Collusive Outcomes in a Sequential Pricing Duopoly with Q-Learning Agents[1]

Master's Initiation to Research Project

Côme Campagnolo[2]

May 23rd, 2024

[1] Supervision by Thierry Foucault, HEC Foundation Chaired Professor of Finance

[2] Master of Economics (M1), ENS Paris-Saclay, France

Introduction

Issue at stake

Automated pricing algorithms relying on reinforcement learning methods seem able to reach collusive outcomes in oligopoly markets.

Definition 1. Algorithmic Collusion

Coordinated behavior among **independent** (without direct communication or agreement), self-interested agents to achieve outcomes similar to those of explicit collusion, through the use of algorithms and automated decision-making systems.

"Algorithmic Collusion"

A Recent Phenomenon

- Algorithmic pricing can be categorized into adaptive and AI-based algorithms.
- AI algorithms, such as those using Q-learning, autonomously learn anti-competitive behaviors.
- Legal interpretation of algorithmic collusion challenges antitrust policies.

Empirical Evidence

- Limited empirical studies on algorithmic collusion.
- Initial evidence suggests significant effects of AI pricing on market outcomes (Assad et al, 2023).
- Difficulty to isolate collusion effects from other pricing effects linked to the introduction of AI-pricing agents.

Research Question

Model

Models based on **Q-learning** aim to understand the emergence and sustainability of such strategies. We simulate a sequential pricing duopoly inspired by Maskin and Tirole (1988)'s economic framework and its adaptation to Q-learning by Klein (2021).

Price Grid Size Variation

The size of the price grid is an important factor of the determination of the best ask and bid prices on financial markets (Cordella & Foucault, 1999). Is it still the case with Q-learning agents?

Research Question

To what extent does the discretization of prices influence Q-learning agent's collusive behaviors?

Q-Learning Mechanism

- The algorithmic agent interacts with an environment by taking actions and receiving **rewards**.
- The agent learns to **maximize its cumulative reward over time by updating its action-value function** $Q(s, a)$ ('s' represents the current state and 'a' represents the action taken in that state).
- The action-value function estimates the expected cumulative reward of taking action 'a' in state 's'.
- The algorithm **iteratively updates the action-value function**[3] based on the observed rewards and transitions between states after interacting with the environment.

[3] see Appendix 1

Bellman Equation and Updating Rule

Definition 2. Value Function

$$V(s) = \mathbb{E}_{\pi}\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s\right]$$

The Bellman equation allows to substitute the calculation of the policy π_s into the value function by backward induction.

Definition 3. Bellman Equation

$$V_i(p_{jt}) = \max_{p}\left[\pi_i(p, p_{jt}) + \mathbb{E}_{p_{j,t+1}}\left[\delta\pi_i(p, p_{j,t+1}) + \delta^2 V_i(p_{j,t+1})\right]\right]$$

- α is the learning rate
- r is the immediate reward
- γ is the discount factor that applies to future rewards
- s' is the next state

Bellman Equation and Updating Rule

Definition 4. Updating Rule of the Q-Learning Algorithm

$$Q_i(p_{i,t}, p_{j,t}) = (1-\alpha) \cdot Q_i(p_{i,t}, p_{j,t}) +$$

$$\alpha \cdot [\pi(p_{i,t}, p_{j,t}) + \gamma \cdot \pi(p_{i,t}, p_{j,t+1}) + \gamma^2 \cdot \max_p Q_i(p, p_{j,t+1})]$$

- $Q(p_{it}, p_{jt})$: discounted previous estimate of the Q-value for the considered action-state pair
- $\max_p Q_i(p, p_{j,t+1})$: maximal Q-value for the next state

Q-Learning is a simple reinforcement learning method which aims to determine the policy that maximizes the value function.

Q-Learning Features

- designed to solve Markov decision processes (MDPs) with discrete states and actions (Watkins, 1989)
- simple => a straightforward economic interpretation
- model-free => does not require knowledge of its environment
- **exploration-exploitation trade-off / ϵ-greedy rule**. The agent has to choose at each episode between exploring new strategies to discover potentially better options and exploiting known strategies to maximize immediate rewards.

$$p_{it} \begin{cases} \sim U\{P\} & \text{with probability} \varepsilon_t \\ = \text{argmax}_p \, Q_i(p, s_t) & \text{with probability} 1 - \varepsilon_t \end{cases}$$

Simulation Setup

1. **Sequential Pricing Duopoly**: Each firm updates its price sequentially[4]: this reflects market conditions where firms react to each other's pricing decisions over time.

Demand Function

$$D_i(p_{i,t}, p_{j,t}) = \begin{cases} 1 - p_{i,t} & \text{if} p_{i,t} < p_{j,t} \\ 0.5(1 - p_{i,t}) & \text{if} p_{i,t} = p_{j,t} \\ 0 & \text{if} p_{i,t} > p_{j,t} \end{cases}$$

Profits

$$\pi_i(p_{it}, p_{jt}) = p_{it} \cdot D_i(p_{it}, p_{jt})$$

[4] not simultaneously as in Calvano et al. (2020)

Simulation Setup

2. **Parametrization**

Q-Learning Parameters

- **Learning Rate** (α): 0.3, controlling how quickly the algorithm updates its knowledge.
- **Discount Factor** (γ): 0.95, emphasizing the importance of future rewards.
- **Exploration Rate** (ϵ): Starts at 1 and decays to 0.0001 over 100,000 episodes, progressively decreasing the rate of exploration of new strategies.

Simulation Setup

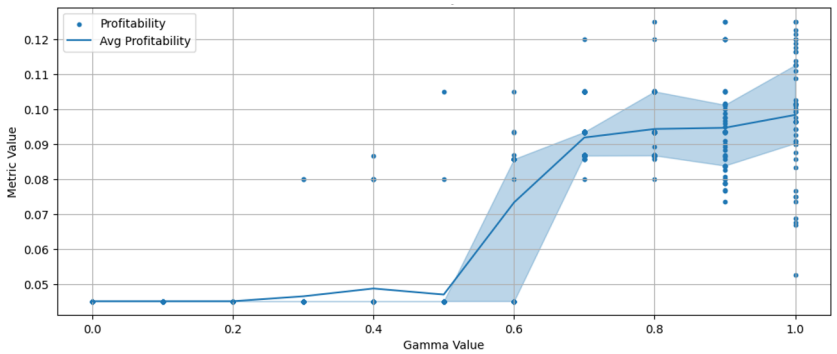
3. **Price Grid Size Variation**(from 2 to 1,000): Does the price grid size variation impact agents' behavior and market outcomes?
4. **Performance Metrics**:
a) **Profitability**

$$\Pi_i = \frac{1}{2} \cdot \frac{\sum_{t=T-1,000}^{T} \pi_i(p_{i_t}, p_{jt})}{1,000}$$

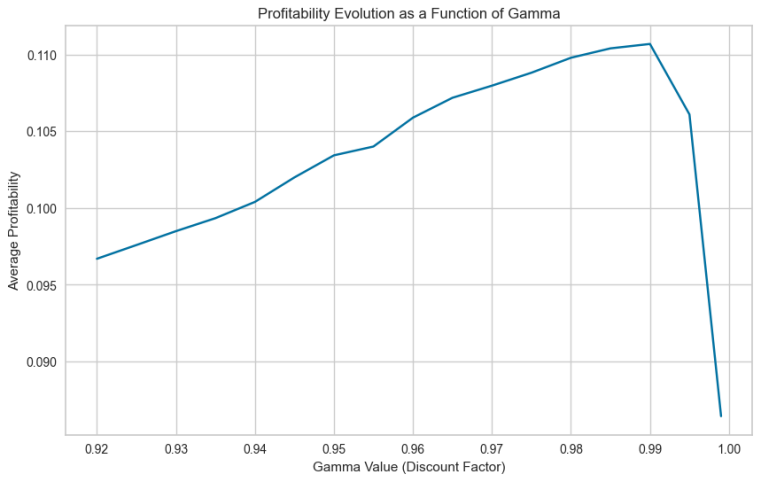
b) **Comparison Benchmarks**: Monopoly benchmark (maximal joint-profit) and dynamic competitive benchmark (Edgeworth price cycles[5]).

[5] cf. Maskin & Tirole

Results (Parametrization)



(a) Discount Factor (0-0.99)



(b) Discount Factor (0.9-0.999)

Figure: Profitability Depending on Discount Factor

=> Choosing the appropriate parameters has a decisive impact on agent's performance.

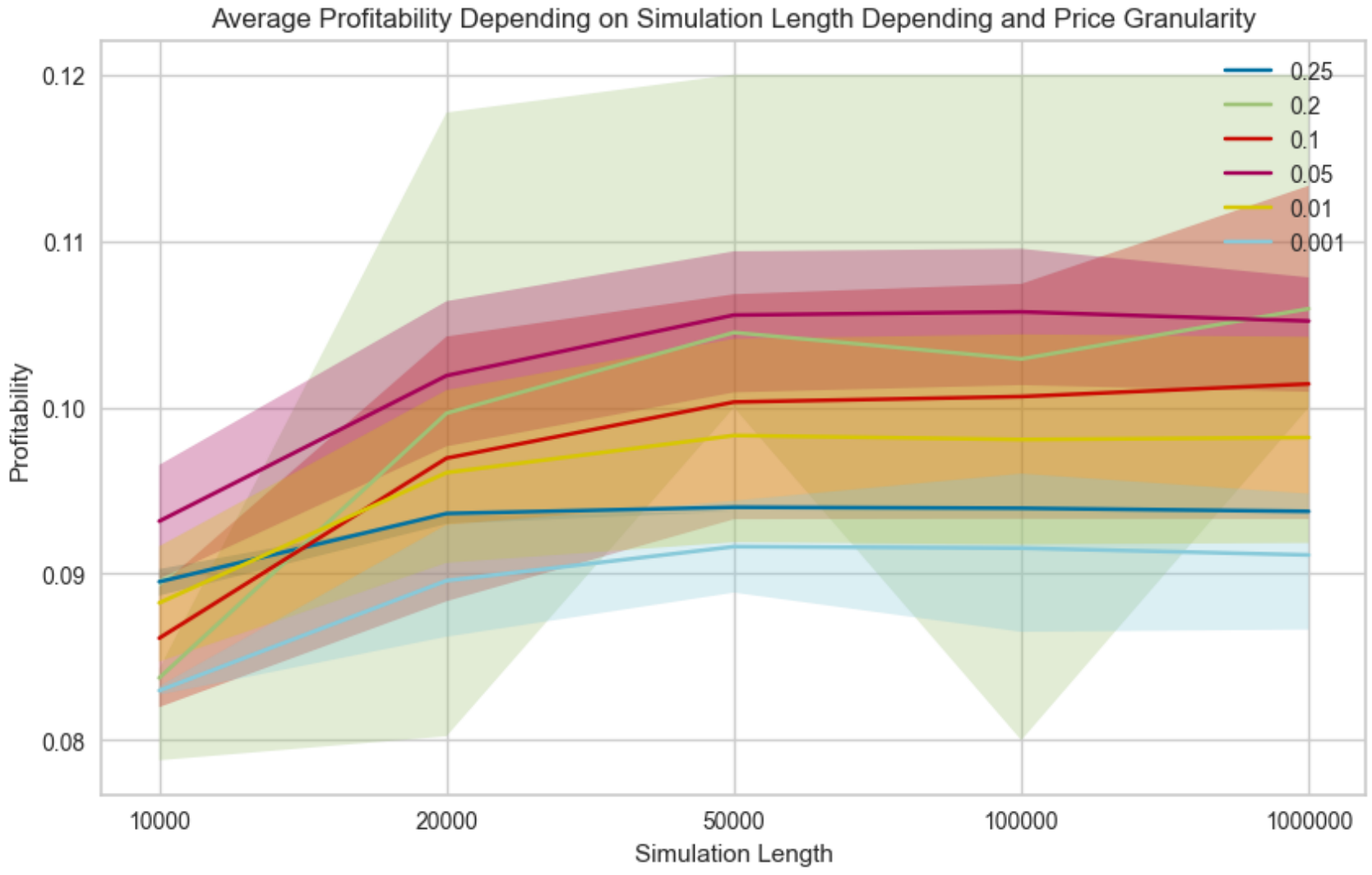


Figure: Mean Profitability and Quartiles Depending on Price Discretization

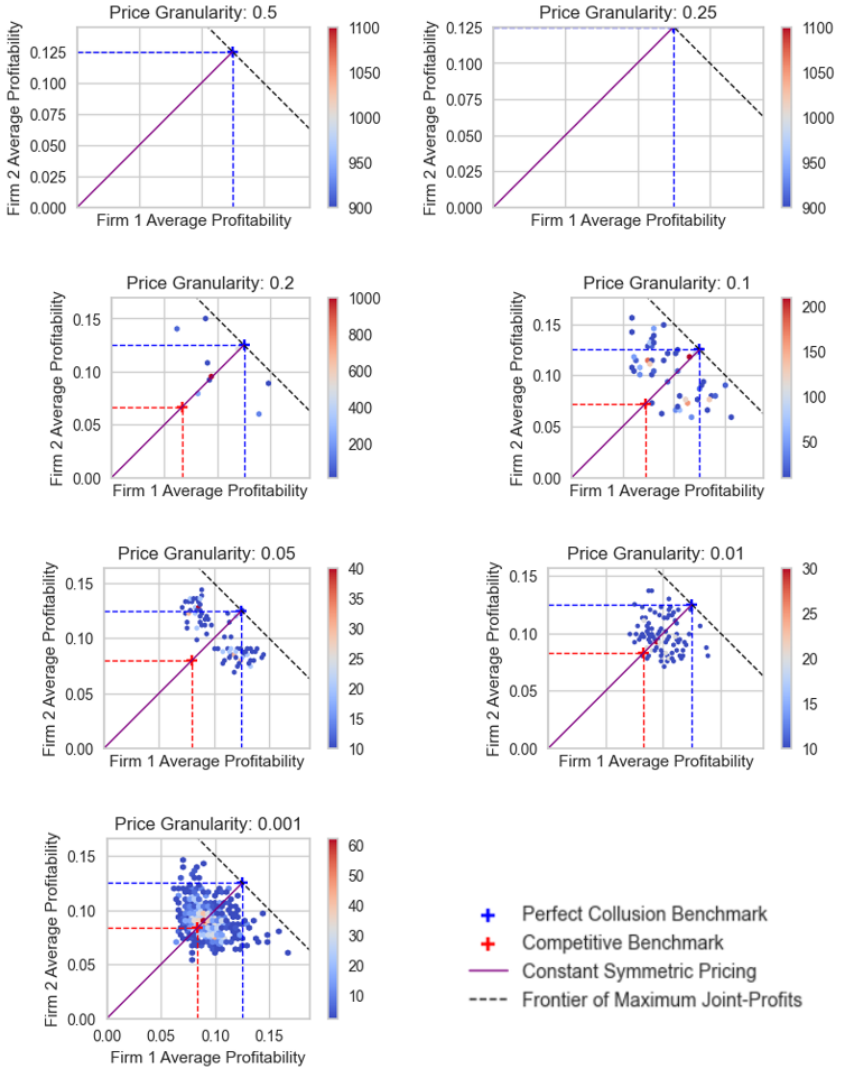


Figure: Joint-Profitability Distribution After 1 Million Episodes Depending on the Price Discretization[6]

[6] See Appendix for detailed results

Results (Pricing Strategies)

Price Granularity*		**0.25**	**0.2**	**0.125**	**0.1**	**0.05**	**0.01**	**0.001**
Average Price Cycle Length**		**1.00**	**1.32**	**2.44**	**2.73**	**3.50**	**7.16**	**12.91**
Most Frequent Price Cycle Length		1	1	3	3	4	5 to 10	10 to 20
% Constant Symmetric Price Pattern*		**100.0**	**68.4**	**27.8**	**19.0**	**6.2**	**0.4*****	**0.0**
Average Profitability		0.125	0.09375	0.11482	0.10757	0.1214		
% Price pairs	Profit							
(0.5, 0.5)	0.125	0.6		5.0	4.4	0.6		
(0.45, 0.45) or (0.55, 0.55)	0.12375					1.4		
(0.4, 0.4) or (0.6, 0.6)	0.12		41.2		10.8	1.8		
(0.375, 0.375) or (0.625, 0.625)	0.11719			14.8				
(0.35, 0.35) or (0.65, 0.65)	0.11375					1.2		
(0.3, 0.3) or (0.7, 0.7)	0.105				3.6	1.0		
(0.25, 0.25) or (0.75, 0.75)	0.09375	99.4		8.0		0.2		
(0.2, 0.2) or (0.8, 0.8)	0.08		27.2		0.2	0.0		
% Price Cycle Pattern*		**0.0**	**31.6**	**72.2**	**81.0**	**93.8**	**99.6**	**100.0**
Average profitability gap between players (after 100,000 episodes) (after 1 million episodes)			0.07745 0.07077	0.04424 0.04335	0.044282 0.044285	0.038 0.0385	0.01644 0.0178	0.0181 0.01589
Average Profitability			0.10027	0.0941	0.09587	0.105	0.09805	0.09154

* Results after 100,000 episodes with 500 simulations for each price interval

** Results after 1,000,000 episodes with 100 simulations for each price interval

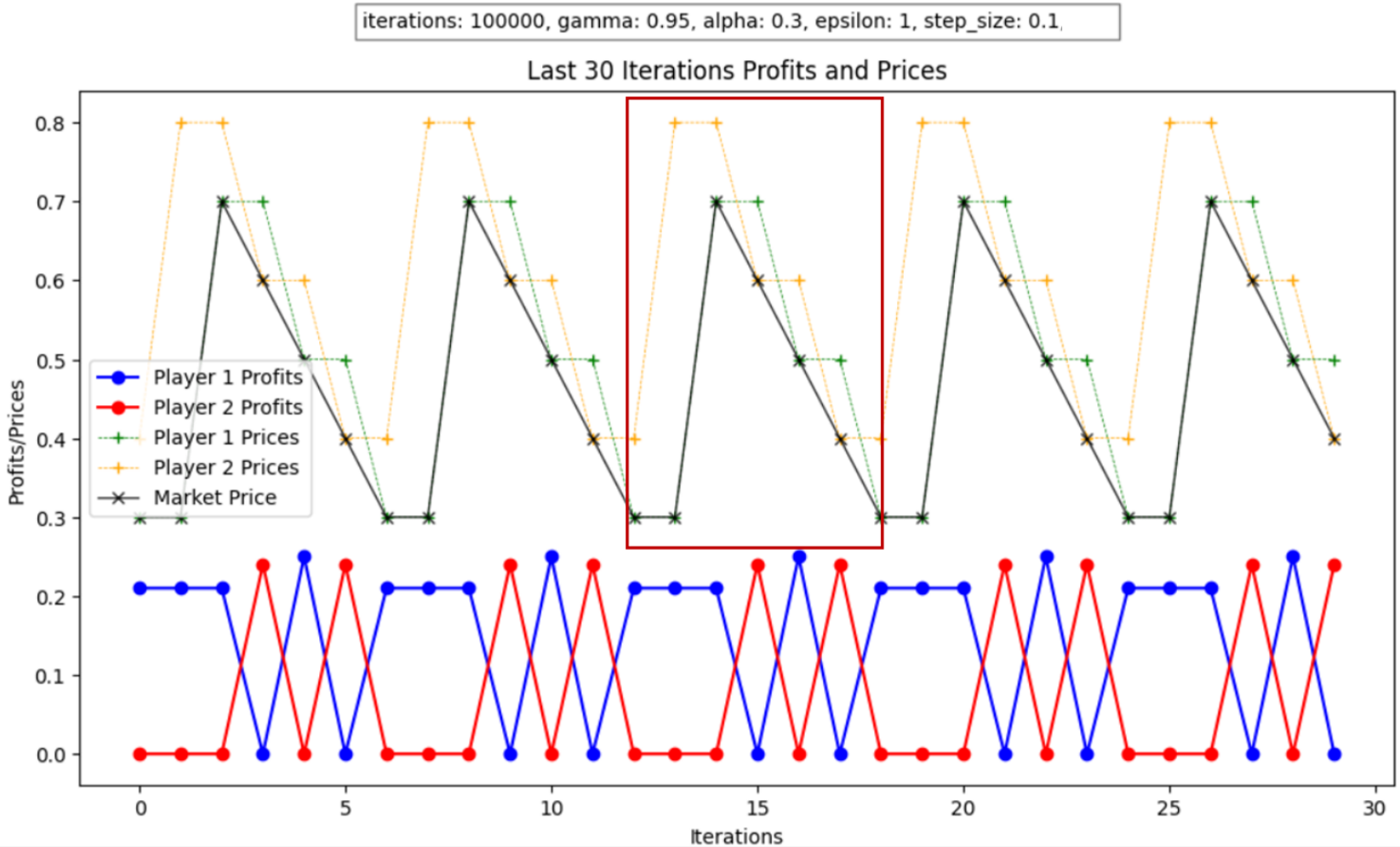


Figure: Example of an Undercutting Price Cycle Pattern

Results (Undercutting Price Cycles)

Definition Edgeworth Price Cycle Equilibrium

Mixed strategy defined by the following dynamic reaction functions:

$$\forall p \in \mathscr{P} \setminus (p_{min}), \forall i,j \in (1,2), R_i(p) = p_j - k$$

with k the price interval and if $p = p_{min}$,

$$R_i(p) = \begin{cases} p_{min} & \text{with a given probability x} \\ p_{max} & \text{with probability 1-x} \end{cases}$$

It is the most competitive Markov Perfect Equilibrium [7].

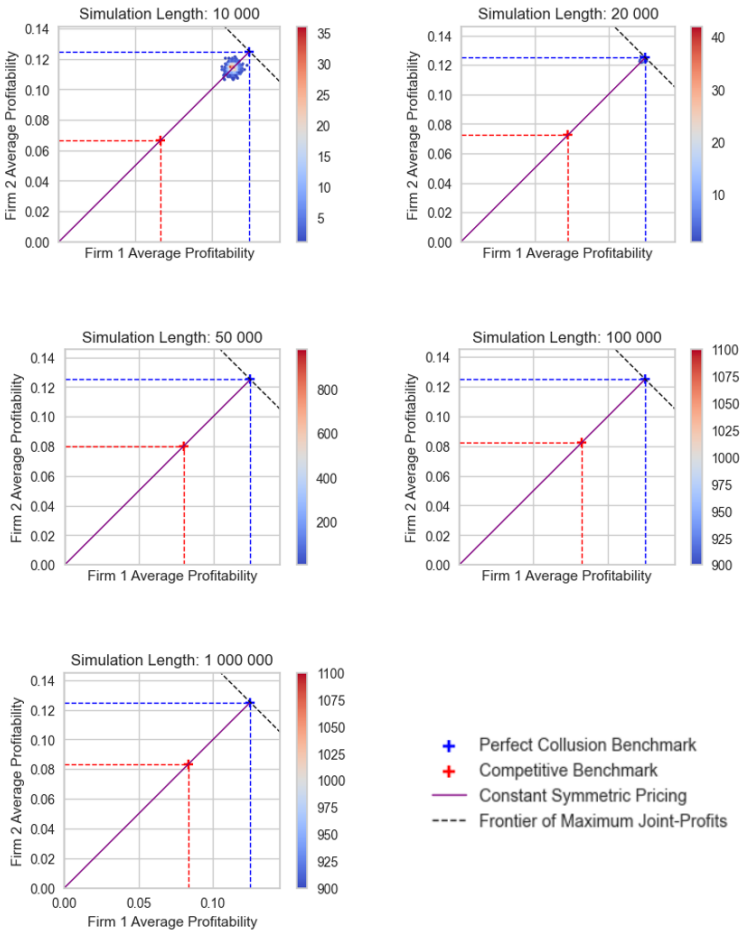
[7] subgame perfect Nash Equilibrium with regard to the Markov property: $\mathbb{P}[s_{t+1} \mid s_t] = \mathbb{P}[s_{t+1} \mid (s_i)_{i \in [1,t]}]$

Conclusion

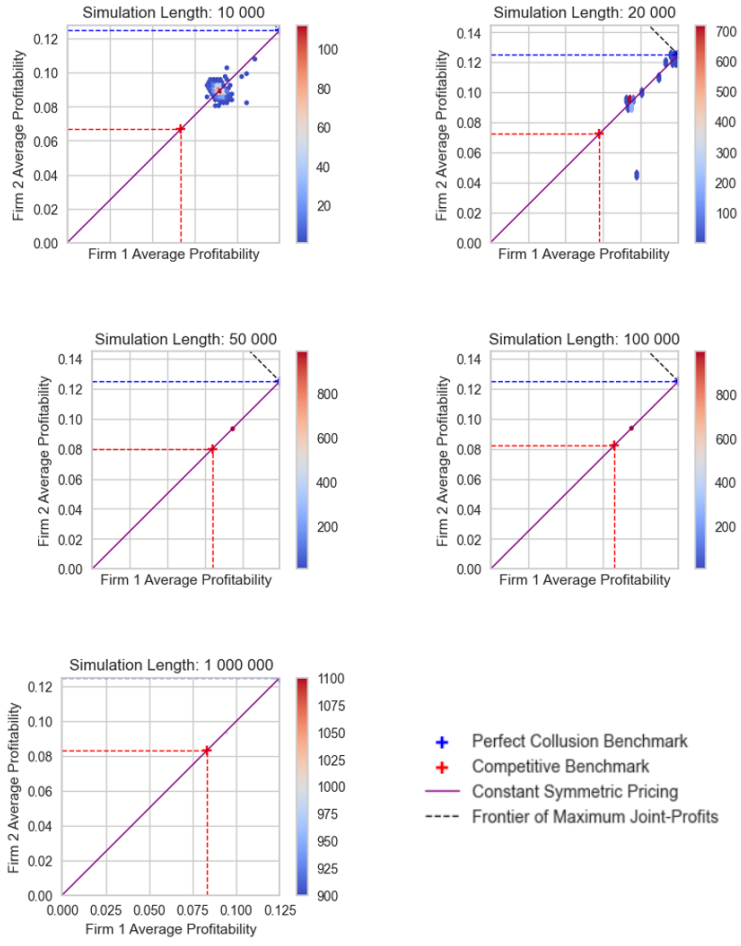
- Replication of Klein (2021)'s paper: Introducing Q-learning agents in a sequential pricing duopoly model validates their capability to achieve collusive outcomes across various parameters, particularly in relation to the price grid.
- Importance of algorithm design: The price grid size is a significant factor in market design, with a substantial impact on profitability levels within the model. A finer price grid doesn't necessarily lead to increased collusion; however, it induces higher price volatility, evidenced by more frequent price changes. There could be an optimal value for the price grid size.
- Despite greater price volatility, players' strategies remain robust to market variations. Q-learning agents demonstrate behaviors akin to traditional pricing strategies like constant pricing and undercutting ('price war'), albeit not identical.

Further Research Questions

1. Our model is restricted to sequential pricing where firms alternatively set prices. To be more realistic, we could introduce a **time delay mechanism** into the model. It would allow further exploration of the role of price discretization, akin to previous research in the financial literature (cf. Cordella & Foucault).
2. Using a simple algorithm allows for a straightforward interpretation but also limitates the complexity of agents behaviors: with more sophisticated algorithms (like DQN or Policy-Gradient Methods), we could investigate mixed strategies or continuous state-action spaces.

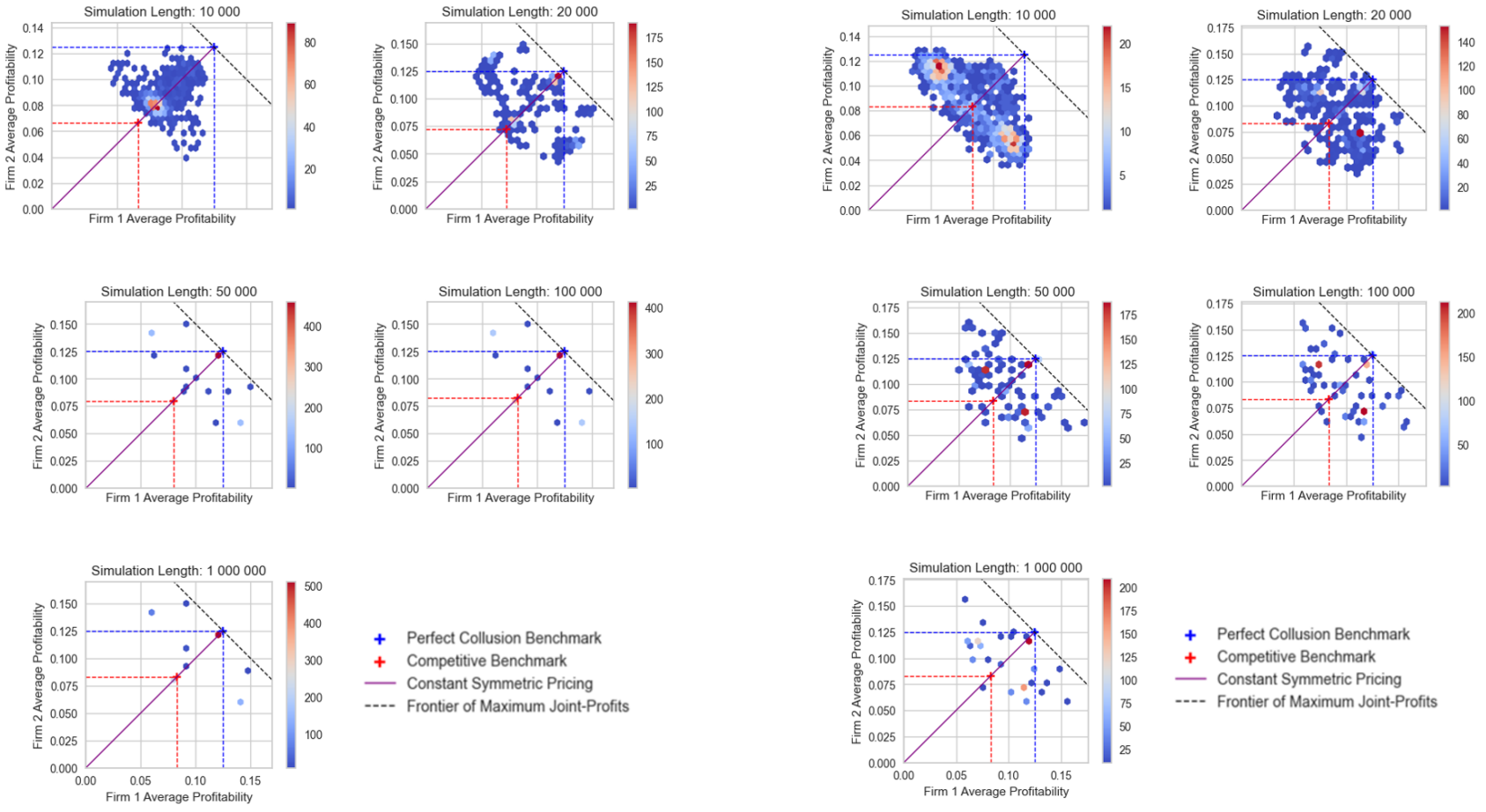


(a) Price Interval = 0.5



(b) Price Interval = 0.25

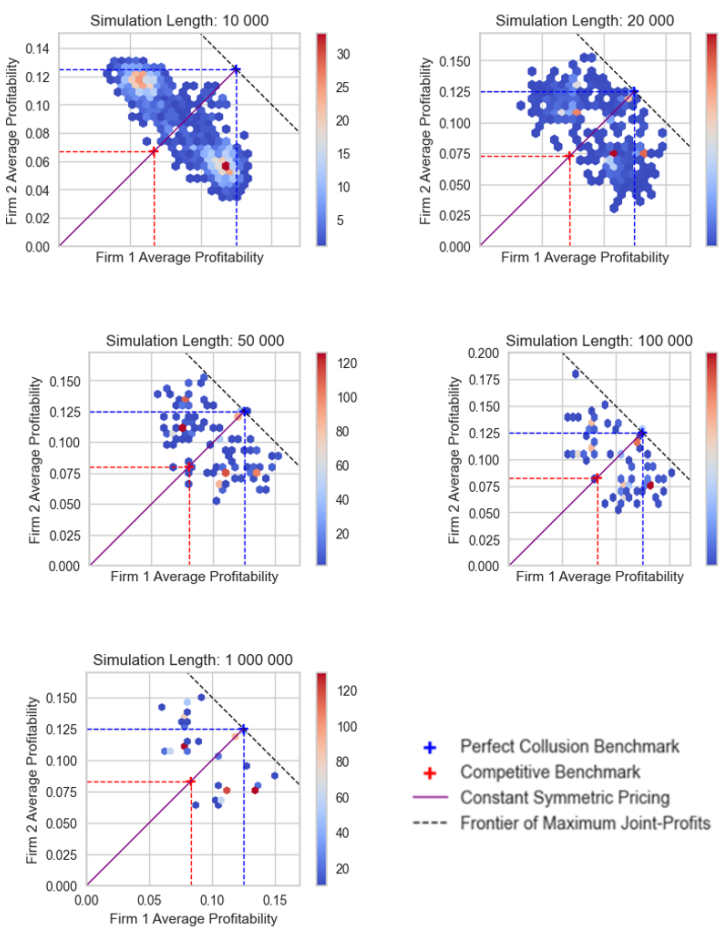
Figure: Profitability Distribution for Price Intervals 0.5 and 0.25



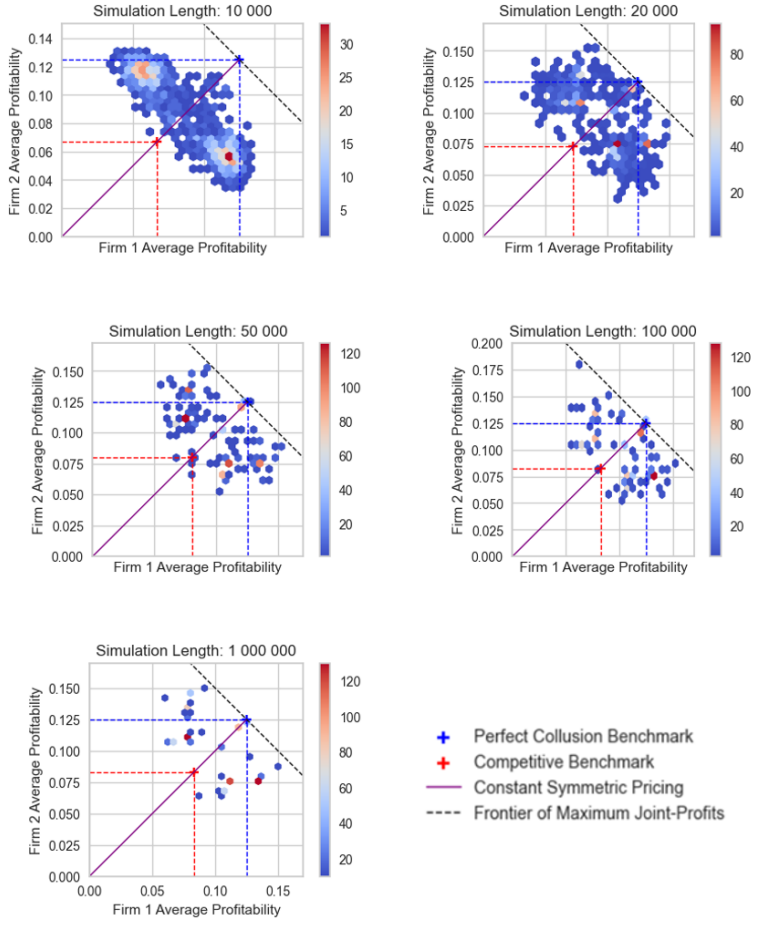
(a) Price Interval = 0.2

(b) Price Interval = 0.125

Figure: Profitability Distribution for Price Intervals 0.2 and 0.125

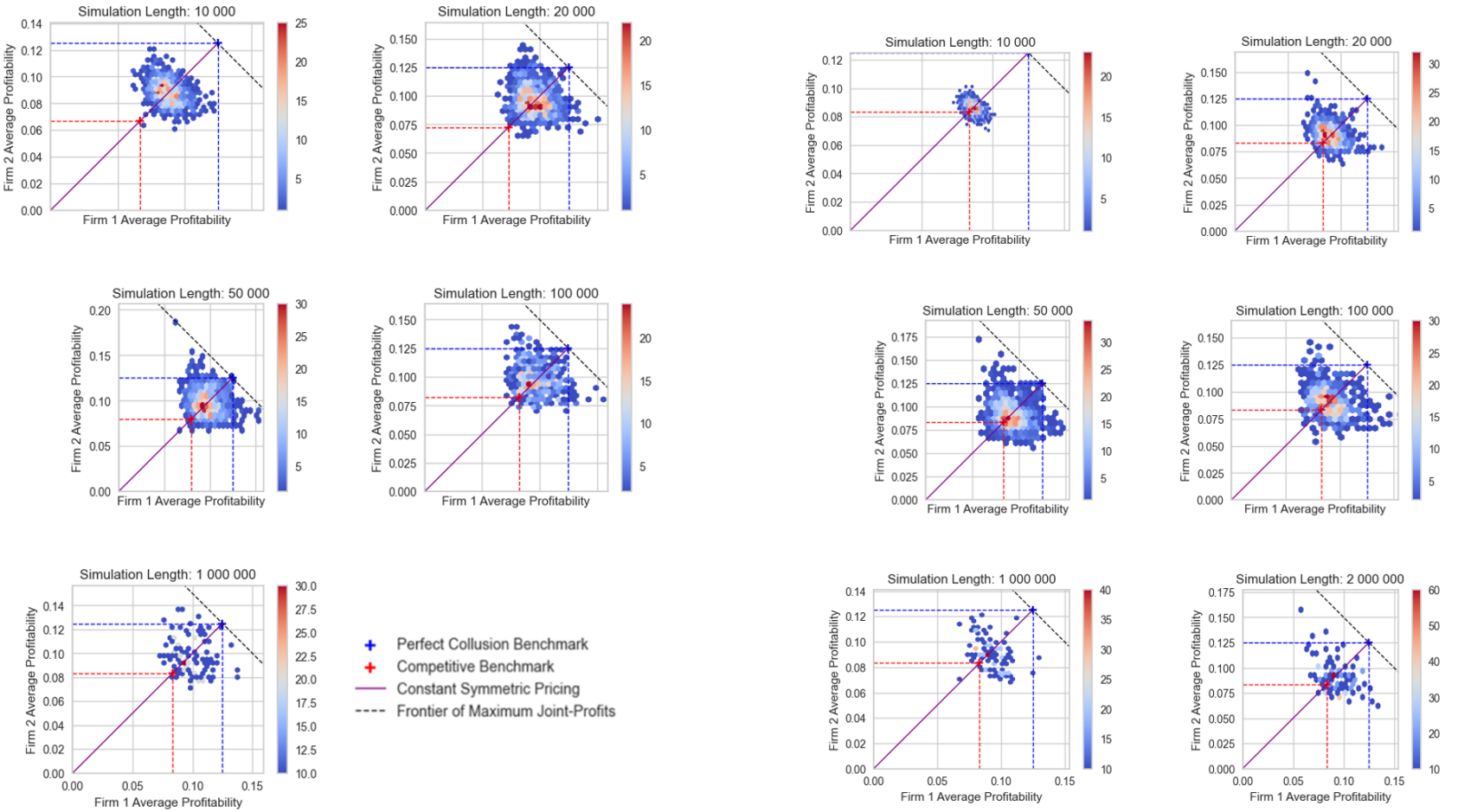


(a) Price Interval = 0.1



(b) Price Interval = 0.05

Figure: Profitability Distribution for Price Intervals 0.1 and 0.05



(a) Price Interval = 0.01

(b) Price Interval = 0.001

Figure: Profitability Distribution for Price Intervals 0.01 and 0.001

References

CALVANO, E., CALZOLARI, G., DENICOLÒ, V., PASTORELLO, S. (2020). Artificial Intelligence, Algorithmic Pricing, and Collusion. *American Economic Review*, 110 (10), 3267-97.

CORDELLA, T., FOUCAULT, T. (1999). Minimum Price Variations, Time Priority, and Quote Dynamics. *Journal of Financial Intermediation*, 8 (3), 141-173.

KLEIN, T. (2021). Autonomous algorithmic collusion: Q-learning under sequential pricing. *The RAND Journal of Economics*, 52 (3), 538-558.

MASKIN, E., TIROLE, J. (1988). A Theory of Dynamic Oligopoly, I: Overview and Quantity Competition with Large Fixed Costs. II: Price Competition, Kinked Demand Curves, and Edgeworth Cycles. *Econometrica*, 56 (3), 549–599.

WATKINS, C. (1989). Learning from Delayed Rewards (Ph.D. thesis). *University of Cambridge*