

Collusion and Artificial Intelligence

A computational experiment with sequential pricing algorithms under stochastic costs

Gonzalo Ballesterio

UdeSA

2021

From Mad Men to Maths Men

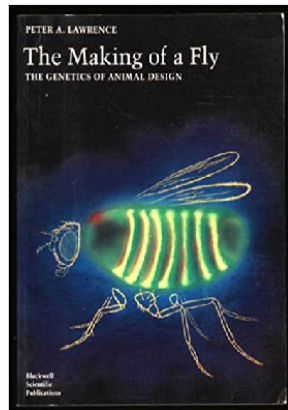
The new world relies on Math Men with hard skills in computer science, maths, and physics. These Math Men develop algorithms to break down Big Data into digestible insights



Something is happening in the online marketplace...

How A Book About Flies Came To Be Priced \$24 Million On Amazon

Two booksellers using Amazon's algorithmic pricing to ensure they were generating marginally more revenue than their main competitor ended up pushing the price of a book on evolutionary biology – Peter Lawrence's *The Making of a Fly* – to \$23,698,655.93.



Artificial Intelligence and competition policy concerns

- Could pricing algorithms learn to collude? Is there empirical evidence? (Assad et al., 2020)
- What could competition authorities do to prevent algorithmic collusion?
- Harrington (2018) proposes the following experimental test:
 1. Create a simulated marketplace populated with learning algorithms.
 2. Let the algorithms repeatedly interact between them over time.
 3. If supracompetitive prices emerge, the algorithms could be candidates to be prohibited.

Literature review

- Calvano *et al.* (2020, AER)
 - ✓ Bertrand game
 - ✓ Q-learning algorithms set prices
 - ✓ Find that algorithms can lead to collusive strategies
- Calvano *et al.* (2021, Int. J. Ind. Organ.)
 - ✓ Cournot game adapted from Green & Porter (1984)
 - ✓ Q-learning algorithms choose quantities
 - ✓ Find that algorithms can lead to collusive strategies
- Klein (2021, Rand J. Econ)
 - ✓ Bertrand game adapted from Maskin & Tirole (1988)
 - ✓ Q-learning algorithms set prices
 - ✓ Find that algorithms achieve collusive outcomes
 - ✓ Find that algorithms converge to focal prices and Edgeworth cycles

This paper

The aim of this paper:

- Extend the results in Klein (2021) by considering stochastic costs
- Analyze how cost uncertainty affects algorithmic collusion

The roadmap to address the aim:

1. Replicate the results of Klein (2021) to serve as a benchmark
2. Simulate the model under stochastic costs
3. Compare the outcomes of the deterministic and stochastic setting

A primer on Q-learning

The objective of the algorithm is to maximize

$$\max_{\{a_t\}_{t=0}^{\infty}} V(s_t) = \sum_{t=0}^{\infty} \delta^t \pi_t(a_t, s_t) \quad \text{s.a} \quad s_{t+1} = g(a_t, s_t) \quad (1)$$


$$s_0 \in \mathbb{R}^n \text{ dado}$$

where the solution is the optimal policy function $h^* : S \rightarrow A$, so $h^*(s_t) = a_t^*$

$$h^*(s_t) = \arg \max_{a_t} V^*(s_t) \quad (2)$$

by *Bellman's principle of optimality*

$$\begin{aligned} V^*(s_t) &= \max_{a_t} [\pi_t(a_t, s_t) + \delta V^*(g(a_t, s_t))] \\ V^* &= T(V^*) \end{aligned} \quad (3)$$

where T is the *Bellman operator*. Assuming that the agent knows the immediate reward function $\pi(\cdot)$ and the state transition function $g(\cdot)$, the optimal policy function h^* is found using the *value iteration method* 

A primer on Q-learning (cont.)

How to find the optimal policy function h^* if the agent does not know the immediate reward function $\pi(\cdot)$ and the state transition function $g(\cdot)$?

Define

$$Q^*(s_t, a_t) \equiv \pi_t(a_t, s_t) + \delta V^*(g(a_t, s_t)) \quad (4)$$

since $V^*(s_t) = \max_{a_t} Q^*(s_t, a_t)$ rewrite as follows

$$Q^*(s_t, a_t) = \pi_t(a_t, s_t) + \delta \max_{a_{t+1}} Q^*(g(a_t, s_t), a_{t+1}) \quad (5)$$

by (3) and (4) it follows that

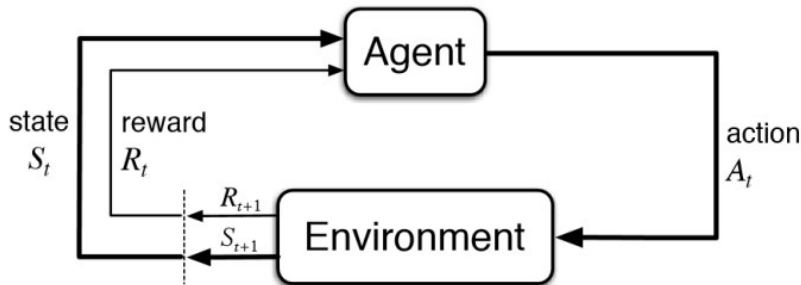
$$h^*(s_t) = \arg \max_{a_t} Q^*(s_t, a_t) \quad (6)$$

Given Q^* , we can find h^* . How to find Q^* ? Using a *Q-Learning* algorithm

How Q-learning algorithm works?

- Starting with an arbitrary \hat{Q}_0 , an estimation \hat{Q}_t is obtained using equation (5) in each period
- Estimates are updated by trial and error
- Model-free approach algorithm
- In a single-agent setting, \hat{Q}_t converges to Q^* as $t \rightarrow \infty$ (Watkins & Dayan, 1992)
- In a multi-agent setting, convergence is not guaranteed

How Q-learning algorithm works? (cont. I)



Sutton & Barto (2018)

How Q-learning algorithm works? (cont. II)

Q-Learning algorithm consists of two modules:

1. **Action-selection module.** Defines how to choose an action

- ε -greedy exploration method:

$$a_t = \begin{cases} a \sim U(A) & \text{with probability } \varepsilon_t \\ \arg \max_a \hat{Q}_t(s_t, a) & \text{with probability } 1 - \varepsilon_t \end{cases}$$

2. **Learning module.** Defines how to learn the quality of each action

$$\hat{Q}_{t+1}(s_t, a_t) = (1 - \alpha) \hat{Q}_t(s_t, a_t) + \alpha \left[\pi_t(s_t, a_t) + \delta \max_{a_{t+1}} \hat{Q}_t(s_{t+1}, a_{t+1}) \right]$$

In this way it obtains the sequence $\{\hat{Q}_0, \hat{Q}_1, \dots, \hat{Q}_t, \hat{Q}_{t+1}, \dots, Q^*\}$

The model: Maskin & Tirole (1988)

- Firms $i = 1, 2$ set prices in turn
- Price space: $P = \{0, \frac{1}{k}, \frac{2}{k}, \dots, 1\}$
- Demand:

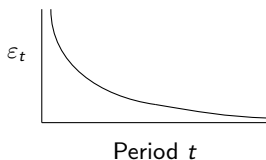
$$D_{it}(p_{it}, p_{jt}) = \begin{cases} 1 - p_{it} & \text{if } p_{it} < p_{jt} \\ \frac{1}{2} (1 - p_{it}) & \text{if } p_{it} = p_{jt} \\ 0 & \text{if } p_{it} > p_{jt} \end{cases}$$

- Marginal cost is high (c_H) with probability ρ or low (c_L) with $1 - \rho$
- Profits: $\pi_{it} = (p_{it} - c_t)D_{it}$
- State space: $S = \{s_t \in \mathbb{R}^2 : s_t = (p_{jt-1}, c_t)\}$ *one-period memory*
- Two equilibria: (1) Focal price and (2) Edgeworth cycle

Calibration

I use the same calibration as Klein (2021) to facilitate comparisons:

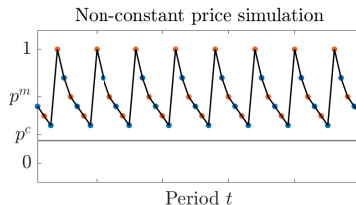
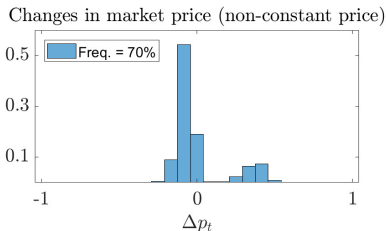
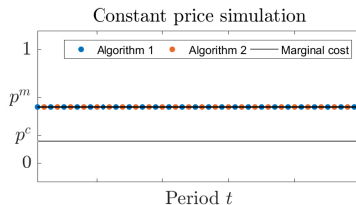
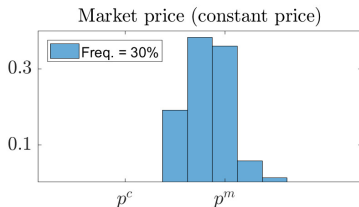
- Price set: $k = 12$
- Discount factor: $\delta = 0.95$
- Probability: $\rho \in \{0.1, 0.5, 0.9\}$
- Learning parameter: $\alpha = 0.3$
- Probability of exploration: $\varepsilon_t = (1 - \theta)^t$ where $\theta = 2.75 \times 10^{-5}$



- Initial state: $s_0 \sim U(S)$
- Time horizon: $T = 500,000$ and Simulations: $R = 300$

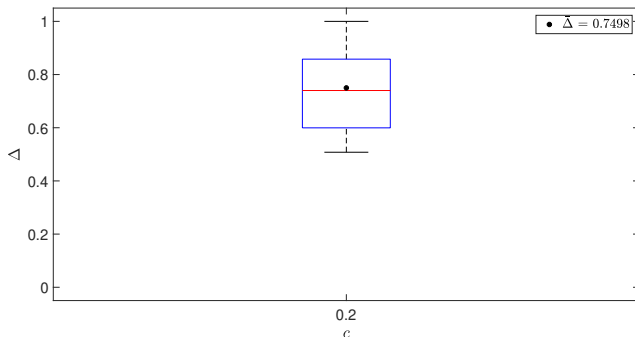
Results: (1) Deterministic cost

- Algorithms converge to a focal price or an Edgeworth cycle



Results: (1) Deterministic cost (cont.)

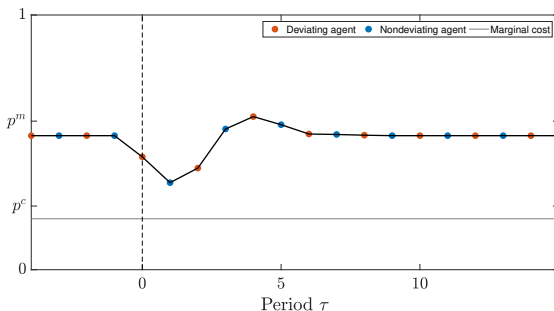
- Algorithms achieve supracompetitive profits



$\Delta = \frac{\bar{\pi} - \pi^C}{\pi^M - \pi^C}$ is the normalized aggregate profit, where $\bar{\pi}$ is the average aggregate profit during the final 1000 periods, π^C is the profit in perfect competition and π^M is the monopoly profit.

Results: (1) Deterministic cost (cont. I)

- Algorithms learn strategies that embody a reward-punishment scheme to sustain supracompetitive outcomes



One algorithm is forced to deviate in period $\tau = 0$. The deviation lasts for one period. The figure plots the average prices for those simulations in which the algorithms have converge to a focal price equilibria.

Results: (2) Stochastic cost

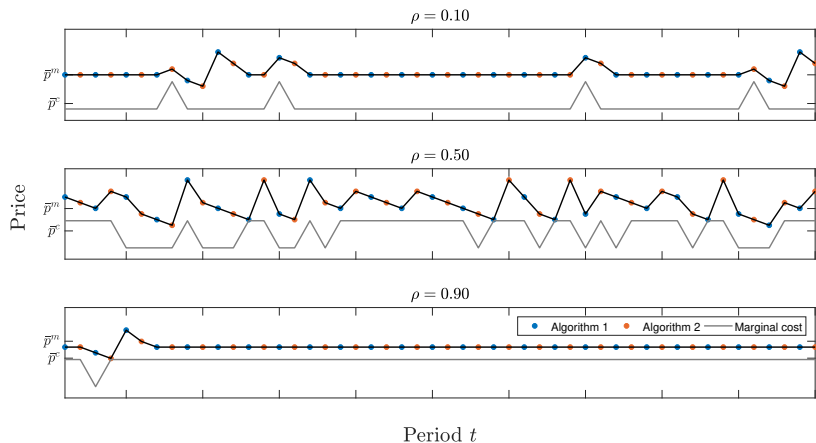
- Algorithms converge to a focal price equilibria or an Edgeworth cycle provided that uncertainty is not too large. When the market environment becomes more uncertain, Edgeworth cycle is the only possible pricing pattern

		Probability ρ											
		0.10				0.50				0.90			
Pattern	Freq.	\bar{p}^c	\bar{p}^m	\bar{p}	Freq.	\bar{p}^c	\bar{p}^m	\bar{p}	Freq.	\bar{p}^c	\bar{p}^m	\bar{p}	
constant	11%	0.08	0.50	0.45	-	-	-	-	41%	0.42	0.67	0.61	
non-constant	89%	0.08	0.50	0.40	100%	0.25	0.58	0.47	59%	0.42	0.67	0.58	

Note: constant pricing pattern is deemed to be achieved if the market price remains constant over at least the 50% of the last 1000 periods Otherwise, a simulation attains a non-constant pricing pattern.

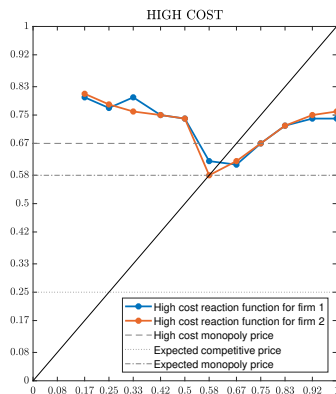
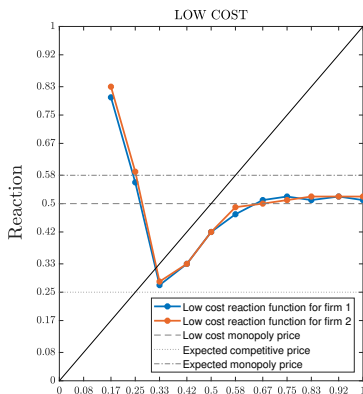
Results: (2) Stochastic cost (cont.)

Constant and non-constant pricing patterns



Results: (2) Stochastic (cont. I)

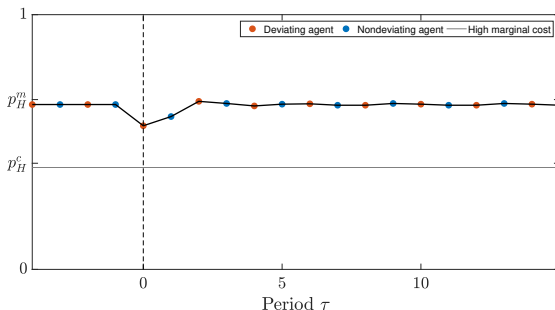
- Algorithms tend to gravitate towards a focal price in the high-cost state and they engage in a price war in the low-cost state



Rivals price in previous period

Results: (2) Stochastic (cont. II)

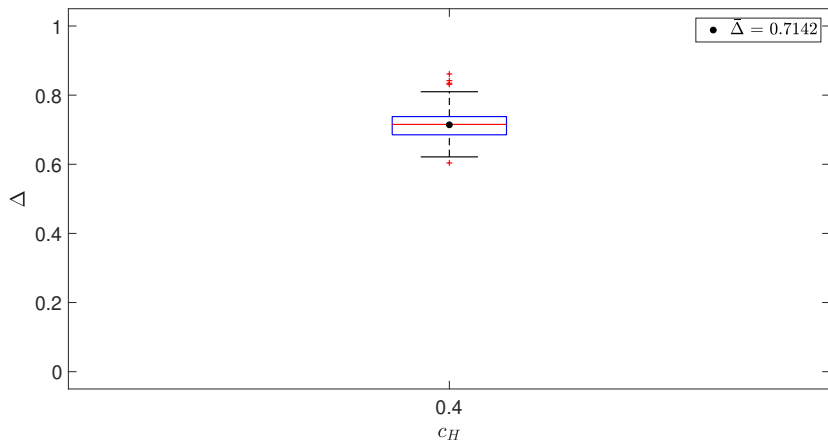
- The price coordination in the high-cost state arises from a reward-punishment scheme



Marginal cost is set equal to the high state and it remains constant over time. Then one algorithm is forced to deviate in period $\tau = 0$. The deviation lasts for one period. The figure plots the average prices for those simulations in which the algorithms have converge to a focal price equilibria.

Results: (2) Stochastic cost (cont. III)

- Algorithms achieve supracompetitive profits, even when they compete under uncertainty



Results: (2) Stochastic cost (cont. IV)

- **Uncertainty reduces profit gains by 3.56%**

The impact of uncertainty on collusion

$$\begin{aligned}\text{Impact of uncertainty} &= \bar{\Delta}_{\text{Stochastic}} - \bar{\Delta}_{\text{Deterministic}} \\ &= 71.42\% - 74.98\% \\ &= -3.56\%\end{aligned}$$

Conclusions and future research

Conclusions:

- Uncertainty makes collusive outcomes more difficult to achieve
- Price wars could be observed in markets where costs frequently vary over time and firms use sequential Q-learning algorithms to set prices

Future research:

- Competition under heterogeneous algorithms (Sanchez-Carta & Katsamakas, 2021).
- Can self-learning algorithms and humans cooperate to achieve collusion? (Crandall *et al.*, 2019; Werner, 2021)

Future perspectives



It's true that the idea of automated systems getting together and reaching a meeting of minds is still science fiction...But we do need to keep a close eye on how algorithms are developing...so that when science fiction becomes reality, we are ready to deal with it

Margarethe Vestager (2017)
EU Commissioner for Competition