period, a group of customers appear to demand the products of these firms.
 - In t -th period, firms choose price $\mathbf{p}_t = (p_t^1, \dots, p_t^n)$ and immediately receives a payoff

$$r_t^i = r^i(\mathbf{p}_t) = (p_t^i - c_i) \cdot d_i(\mathbf{p}_t),$$

where c_i represents the marginal cost and the demand for product i is given by a perturbed multinomial logit model, i.e.,

$$d_i(\mathbf{p}_t) = \frac{\exp\left(\frac{a_i - p_t^i}{\mu}\right)}{\exp\left(\frac{a_0}{\mu}\right) + \sum_{j=1}^n \exp\left(\frac{a_j - p_t^j}{\mu}\right)} + \sigma \epsilon_t.$$

a_i , $i = 1, \dots, n$: vertical differentiation of products i , μ : horizontal differentiation, a_0 : for external product, $\sigma \epsilon_t$: demand shocks.

Numerical Experiment I: Bertrand Oligopoly Competition

Setup (continued)

- For benchmarking, we compute *monopolistic price* and *Bertrand-Nash competition price* from one-shot game.

- Monopolistic prices:

$$\mathbf{p}_M = (p_M^1, \dots, p_M^n) = \arg \max_{\mathbf{p}=(p^1, \dots, p^n)} \sum_{i=1}^n \mathbb{E}[r^i(p^i, \mathbf{p}^{-i})].$$

- Bertrand-Nash equilibrium prices:

$$p_N^i = (p_N^1, \dots, p_N^n) = \arg \max_{p^i} \mathbb{E}[r^i(p^i, \mathbf{p}_N^{-i})].$$

- Both prices are explicitly known (Li and Huh (MSOM, 2011))

► Expression

Numerical Experiment I: Bertrand Oligopoly Competition

Repeated Prisoner's Dilemma

Symmetric case, $n = 2$

- Parameters $a_1 = a_2 = 2$, $c_1 = c_2 = 1$, $a_0 = 0$, $\mu = 0.25$ give

$$p_N = 1.4729 \text{ (D)}, \quad p_M = 1.9250 \text{ (C)}$$

- Per-period reward (a Prisoner's-Dilemma structure):

(r^1, r^2)	C	D
C	(0.3375, 0.3375)	(0.1180, 0.3679)
D	(0.3679, 0.1180)	(0.2229, 0.2229)

- (C, C) is jointly optimal (*collusion*); (D, D) is the unique pure Nash of the *one-shot* game.
- Each firm's objective $\sum_{t=0}^{+\infty} \mathbb{E}[\beta_i^t r_t^i(\pi_t^1, \pi_t^2)]$
- Short memory: both agents can observe and remember what both of them played in the last period. Take the actions of the last round as the state of this game.

Numerical Experiment I: Bertrand Oligopoly Competition

Repeated Game and Multiple Equilibria

- Our algorithm recovers **three** pure-strategy Nash equilibria (matching the brute-force search of den Boer et al. (2023)):

	(C, C)	(C, D)	(D, C)	(D, D)
All Defect (AD)	D	D	D	D
Grim Trigger (GT)	C	D	D	D
Win-Stay-Lose-Shift (WSLS)	C	D	D	C

- GT and WSLS sustain **collusion** via a reward–punishment scheme; All-Defect is the static outcome.

Numerical Experiment I: Bertrand Oligopoly Competition

Initialization Determines the Outcome

- Starting near P_{AD}

⇒ **defection**.

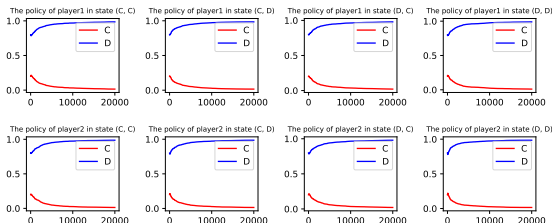
- Starting near P_{GT} or

P_{WSLS} ⇒ **collusion**,

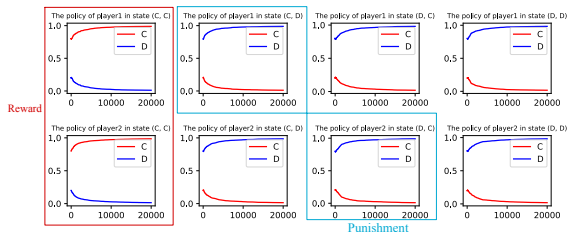
enforced by:

- **Reward** on the cooperative path,
- **Punishment** after a deviation.

- Same algorithm, same game — only the starting policy differs.



Convergence to AD



Convergence to GT

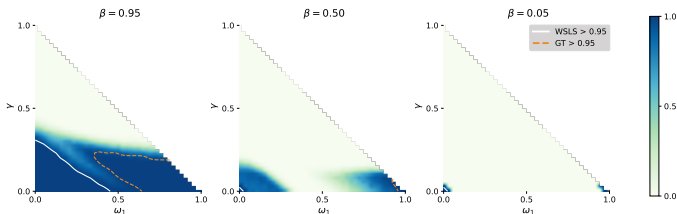
Numerical Experiment I: Bertrand Oligopoly Competition

How Robust Is Initialization-Dependence?

- Initialize on the simplex of the three equilibrium policies:

$$\text{Initial policy} = \gamma E_{AD} + \omega_1 E_{GT} + \omega_2 E_{WSLS}, \quad \gamma + \omega_1 + \omega_2 = 1.$$

- Proportion of paths converging to collusion (GT/WSLS) across initial policies and discount factors β . White / dashed contours enclose regions where $\geq 95\%$ converge to WSLS / GT.



Numerical Experiment I: Bertrand Oligopoly Competition

Regulatory Implications

- Main takeaway: **Initialization is not a neutral technicality.**
 - Initialization selects the equilibrium.
 - And firms rarely initialize at random — they calibrate to **shared historical data**.

Common data $\Rightarrow$ correlated initial conditions $\Rightarrow$
tacit coordination on the collusive basin,
without any communication or intent.

Regulatory implications

- **Intent-based doctrine of traditional antitrust law in AI age?** A "meeting of minds" that antitrust law traditionally targets is simply not presented here.
- **Auditing inputs?** If initialization drives the outcome, then *shared data sources, common pricing vendors, and identical calibration practices* become the relevant risk factors.

Numerical Experiment I: Bertrand Oligopoly Competition

Level of Sophistication: A New Axis

- We vary the agent's sophistication in the experiments. Consider two types of agent
 - **Naive learner:** treats the environment as a single-agent problem — ignores rivals when updating values and choosing actions. Our algorithm can model such learner by
 - (i) *Frozen beliefs:* opponent beliefs fixed at uniform; belief-learning rate set to zero.
 - (ii) *Opponent-independent updates:* policy and Q -values updated *blockwise* on the agent's own lagged price only.
 - **Sophisticated learner:** observes rivals, forms beliefs, and conditions on opponents' past actions (the Section 3 algorithm).

Key distinction

The state still records rivals' prices — but the naive agent *acts as if only its own pricing history were payoff-relevant*.

Numerical Experiment I: Bertrand Oligopoly Competition

Level of Sophistication: A New Axis (Continued)

- Two agents with different level of sophistication play with each other:

Table 1: converging to collusion or not

	Naive	Sophisticated
Naive	Never	Never
Sophisticated	Never	Possible

- Define

$$\Delta = \frac{\bar{r} - \bar{r}(\mathbf{p}_N)}{\bar{r}(\mathbf{p}_M) - \bar{r}(\mathbf{p}_N)}, \quad (1)$$

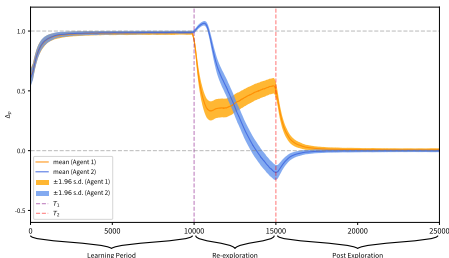
where $\bar{r}$ represents the empirical average reward obtained by the agent on the sample paths.

- Values of $\Delta > 0$ indicate the emergence of super-competitive payoffs on sample paths.

Numerical Experiment I: Bertrand Oligopoly Competition

Naive vs. Naive: Collusion is Fragile

- Three phase of learning:
 - *Learning phase* ($0-T_1$):
collusion-biased init $\Rightarrow$
 $\Delta_{T_1} > 0$.
 - *Re-exploration* (T_1-T_2):
firm 2 is forced to explore by
increasing η from 0.01 to 3. It
finds D dominates in its Q
value
 - *Post exploration* (T_2+):
Return η back to 0.01.
Learning leads to
Bertrand-Nash, $\Delta \rightarrow 0$.



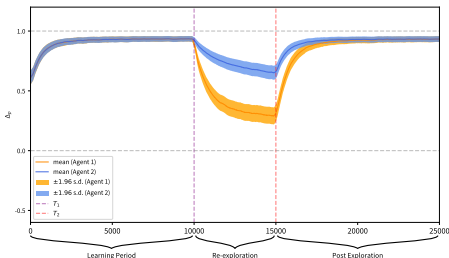
Takeaway

Without opponent modeling, seemingly collusive behavior is a **metastable artifact of imperfect exploration** — not a self-sustaining equilibrium.

Numerical Experiment I: Bertrand Oligopoly Competition

Sophisticated vs. Sophisticated: Robust Collusion

- Sophisticated vs. Sophisticated players:
 - Same shock applied after both firms converge to **WSLS** by T_1 .
 - Firm 2's Q value of D in state (C, C) rises but soon slows, never overtakes C.
 - Reason: defecting in (C, C) **triggers retaliation**
 - When the shock ends, firms **return to the collusive equilibrium**.



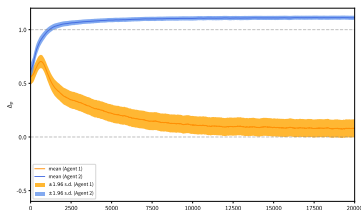
Takeaway

Opponent characterization enables sustaining collusion and make it survive perturbations.

Numerical Experiment I: Bertrand Oligopoly Competition

Naive vs. Sophisticated: Exploitation

- Asymmetric pairing between sophisticated and naive agents yields **one-sided exploitation**, not mutual collusion.
 - The sophisticated agent converges to a *randomized policy* that:
 - keeps the naive agent choosing C (profit above competitive zero);
 - occasionally switches to D to exploit that predictability.
 - Result: $\Delta > 1$ (sophisticated), $\Delta \approx 0$ (naive).



Takeaway

A sophisticated algorithm can **steer** a naive rival to its own advantage — extracting rents rather than sharing them.

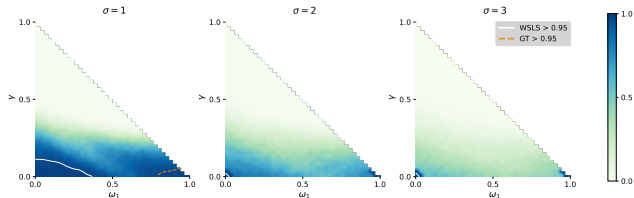
Implications

- **The mechanism:** whether agents model rivals determines not just *if* collusion forms, but whether it *survives* exploration.
- Our findings reconcile two seemingly contradicting literature:
 - *Abada & Lambin (2023, MS)* — no opponent beliefs; exploration will reverse collusion; matches our fragile naive-vs-naive result.
 - *Calvano et al. (2020)* — rivals enter only implicitly via state; conjecture that sophistication raises collusion; matches our robust sophisticated-vs-sophisticated result
- In addition, both literature study *homogeneous* populations. We also study **mixed sophistication** and show sophisticated algorithms can steer naive ones.

Numerical Experiment I: Bertrand Oligopoly Competition

The Impact of Demand Noise

- Vary demand-noise level σ in the payoff process
 - As σ rises (left $\rightarrow$ right panel), the **collusive basin shrinks substantially**.
 - $\Rightarrow$ Collusion is **harder to form and sustain in volatile markets**.



Contrast with Green–Porter (1983, 1984, Econometrica)

- GP assumes that firms are *endowed* with trigger strategies, so noise triggers price wars under imperfect monitoring. Noise facilitates coordination.
- Here, firms must **learn** reward–punishment patterns — noise instead **degrades payoff informativeness**, hindering coordination.

Numerical Experiment I: Bertrand Oligopoly Competition

More Players: Competition Erodes Collusion

- Switch to more than 2 agents.
- Per-firm extra-profit index Δ as the number of firms grows from 2 to 5; each firm initialized to play C with probability θ .

	$\theta = 0.6$	$\theta = 0.7$	$\theta = 0.8$	$\theta = 0.9$
2 players	40.67%	96.95%	99.88%	100%
3 players	1.15%	3.24%	12.25%	50.02%
4 players	1.02%	1.68%	2.89%	5.28%
5 players	0.95%	1.49%	2.08%	3.94%

- **More firms $\Rightarrow$ lower Δ :** extra profits collapse as competitive pressure intensifies.
- **Higher $\theta \Rightarrow$ higher Δ :** a stronger initial propensity to cooperate aids convergence to collusion (echoes Initialization part).

Numerical Experiment II: Algorithmic Market-Making

- Market making is now overwhelmingly automated, raising collusion concerns (Xiong and Cont (2021, MF), Dou et al. (2023), Colliard et al. (2026, RFS))
- Most studies use *standard Q-learning*, whose convergence is not guaranteed in multi-agent settings. Observed supra-competitive prices may be an **artifact of unstable learning**, not genuine strategy.
- We apply the two-timescale algorithm to a market-making game, adding **memory and state dependence** to test whether collusion is real.

Numerical Experiment II: Algorithmic Market-Making

The Market-Making Model (after Colliard et al. (2026, RFS))

- One risky asset, binary payoff $v \in \{v_L, v_H\}$ with $\mu_H = \mathbb{P}(v = v_H)$; revealed at end of each episode.
- Each episode a new investor arrives with private valuation $v_t^B = v_t + L_t$, $L_t \sim N(0, \sigma^2)$ (liquidity shock).
- Two market makers post ask prices (a_{1t}, a_{2t}) ; investor buys at the **lowest ask** $\leq v_t^B$.
- AMM i earns in each episode

$$r_t^i(a_{it}, a_{-it}, v_t^B) = \frac{1}{|D(a_{1t}, a_{2t})|} \cdot \mathbb{I}_{\{a_{it} = a_t^{\min} \leq v_t^B\}} (a_t^{\min} - v_t)$$

where $a_t^{\min} = \min\{a_{1t}, a_{2t}\}$ and

$$D(a_{1t}, a_{2t}) = \{i \in \{1, 2\} : a_{it} = a_t^{\min}\}.$$

- Lower asks attract informed buyers $\Rightarrow$ **adverse selection**, severity measured by $\Delta v = v_H - v_L$.

Numerical Experiment II: Algorithmic Market-Making

The Market-Making Model (after Colliard et al. (2026, RFS))

Key departure from Colliard et al.

- In our paper, state = rival's *previous* quotes: $s_{t+1} = (a_{1t}, a_{2t})$. Quotes can now condition on the rival's past behavior.
- By contrast, the agents in Colliard (2026) are **stateless bandits player** reacting only to their own past choices.

Numerical Experiment II: Algorithmic Market-Making

Benchmark and Spread Measures

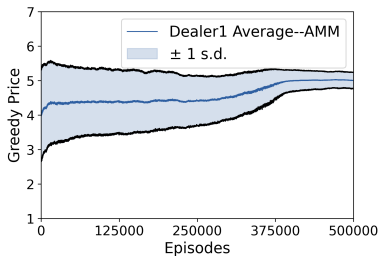
Following Glosten-Milgrom (1985, JFE), we compute

- **Quoted Spread (QS):** unconditional markup of best ask over expected value, $QS_t = \mathbb{E}[a_t^{\min} - v_t]$. The cost of immediacy; positive to offset adverse selection.
- **Realized Spread (RS):** markup *conditional on a trade*, $RS_t = \mathbb{E}[a_t^{\min} - v_t \mid a_t^{\min} \leq v_t^B]$. The rent actually earned — driven to **zero** under Bertrand–Nash. Call such price the Glostein-Milgrom price.
- $RS < QS$: the gap measures the adverse-selection cost. A **strictly positive RS** is the smoking gun for collusion.

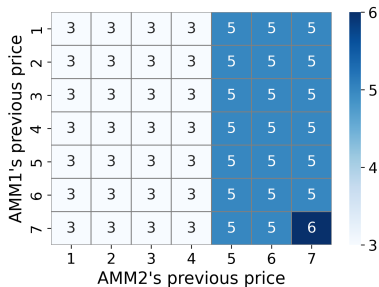
Numerical Experiment II: Algorithmic Market-Making

State-Contingent Collusion

- Baseline $(\sigma, \Delta v) = (5, 4)$; Glosten–Milgrom price ≈ 3 –4.
- Learned price converges **supra-competitively**;
QS = 2.58, RS = 1.72 > 0.
- Panel (b): clear **punishment rule** — MM1 quotes the collusive \$5, but switches to the competitive \$3 once it detects MM2 undercutting.



(a) Expected price path



(b) Quote with max prob.

Numerical Experiment II: Algorithmic Market-Making

State-Contingent Collusion (Continued)

vs. Colliard et al. (2026, RFS)

- Their bandits get *stuck* above competition through **incomplete learning**.
- With states, our agents learn to **punish** — this is collusion, not a learning artifact.

Numerical Experiment II: Algorithmic Market-Making

Collusion in the Presence of Adverse Selection

- Fix Δv and change σ . Compute QS, RS, and QS-RS.
- Observations:
 - QS-RS decreases in a market with more liquidity traders because the adverse selection costs declines
 - RS increases in a market with more liquidity traders because AMM can easily earn higher rents.

Metric	$\sigma = 0.5$	1.0	3.0	5.0	7.0
QS	2.00 (0.00)	2.00 (0.00)	2.46 (0.50)	2.58 (0.49)	2.85 (0.57)
RS	0.00 (0.00)	0.00 (0.00)	1.00 (0.42)	1.72 (0.45)	2.26 (0.54)
QS-RS	2.00 (0.00)	2.00 (0.00)	1.46 (0.08)	0.86 (0.05)	0.59 (0.03)

Conclusions

- We develop a population game based approach to solve multi-agent reinforcement learning problems.
- The new approach relaxes some restrictive assumptions in the literature and can compute Nash equilibria very efficiently for complex cases in which there exist only mixed-policy Nash equilibria and more than 2 agents.
- Our algorithm has the potential to help understanding of agent collusions

Belief of LLM

- We tried to use LLM as experimental subjects in a bubble game.
- Chain-of-thought analysis reveals that LLM do hold belief on we human beings as opponents in the game. It reads dangerously

****Evaluating human versus machine buying behavior****

I'm considering that the normative answer under expected utility suggests buying could be beneficial, as there's a huge potential profit if I'm not last. However, given human behavior in games, especially when confusion is possible, they might act naively. While the optimal response against humans might be to buy, thinking they could act irrationally, against machines doesn't present a similar opportunity. Thus, I conclude if opponents are humans, the answer is likely YES; if they're machines, it's a solid NO.

(ChatGPT5.2; high reasoning mode)

- Welcome to our talk:
 - Cartea, Chang, Chen, and Zhong, "AI bubbles with Large Language Models", Presented in BFS Session MS_255, Aula F, July 1 10:30-12:30.

Appendix I: Stochastic Approximation with Two Timescales

Discrete-Time Dynamics

- For ease of exposition, consider the case in which all the agents use the same learning rates.
- The updating of X , $\hat{X}$ and Q are given by

$$\begin{cases} X_{l+1}^k = X_l^k + \frac{1}{N_l} \left(\mathbf{e}_{s_l, b_l^k} - \mathbf{e}_{s_l, a_l^k} \right), \\ \hat{X}_{l+1}^k = \hat{X}_l^k + \frac{1}{N_l} \left(\mathbf{e}_{s_l, a_l^k} - \text{diag}\{\mathbb{I}(s)\}_{(s, a^k) \in \mathbb{S} \times \mathbb{A}^k} \cdot \hat{X}_l^k \right), \\ Q_{l+1}^k = Q_l^k + \alpha_{\nu_l(s_l, b_l)} \left(r_l^k + \beta \text{Popu}_l^k(s') - Q_l^k \right). \end{cases} \quad (2)$$

Theoretical Framework

- All of them can be rewritten in the following form of coupled recursions with controlled Markov noise:

$$\begin{aligned}\theta_{n+1} &= \theta_n + a_n \left[h(\theta_n, \omega_n, Z_n^{(1)}) + M_{n+1}^{(1)} \right], \\ \omega_{n+1} &= \omega_n + b_{v_n} \left[g(\theta_n, \omega_n, Z_n^{(2)}) + M_{n+1}^{(2)} \right],\end{aligned}\tag{3}$$

- Two time scales
- Controlled Markov state
- We characterize the asymptotic behavior of $(X_l, \widehat{X}_l, Q_l)$, using
 - Fast timescale: multi-timescale stochastic approximation with controlled Markov noise (Karmakar and Bhatnagar (2018, MOR))
 - Slow timescale: quasi-Fejér analysis + martingale convergence theory (Duvocelle et al. (2023, MOR), Mertikopolous et al. (2024, Math Program))

Fast Timescale

- In the **fast** timescale α , we first re-write the dynamic (2) as follows:

$$\begin{cases} X_{l+1}^k = X_l^k + \alpha_{\nu_l} \frac{1/N_l}{\alpha_{\nu_l}} \left(\mathbf{e}_{s_l, b_l^k} - \mathbf{e}_{s_l, a_l^k} \right), \\ \hat{X}_{l+1}^k = \hat{X}_l^k + \alpha_{\nu_l} \frac{1/N_l}{\alpha_{\nu_l}} \left(\mathbf{e}_{s_l, a_l^k} - \text{diag}\{\mathbb{I}(s)\}_{(s, a^k) \in \mathbb{S} \times \mathbb{A}^k} \cdot \hat{X}_l^k \right), \\ Q_{l+1}^k = Q_l^k + \alpha_{\nu_l} \left(r_l^k + \beta \text{Popu}_l^k(s') - Q_l^k \right). \end{cases} \quad (4)$$

Through the two-timescale mechanism in conjunction with (4), we show that X_l , $\hat{X}_l$ and Q_l will converge to the solution to the following ODE:

$$\begin{aligned} \dot{\mathcal{X}}_t &= 0; \\ \dot{\hat{\mathcal{X}}}_t &= 0; \\ \dot{Q}_t &= G(\mathcal{X}_t, \hat{\mathcal{X}}_t, Q_t). \end{aligned}$$

In other words, both X_l and $\hat{X}_l$ will converge and that Q_l will be very close to $\lambda(X_l, \hat{X}_l)$ as l increases.

Fast Timescale (Continued)

- Here, $Q = \lambda(X, \hat{X})$ satisfies

$$G(X, \hat{X}, Q) = 0$$

where we define $G(X, \hat{X}, Q) = (G^1(X, \hat{X}, Q), \dots, G^n(X, \hat{X}, Q))$ with $G^k(X, \hat{X}, Q)$ is a vector indexed by $(s, \mathbf{a}) \in \mathbb{S} \times \prod_{k=1}^n \mathbb{A}^k$ and

$$G_{(s, \mathbf{a})}^k(X, \hat{X}, Q) = \pi(s) \left(\prod_{k=1}^n x^k(s, a^k) \right) g_{(s, \mathbf{a})}^k(X, \hat{X}, Q),$$

with $\pi(s)$ representing the stationary distribution of states in a Nash equilibrium and

$$g_{(s, \mathbf{a})}^k(X, \hat{X}, Q) = \mathbb{E}[r^k(s, \mathbf{a})] + \beta \sum_{s' \in \mathbb{S}} \mathbb{P}(s'|s, \mathbf{a}) \text{Popu}Q^k(s') - Q^k(s, \mathbf{a}).$$

Slow Timescale

- In the **slow** timescale $\frac{1}{N_l}$,
 - the beliefs $\hat{X}_l$ will be consistent with the true policy X_l ; i.e., $\|\hat{X}_l - X_l\| \rightarrow 0$ as $l \rightarrow +\infty$.
 - the evolution of policy vector X_l will converge to the stationary point to the dynamical system

$$\dot{X} = f(X, X) \quad (5)$$

where $f(X, \hat{X}) = (f^1(X, \hat{X}), \dots, f^n(X, \hat{X}))$ is a vector-valued function such that each $f^k(X, \hat{X})$ is a vector indexed by $(s, a) \in \mathbb{S} \times \mathbb{A}^k$:

$$f(X, \hat{X})_{s,a}^k = \pi(s) \left[\rho^k(a; s, \hat{X}^{-k}, \lambda(X, \hat{X})) - X^k(s, a) \right].$$

with

$$\rho^k(b; s, \hat{X}^{-k}, Q^k) = \frac{\exp \left(F^k(b; s, \hat{X}^{-k}, Q^k) / \eta \right)}{\sum_{b' \in \mathbb{A}_k} \exp \left(F^k(b'; s, \hat{X}^{-k}, Q^k) / \eta \right)}.$$

if the stationary point to (5) is stable in the sense...

Assumptions

Assumption (Finiteness of reward, action space, and state space)

The state space $\mathbb{S}$ and action space $\mathbb{A}$ are finite. There exists a sufficiently large $B > 0$ such that $|r^k(s, a)| < B$ for all s and a .

Assumption (Irreducible state space)

Given any pair of states (s, s') , there exists at least one sequence of actions such that s' is reachable from s with some positive probability within a finite number, n , of stages.

Assumptions (Continued)

Assumption (Two Timescales)

1 a.

$$\sum_{l=0}^{\infty} \alpha_l = \infty, \quad \sum_{l=0}^{\infty} \alpha_l^2 < \infty, \quad \sum_{l=0}^{\infty} \frac{1}{N_l} = \infty, \quad \sum_{l=0}^{\infty} \frac{1}{N_l^2} < \infty,$$

b.

$$\frac{1}{N_l} = o(\alpha_l)$$

c. Letting c_l stand for either α_l or $1/N_l$. For any $\gamma \in (0, 1)$,

$$\sup_{l \geq 0} \frac{c_{[\gamma l]}}{c_l} < \infty,$$

where $[\gamma m]$ is the integral part of γm .

Assumptions (Continued)

Assumption (local stability of $\dot{X} = f(X, X)$)

Let X^ be the stationary point of the system (5). Assume that all the real parts of the eigenvalues of the Jacobian of f at X^* are negative.*

Convergence Analysis

Theorem (Chen et. al. (2024))

Under the above assumptions, given $\epsilon > 0$ and $p > 0$, we can find sufficiently small η and sufficiently large c and a neighborhood $\mathcal{W}$ of X^ such that, if $X_0^\eta \in \mathcal{W}$, we have*

$$\mathbb{P} \left(\lim_{l \rightarrow +\infty} (X_l^\eta, Q_l^\eta) = (X_*, Q_*) \right) > 1 - p,$$

where X_ is the ϵ -Nash equilibrium of the original problem and Q^* is the corresponding Nash Q-value.*

► Back

Appendix II: Numerical Example Setup

Benchmark Price Details

Monopoly price (Li & Huh 2011): $p_M^i = \mu [1 + W(\sum_{i=1}^n A_i)] + c_i$, with $A_i = \exp\{(a_i - a_0 - c_i)/\mu - 1\}$ and W the Lambert- W function.

Bertrand-Nash price: $p_N^i = \frac{\mu}{1 - V(A_i B_0)} + c_i$, where $V(x) \in (0, 1)$ solves $v e^{v/(1-v)} = x$ and B_0 solves $B_0 + \sum_{i=1}^n V(A_i B_0) = 1$. Under the same parameters, $p_M \geq p_N$.

◀ Back to experiment