Algorithmic collusion under sequential pricing and stochastic costs

Gonzalo Ballestero*

March 2026

Abstract

The use of pricing algorithms raises concerns about algorithmic collusion. This paper considers a sequential pricing model where marginal cost fluctuates over time. I find that Q-learning algorithms autonomously collude even under cost uncertainty. Collusion is sustained by strategies that involve reward-punishment schemes. It suggests that cost uncertainty is not an obstacle to autonomous algorithmic collusion.

Keywords: Competition Policy, Artificial Intelligence, Collusion

JEL Codes: D43, K21, L13

*Pennsylvania State University. E-mail: gballestero@psu.edu. I am grateful for the valuable discussions with Ran I. Shorrer. I thank Nageeb Ali, Joris Pinkse, Lucia Quesada, Conor Ryan, and Jidong Zhang. I would also like to thank the editor, Giacomo Calzolari, and anonymous referees for their valuable comments. This paper benefited from the comments of the participants to the EARIE 2024 Summer School at the University of Amsterdam, the Midwest Economic Theory Conference at Virginia Tech, the PSU Applied Micro Brown Bag, the PSU Theory Lunch, and the Pennsylvania Economic Theory Conference. All errors and omissions are my own.

1 Introduction

Algorithms are increasingly being used to set prices (Brown and MacKay, 2021; Chen et al., 2016; Musolff, 2022). The use of algorithms has raised concerns among policymakers and competition authorities about the possibility that algorithms may autonomously collude (Calvano et al., 2020a; Descamps et al., 2021; Ezrachi and Stucke, 2017; Harrington, 2018; OECD, 2017). A growing literature shows through computational simulations that self-learning algorithms may reach supra-competitive prices, which are sustained by strategies that involve a reward-punishment scheme (Calvano et al., 2020b; Fish et al., 2024; Klein, 2021). However, as there has been some skepticism about the possibility of autonomous collusion emerging in practice (see, for e.g., Deng 2018; Dorner 2021), these studies acknowledge the need for further research to establish the robustness of these findings.

In this paper, I analyze whether autonomous algorithmic collusion can arise under cost uncertainty. Previous research has focused on other sources of uncertainty, such as stochastic demand, random entry and exit (see Calvano et al. 2020b, 2021). However, pricing algorithms are increasingly used in markets where costs fluctuate over time. One example is the gasoline market, where gas stations face common cost shocks over time as the crude oil price fluctuates. To adapt to these changing circumstances, gasoline retailers use artificial intelligence software to adjust prices quickly (Assad et al., 2024).[1] It is therefore important to study the possibility of autonomous algorithmic collusion emerging under cost uncertainty.

I extend the sequential algorithmic pricing duopoly model of Klein (2021) by allowing for stochastic marginal costs. I consider a Bernoulli marginal cost process, in which marginal costs can be either high or low with equal probability, and also examine the effects of introducing persistence through a Markov process. My main result is that Q-learning can collude under cost uncertainty in a sequential pricing setting. Collusion is sustained by strategies that involve reward-punishment schemes. It suggests that cost uncertainty is not an insurmountable obstacle to autonomous algorithmic collusion. These findings indicate that

[1] See also https://www.wsj.com/why-do-gas-station-prices-constantly-change-blame-the-algorithm.

autonomous algorithmic collusion may be more than merely a theoretical possibility. The strategies that algorithms learn purely by trial and error are remarkably similar to those considered by Eckert (2004), Maskin and Tirole (1988), and Noel (2008). That is, algorithms learn to coordinate on a collusive equilibrium in which algorithms match each other at a price level that varies with marginal costs. In addition, algorithms often converge to an Edgeworth price cycling pattern in which they gradually undercut one another, and then the price jumps to initiate a new cycle. Evidence shows that the use of repricing algorithms yields a similar pricing pattern in online markets (e.g., Edelman and Ostrovsky 2007; Musolff 2022).

Following the tradition of previous research, this paper is intended as a proof-of-concept demonstration. There are two important caveats to keep in mind. First, in this paper, I focus on symmetric Q-learning algorithms and do not consider more state-of-the-art algorithms. Maintaining the focus on the same type of algorithms facilitates comparisons with previous findings in the literature on algorithmic collusion. Secondly, this paper focuses on a single source of uncertainty, namely, stochastic marginal costs. Other sources remain to be addressed.

The remainder of the paper is organized as follows. Section 2 provides a review of the literature. Section 3 describes the economic environment and the algorithms. Section 4 discusses the results. Section 5 shows that the results are robust to the modeling choices. Finally, Section 6 concludes with a brief discussion of the limitations of the analysis and future research directions.

2 Related Literature

This article contributes to the growing literature on algorithmic collusion. In a seminal paper, Calvano et al. (2020b) show that Q-learning algorithms, a simple type of reinforcement learning algorithm, may reach supra-competitive prices in an infinitely repeated Bertrand

game without explicit instructions to do so. Most importantly, those high prices are sustained by collusive strategies that involve a reward-punishment scheme.[2] Their findings are robust to demand uncertainty and random entry. Klein (2021) attain similar results in the sequential repeated pricing game.[3] Asker et al. (2024) consider Q-learning but with a more sophisticated *synchronous* learning protocol (see Calvano et al. (2023) for a deeper discussion). Hansen et al. (2021) provide evidence for another classic learning algorithm. Hettich (2021) consider a related algorithm called Deep Q-learning. Using a large language model, Fish et al. (2024) show that LLM-based pricing agents autonomously collude in oligopoly settings. A more extensive discussion of the literature is provided by Abada et al. (2024a). See also Assad et al. (2021) and Kühn and Tadelis (2018) for policy implications.[4]

Extending the existing literature, a recent work by Calvano et al. (2021) shows that Q-learning algorithms can learn to collude under demand uncertainty and imperfect monitoring.[5] To facilitate comparisons, I also focus on Q-learning, but the main difference is that I consider another source of uncertainty, namely fluctuating marginal costs. This major departure is motivated by the fact that in many markets the marginal costs are inherently volatile, including notably the gasoline retailing market, as discussed above. In line with Calvano et al. (2020b, 2021), I show that algorithmic collusion can arise under cost uncertainty in a sequential pricing duopoly game. The algorithms exhibit pricing dynamics consistent with Markovian equilibria. These findings extend the results of Klein (2021) by allowing for

[2]This interpretation remains a matter of debate (see, e.g., den Boer et al. (2022), Epivent and Lambin (2024) and Lambin (2024)). In addition, Eschenbaum et al. (2022) criticize the claim that reinforcement algorithms with collusive pricing policies can successfully extrapolate collusive behavior from training environments to the market. Furthermore, Abada et al. (2024b) show that self-trained Q-learning can be exploited by more sophisticated algorithms.

[3]Waltman and Kaymak (2008) show that Q-learning algorithms can lead to supra-competitive prices in a repeated Cournot game. A similar finding was shown by Kimbrough and Murphy (2009) using a different learning algorithm. Banchio and Skrzypacz (2022) show that Q-learning leads to collusive outcomes in first-price auctions. Johnson et al. (2023) study platform design to reduce the harm to consumers from autonomous collusion by pricing algorithms based on Q-learning. Dou et al. (2024) show that adopting Q-learning-based trading algorithms can lead speculators to sustain supra-competitive profits. Werner (2026) studies collusion with human and Q-learning algorithms in lab experiments.

[4]A recent line of work discusses approaches for detecting collusive behaviors by auditing algorithms (Arunachaleswaran et al., 2024; Hartline et al., 2024).

[5]Abada and Lambin (2023) consider a model of imperfect monitoring with no uncertainty.

fluctuating marginal costs. Furthermore, this paper is also aligned with previous results in the literature on computational methods for oligopoly models (see Doraszelski and Pakes 2007; Ericson and Pakes 1995; Farias et al. 2012; Pakes and McGuire 2001). Noel (2008) employs a dynamic programming approach to simulate the canonical sequential repeated pricing game when marginal costs vary from period to period with no serial correlation. This paper also finds that numerical simulations converge to Markov perfect equilibria. However, the approach I consider in this article differs from dynamic programming as Q-learning is a model-free algorithm (i.e., it does not require any prior knowledge about the underlying economic environment) that learns purely by trial and error.

Empirical work on algorithmic pricing has also been flourishing. Assad et al. (2024) document a rise in margins of retail gasoline stations from the adoption of automated pricing algorithms. Brown and MacKay (2021) study the effect of high-frequency price adjustments on competition between online retailers. They find that the adoption of pricing algorithms that allow for more frequent price changes can increase price levels. Chen et al. (2016) show that in 2015, more than one-third of the best-selling Amazon products adopted algorithmic pricing, and these tended to have higher prices and sales. Musolff (2022) find that the use of rule-based pricing algorithms by online retailers leads to a pricing pattern similar to Edgeworth cycles that decreases competition. However, these studies treat algorithms as black boxes. In this study, I examine how pricing algorithms operate in a computer-simulated marketplace, which allows me to fully understand their behavior.

Finally, another strand of the literature studies how algorithms influence competition by offering better demand predictions (Martin and Rasch, 2024; Miklós-Thal and Tucker, 2019; O'Connor and Wilson, 2021) or by serving as commitment devices (Lamba and Zhuk, 2022; Salcedo, 2015). However, these studies do not analyze whether algorithms can learn to collude, but whether their use can affect the pricing game such that the equilibrium involves higher prices. In contrast, I investigate the collusion capacity of pricing algorithms.

3 Experimental design

This section describes the economic environment used in the computational simulations and the theoretical equilibrium outcomes that will guide the analysis. It also provides a description of the algorithms used.

3.1 Economic Environment

Following Klein (2021), I consider a sequential pricing duopoly model adapted from Maskin and Tirole (1988). This model has traditionally been used in empirical studies of the gasoline market because it matches well many features of the pricing dynamics observed in those markets (see Eckert 2013, for a comprehensive review). There are two firms, indexed by $i = 1, 2$, that compete on price in a market for a homogeneous good. Competition takes place in infinitely repeated discrete time indexed by $t = 0, 1, 2, ...$. Firm 1 chooses a price in every odd period, and firm 2 sets its price in even periods. This timing reflects commitments to price for two periods. The price space is discrete and it is given by $P = \{0, \frac{1}{k}, \frac{2}{k}, ..., 1\}$, where $k > 0$. Each firm aims to maximize its cumulative stream of discounted future profits

$$\max \sum_{t=0}^{\infty} \delta^t \pi_i(p_{it}, p_{jt}, c_t), \tag{1}$$

where $\delta \in (0, 1)$ is the discount factor. It is assumed that there are no fixed costs or capacity constraints, therefore the profit of firm i at time t can be written as

$$\pi_i(p_{it}, p_{jt}, c_t) = (p_{it} - c_t)\, D_i(p_{it}, p_{jt}), \tag{2}$$

where D_i is the demand function, which is defined as follows

$$D_i(p_{it}, p_{jt}) = \begin{cases} 1 - p_{it} & \text{if} \quad p_{it} < p_{jt} \\ \frac{1}{2}(1 - p_{it}) & \text{if} \quad p_{it} = p_{j,t} \\ 0 & \text{if} \quad p_{it} > p_{jt} \end{cases}. \tag{3}$$

Both firms have the same marginal costs, c_t. I extend Klein (2021) by supposing that marginal costs are stochastic and follow a Markov process. Marginal costs can be either low (c_L) or high (c_H) in any period. That is, firms face common cost shocks over time. Each firm observes the current marginal cost before setting its price. The probability that in any period the marginal cost will remain unchanged from the previous period is $\rho \in (0, 1)$, and the probability that the costs switch states is $1 - \rho$. These assumptions imply that the marginal cost follows an irreducible, two-state Markov chain. Therefore, there exists a unique stationary distribution, denoted by $\lambda = (\lambda_L, \lambda_H)$, where λ_L and λ_H represent the long-run probabilities of being in the low- and high-cost states, respectively. In the symmetric case considered here, the stationary distribution is $\lambda = (0.5, 0.5)$.

Equilibrium. In this paper, I assume that firms use algorithms to set prices. However, this section describes the theoretical equilibrium outcomes, which serve as a benchmark to guide the analysis. Following Maskin and Tirole (1988), suppose that strategies only depend on variables that are directly payoff-relevant. This assumption implies that firm i's strategy is a dynamic reaction function, $p_{it} = R_i(p_{jt-1}, c_t)$, which depends on the price set previously by i's rival and on the current marginal cost. The assumption that marginal costs follow a Markov process implies that players only need to know the current realization of the marginal cost.

Extending the seminal analysis of Maskin and Tirole (1988), Eckert (2004) and Noel (2008) show that three classes of symmetric Markov perfect equilibria (MPE) are possible for a sufficiently high discount factor when marginal costs fluctuate: single focal price,

alternating focal price, and Edgeworth cycle equilibria.[6] Single focal price equilibria are characterized by constant prices over time regardless of current marginal cost. Formally, define a single focal price equilibrium as one in which, for some price p^f, $R(p^f, c_H) = R(p^f, c_L) = p^f$. That is, in a single focal price equilibrium, there exists a price that, if set by a firm's rival in the previous period, the firm will match no matter the current marginal costs. In contrast, in an alternating focal price equilibrium, the focal price depends on the current marginal cost. Formally, in an alternating focal price equilibrium, there are some prices p_L^f and p_H^f such that $R(p_L^f, c_L) = p_L^f$, $R(p_H^f, c_H) = p_H^f$ but $R(p_L^f, c_H) \neq p_L^f$ and $R(p_H^f, c_L) \neq p_H^f$. That is, a firm matches p_L^f when costs are low but not when costs are high, and matches p_H^f when costs are high but not when they are low. There exists a multiplicity of focal prices, which are bounded away from the marginal costs. Note that there are no restrictions on the prices set by a firm in response to prices other than the focal prices. Finally, in the Edgeworth cycle equilibria, firms gradually undercut each other. Once the prices have fallen to the current marginal cost, undercutting ceases, and firms play a war of attrition against each firm mixing between raising the price back above the static monopoly price and remaining at the current marginal cost. Importantly, note that Edgeworth cycles involve prices above marginal cost.

3.2 Pricing algorithms

Following Klein (2021), I assume that firms delegate their pricing decisions to an algorithm called Q-learning. There are several reasons for focusing on Q-learning algorithms. First, Q-learning is one of the most relevant and popular reinforcement learning algorithms, and many of the state-of-the-art reinforcement learning algorithms build on the main ideas of Q-learning (see Arulkumaran et al. 2017). Although little is known about the specific algorithm that companies actually use, Assad et al. (2024) reports that gasoline retailers adopt

[6] Maskin and Tirole (1988) shows that there are two classes of symmetric MPE when marginal costs are constant over time: focal price and Edgeworth cycle equilibria. Allowing for stochastic marginal costs enlarges the set of symmetric MPE. See Maskin and Tirole (2001) for general properties of MPEs.

pricing software based on AI-learning algorithms. Furthermore, Calzolari and Hanspach (2024) indicate that 27% of third-party pricing tools use self-learning AI algorithms. These observations highlight that learning algorithms are popular in practice, and firms have access to this pricing technology. Second, Q-learning algorithms can be fully characterized by just a few parameters, the economic interpretation of which is clear. This allows potentially arbitrary modeling choices to be kept to a minimum (Calvano et al., 2020b). Third, the literature on algorithmic collusion has mainly focused on Q-learning (see Abada et al. 2024b, for a review). Maintaining the focus on the same type of algorithms facilitates comparisons with previous findings.

Q-learning is a reinforcement learning algorithm that aims to maximize the expected discounted value of future profits. A powerful feature of Q-learning is that it does not require any prior knowledge of the underlying economic environment. The algorithm only learns through experience, associating states and actions (i.e., prices) with the payoff they generate. Specifically, the algorithm starts from an arbitrary set of initial values attached to all feasible actions and updates those values according to some pre-specified rule. It is this updating rule that determines how the algorithm learns. Following the one-period memory assumption, each algorithm i observes the rival's previous price, the current marginal cost, and the marginal cost from the previous period before setting its own price. This information defines the algorithm's state variable, $s_{it} = (p_{j,t-1}, c_{t-1}, c_t)$.[7] After observing the realized state, the algorithm selects a price $p_{it} \in P$ using some action-selection policy (i.e., ε-greedy policy). It then observes the resulting outcome and updates the value it attaches to the chosen action. By trying all actions in each state repeatedly, it learns which are best overall, judged by long-term discounted profits. Appendix A provides a more detailed description of these algorithms. For a textbook treatment on Q-learning, see Sutton and Barto (2018). Calvano et al. (2020b) and Johnson et al. (2023) also provide a general primer on Q-learning.

[7]This differs from the theoretical model described in Section 3.1, where the state consists of the rival's previous price and the current marginal cost. Here, I include both current and lagged marginal costs since algorithms may require additional information to distinguish deviations from cost-driven price changes. See Section 4.2 for a discussion.

Calibration, baseline specification, and convergence criterion. To facilitate comparisons, I use the same baseline calibration as in Klein (2021) and standard hyperparameters.[8] The learning rate is set at $\alpha = 0.15$, and the exploration rate is $\beta = 4 \times 10^{-6}$. The discount factor δ is 0.95. I set $k = 12$ price intervals between 0 and 1, so the cardinality of the price space is $|P| = 13$. The low marginal cost is set to $c_L = 0$ and the high marginal cost $c_H = \frac{1}{6}$. These magnitudes were deliberately chosen to ensure that the static Nash (i.e., one increment above marginal cost) and monopoly price are in the action space.[9] The set of states is $S = P \times \{c_L, c_H\} \times \{c_L, c_H\}$. Thus, the Q-matrix has $2^2 \times 13^2 = 676$ entries. The Q-values are initialized with all zeros, although results are not sensitive to initialization.

I consider two relevant values for the cost persistence parameter. First, setting $\rho = 0.9$ reflects high persistence, where current marginal costs are informative about future values. Although changes to marginal costs are largely unexpected, any shock has long-term implications. Alternatively, setting $\rho = 0.5$ implies that costs are equally likely to be high or low in any given period, with the same probability of changing or remaining the same in the next period. Note that in this case, marginal costs follow a Bernoulli distribution and so agents face sizable uncertainty.

I run $1,000$ sessions until convergence. Each session is run under the same set of parameters, and the initial state is drawn randomly at the beginning of each session. Following Calvano et al. (2020b), in each session, convergence is said to be achieved, and the session is stopped if the best action for each state does not change for 100,000 consecutive periods for each agent. In any case, each session is stopped after one billion iterations. In spite of the lack of any theoretical guarantee, the algorithms always achieve a stable behavior and converge to a limit strategy. However, the learning process is slow, and convergence typically takes millions of iterations. Appendix B.1 shows that algorithms require, on average, over 3 million iterations to achieve convergence.

[8] I refer to Calvano et al. (2020b) for a discussion of the choice of the parameter values.

[9] Define the static Nash price as $p^N = c + \frac{1}{k}$ and the static monopoly price as $p^M = \frac{1+c}{2}$, where the marginal cost, c, can be either c_L or c_H.

4 Results

4.1 Prices, profits and equilibrium play

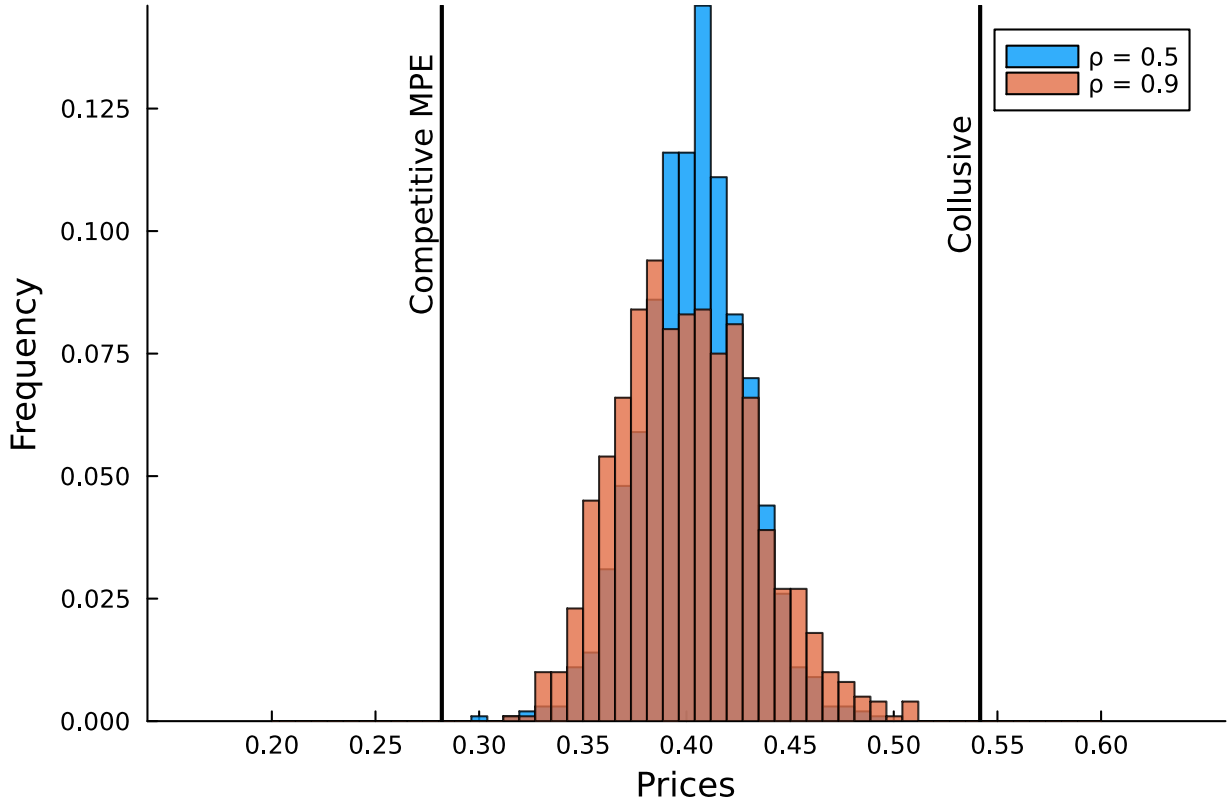
I focus on the limiting behavior of the algorithms and evaluate simulation outcomes against two benchmarks: (i) the collusive benchmark and (ii) the competitive benchmark. The collusive benchmark is defined as the outcome where firms jointly maximize profits. That is, firms set the static low-cost monopoly price when the marginal cost is low and the static high-cost monopoly price when the marginal cost is high. Eckert (2004) show that a tacitly collusive alternating focal price equilibrium supporting the monopoly prices can be sustained through strategies similar to those constructed by Maskin and Tirole (1988) under certain conditions. A deviation from the tacitly collusive focal price is responded to with a price sufficiently low to force a restoration. The competitive benchmark is not straightforward. A natural candidate would be the Bertrand-Nash equilibrium. However, because of the sequential move assumption, an equilibrium where prices are equal to marginal costs cannot exist in this framework. For this reason, the competitive benchmark is defined as the most competitive MPE, which is the Edgeworth cycle.

Prices. Having defined the relevant benchmarks, I analyze the average market price set by the algorithms upon convergence. Figure 1 shows the average market price across 1,000 sessions. The solid vertical lines represent the benchmarks discussed above. Given that the stationary distribution of marginal costs is given by $\lambda = (0.5, 0.5)$, the competitive benchmark is the average price resulting from playing the Edgeworth cycle MPE separately under low and high marginal costs.[10] Similarly, the collusive benchmark is the average price that would occur from alternating between low and high monopoly prices. The figure shows that the average market price tends to be above the competitive level.

I then consider the limit behavior that underpins the non-competitive prices described above. Based on the learned strategy upon convergence, I classify each session according to

[10]See Appendix B.2 for details about the empirical cost distribution.

Figure 1: Average market price upon convergence



Notes: The solid lines represent the relevant benchmarks. The competitive benchmark is the Edgeworth cycle MPE.

the equilibrium pricing patterns implied by the corresponding MPE discussed in Section 3.1. Table 1 reports that the vast majority of sessions display price cycles, most of which have a period of 8, while a small fraction of sessions converge to an alternating focal price.[11] All other sessions exhibit pricing behavior referred to as partial focal. That is, algorithms coordinate on a fixed price at one cost state level, but engage in a price cycle at the other cost state. See Appendix B.2 for an illustration of the pricing patterns described above, and more detailed information about the observed cycles.

Although the pricing patterns observed in the simulations resemble the MPE described by Eckert (2004), Maskin and Tirole (1988), and Noel (2008), there are two important

[11]Musolff (2022) documents that online retailers adopt pricing algorithms that lead to a pricing pattern consistent with Edgeworth cycles. Leisten (2024) builds a theoretical model of algorithmic pricing and managerial override that also generates asymmetric price cycles.

distinctions. First, the nature of the strategies differs. The theoretical MPE often requires mixed strategies (either off-equilibrium or along the equilibrium path), whereas Q-learning converges to pure strategies by design (i.e., the algorithm always selects the action with the highest learned Q-value in each state). Second, there is a key difference in the information structure. The theoretical model defines the state as $s_{it} = (p_{jt-1}, c_t)$, but in the simulations it is $s_{it} = (p_{jt-1}, c_{t-1}, c_t)$. I include c_{t-1} so that algorithms can, in principle, distinguish between price changes caused by exogenous cost shocks and unilateral deviations from the implicit collusive agreement. See Section 4.2 for a discussion.

Profits. All sessions typically display average market prices above the competitive level. This suggests that algorithms gain supra-competitive profits. To facilitate comparisons, I use a normalized measure of profits gained:

$$\Delta = \frac{\bar{\pi} - \bar{\pi}^C}{\bar{\pi}^M - \bar{\pi}^C}, \tag{4}$$

where $\bar{\pi}$ is the average per-firm profit upon convergence, $\bar{\pi}^M = 0.106$ (i.e., 0.125 with probability 0.5 and 0.087 with probability 0.5) is the profit under full collusion (monopoly), and $\bar{\pi}^C = 0.059$ (i.e., 0.070 with probability 0.5 and 0.047 with probability 0.5) is the average per-period profit under the Edgeworth cycle equilibrium. Thus, $\Delta = 0$ corresponds to the competitive outcome and $\Delta = 1$ to the perfectly collusive outcome. Table 1 shows that the profit gain is about $\Delta = 0.52$ for both values of the cost persistence parameter. The profit gains are expected to be similar since, in the long run, the fraction of time the marginal cost spends in the low state is 0.5 for any $\rho \in (0, 1)$. Collusion is not full, yet algorithms reach substantial supra-competitive profits.[12]

Equilibrium play. Even if the algorithms converge to a limit strategy, this may not

[12]To ensure that supra-competitive profits are not merely an artifact of the price grid being skewed toward high values, I compute the normalized profit measure using profits under uniform random pricing, where each firms draw prices uniformly from the action set. Profits under random pricing are $\bar{\pi}^R = 0.048$ (i.e., 0.076 with probability 0.5 and 0.020 with probability 0.5). Using this value yields $\Delta = -0.23$, indicating that random pricing leads to lower profits than the competitive benchmark. This confirms that the supra-competitive profits observed in the simulations cannot be attributed to the structure of the price grid.

be an optimal response to that of the rival. To verify if algorithms learn to best-response to rival's limit strategy, I calculate the theoretical Q-matrix under the assumption that the rival uses his limit strategy. With these Q-matrices, I then determine the algorithms' optimal strategies and compare them to their own limit strategies. Table 1 reports that in around 28 percent of the Bernoulli sessions and 26 percent of the Markov sessions, both algorithms play a best response to the rival's limit strategy, on path. When an algorithm is not playing a best response, it is also possible to calculate the lost payoff both on path and over all states. I express this in percentage terms and refer to it as the Q-loss, as in Calvano et al. (2020b). The average Q-loss on path is 8.4 percent in the Bernoulli case and 13.4 percent in the Markov case. The corresponding average Q-loss over all states is higher, 11.6 percent for Bernoulli and 19.3 percent for Markov. These findings suggest that learning optimal responses is more challenging under cost uncertainty than in Klein (2021).

4.2 Collusive strategies

The previous subsection concludes that algorithms coordinate on supra-competitive prices. Following the approach used in Calvano et al. (2020b) and Klein (2021), I analyze whether the algorithms learned strategies that embody a reward-punishment scheme.[13] Starting, in period $\tau = 0$, from the prices the algorithms have converged to, I exogenously force one algorithm to deviate in period $\tau = 1$ by undercutting its competitor. The other algorithm instead continues playing its learned strategy. The deviation lasts for one period only, and the deviating agent always chooses the most profitable deviation. By doing this, I observe the reaction of the algorithms in the subsequent periods when the forced cheater reverts to its learned strategy.

Figure 2 depicts the average response to price deviations by cost state for different values of the persistence parameter ρ.[14] It shows that deviations are punished, and prices return

[13] Xie and Chen (2004) use a similar approach.

[14] When the algorithms converge to a cycle, I consider deviations starting from every point of the cycle and take the average of all of them.

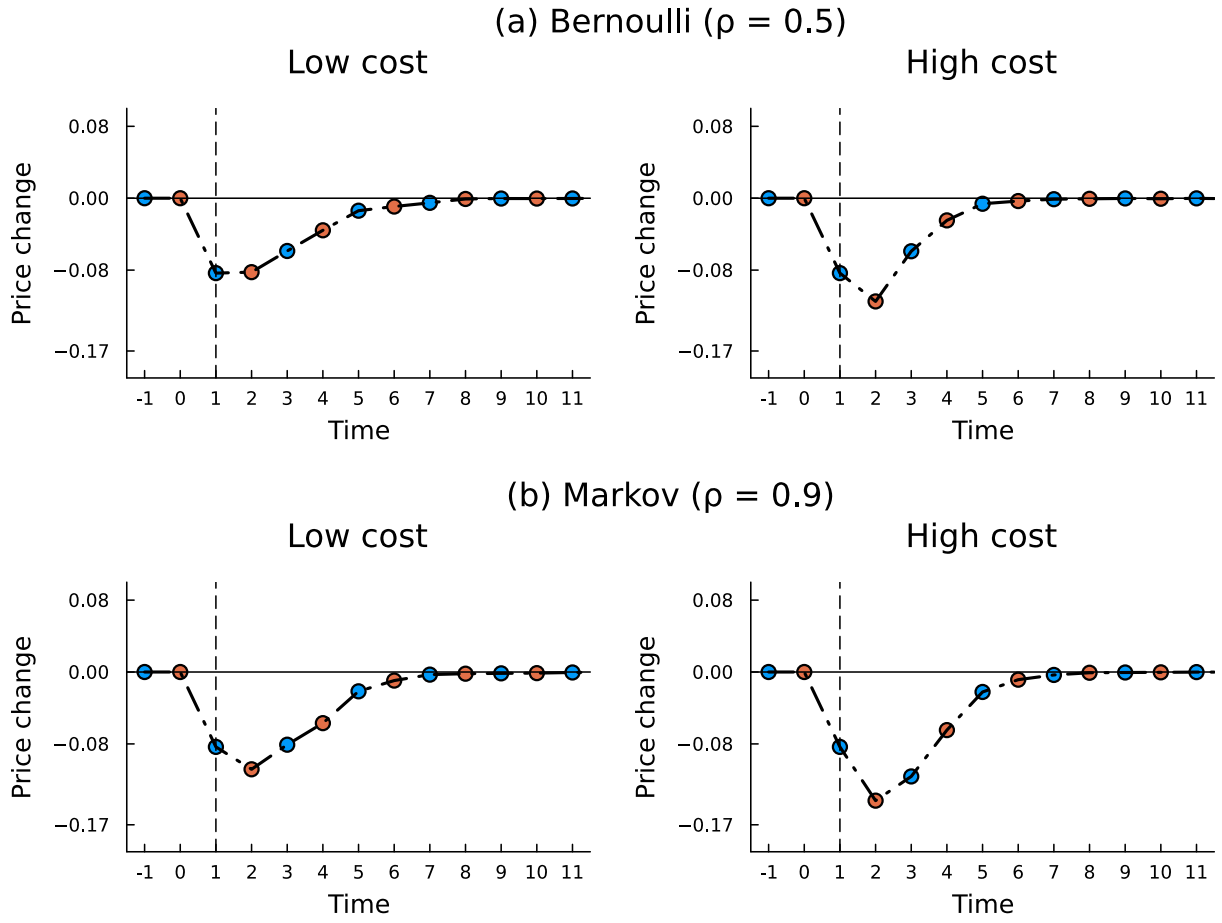
Table 1: Descriptive statistics

	Alternating focal	Partial focal	Cycle	All
Panel A: Bernoulli ($\rho = 0.5$)				
Frequency	0.019	0.169	0.812	1.000
Average market price	0.401	0.400	0.404	0.404
Average normalized profit	0.609	0.539	0.519	0.524
Standard deviation normalized profit	0.111	0.099	0.077	0.083
Frequency of Nash equilibria	0.421	0.249	0.292	0.287
Average Q-loss (on path)	0.117	0.098	0.080	0.084
Standard deviation Q-loss (on path)	0.044	0.048	0.050	0.050
Average Q-loss (all states)	0.159	0.135	0.112	0.116
Standard deviation Q-loss (all states)	0.051	0.060	0.059	0.060
Panel B: Markov ($\rho = 0.9$)				
Frequency	0.039	0.325	0.636	1.000
Average market price	0.410	0.406	0.398	0.401
Average normalized profit	0.693	0.572	0.491	0.525
Standard deviation normalized profit	0.132	0.105	0.103	0.116
Frequency of Nash equilibria	0.385	0.329	0.220	0.262
Average Q-loss (on path)	0.095	0.118	0.145	0.134
Standard deviation Q-loss (on path)	0.057	0.077	0.087	0.084
Average Q-loss (all states)	0.198	0.191	0.194	0.193
Standard deviation Q-loss (all states)	0.060	0.080	0.094	0.089

to the pre-deviation level after a few periods. Importantly, since marginal costs vary over time, a decrease in price after the deviation may be because the other algorithm triggers a punishment or simply because a downward cost shock occurs and the agent responds to it (i.e., the next section shows that downward cost shocks trigger price reductions and upward cost shocks lead to price increases). To avoid such confounding effects, marginal cost is set to either the low or high state and remains constant over time.

Although Figure 2 shows that average prices gradually return to the pre-deviation level, this is actually the consequence of different sessions jumping up in price in different periods. Appendix C.1 reports more information on the distribution of the price responses. Table 2 shows that these deviations are unprofitable. The nondeviating algorithm reduces the price in period $\tau = 2$ by 8 to 30 percent, and the punishment typically ends after $4 - 5$ periods.

Figure 2: Punishment after a unilateral deviation by cost state



Notes: One algorithm is forced to deviate in period $\tau = 1$. The deviation lasts for one period only. To avoid confounding effects, the marginal cost is fixed. The figure plots the average price. For sessions leading to a price cycle, I consider deviations starting from every point of the cycle and take the average of all of them. The variable on the vertical axis is the difference between the current and the long-run price. The top two panels depict the results for $\rho = 0.5$. The bottom two panels depict the results of $\rho = 0.9$.

Importantly, due to the one-period memory assumption, algorithms are capable of distinguishing whether a price cut is driven by exogenous cost shocks or represents a deviation from the implicit agreement. Consequently, punishments are effectively triggered by unilateral deviations rather than spurious effects. Appendix C.2 shows that price hikes also lead to price wars. As in Epivent and Lambin (2024), algorithms learn to punish any (upward or downward) price deviation.

Table 2: Unprofitability of a unilateral deviation

	Low-cost state	High-cost state
Panel A: Bernoulli ($\rho = 0.5$)		
Relative price change by the nondeviating agent in period $\tau = 2$	-0.304	-0.080
Average percentage gain from the deviation in terms of discounted profits	-0.093	-0.079
Length of punishment	3.998	4.202
Panel B: Markov ($\rho = 0.9$)		
Relative price change by the nondeviating agent in period $\tau = 2$	-0.313	-0.198
Average percentage gain from the deviation in terms of discounted profits	-0.102	-0.083
Length of punishment	4.525	4.555

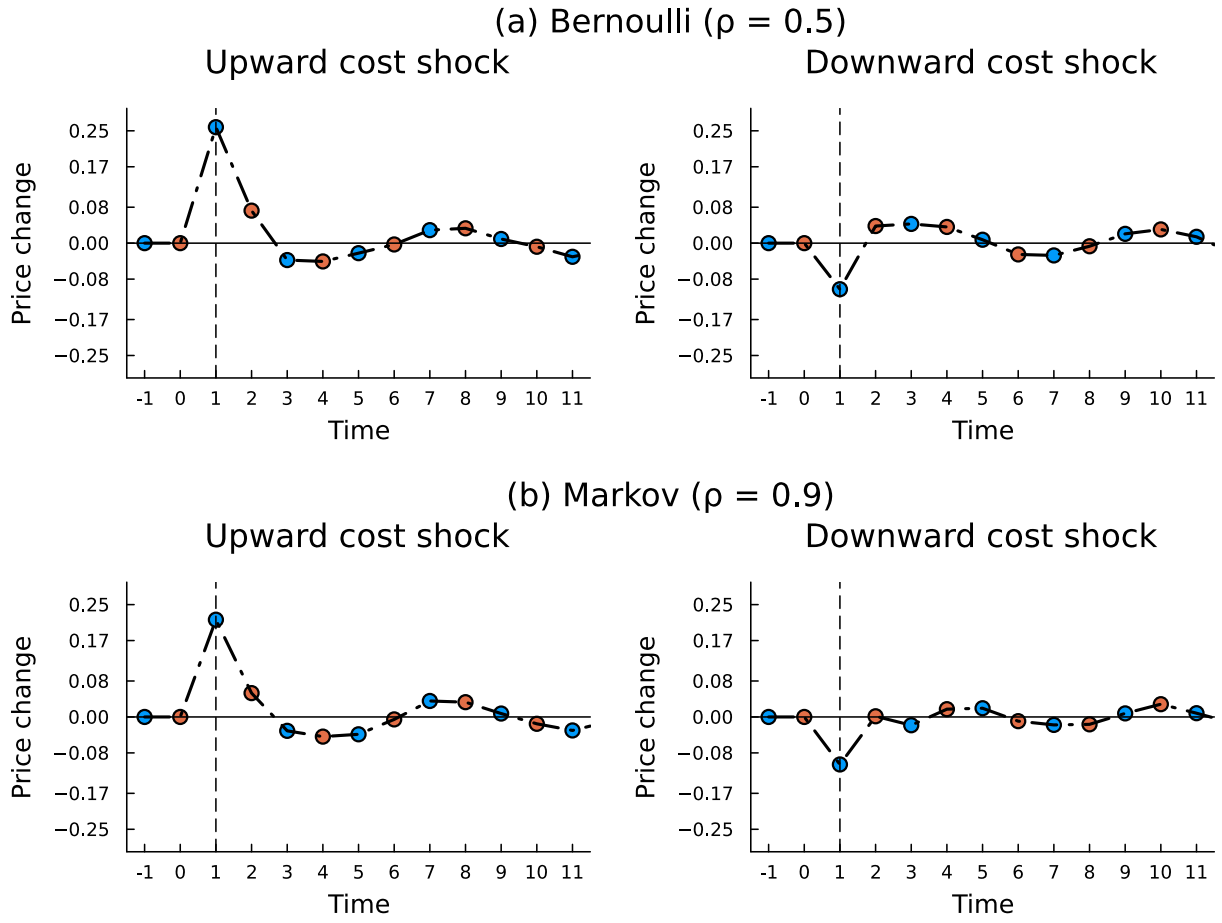
Notes: This table reports the response to a one-period unilateral price deviation. One algorithm is exogenously forced to deviate by choosing the most profitable undercutting price, while the other algorithm instead continues to play its learned strategy. Relative price change in period $\tau = 2$ is the percentage change in price between the deviation period and the following period. Average percentage gain from deviation is the relative difference between the deviating firm's average discounted profits along the deviation path and the counterfactual no-deviation profits. Length of punishment is the number of periods until both firms' prices return to their pre-deviation levels.

4.3 Asymmetric price responses to cost shocks

In this section, I study how algorithms respond to cost shocks, either upward or downward. The analysis is similar to that in the previous subsection. Beginning in period $\tau = 0$, from the prices that algorithms have converged in either the low or high-cost state, I simulate a cost shock (either upward or downward) occurring in period $\tau = 1$. The shock lasts for one period, and after that, the cost reverts to either its low or high state. Then, I compare post-shock prices to what prices would have been in the absence of a shock. Figure 3 plots the average difference between no-shock and post-shock prices. The left-hand panel presents the results for upward cost shocks, and the right-hand panel presents the results for downward cost shocks. Cost reductions lead to lower prices, and cost increases provoke price hikes.

Although costs vary by identical absolute amounts in both panels, upward cost shocks trigger a larger pass-through than downward cost shocks. This asymmetry occurs mainly because the vast majority of the sessions display price cycles. An upward cost shock can trigger a new relenting phase and, if it does, a significant price jump. In contrast, a downward cost shock will not cause a large price decrease, as it only allows more room for undercutting, and undercuts tend to be small (see Appendix B.2 for more details). The result is that cost increases can be passed through to prices more quickly than decreases (Chesnes, 2016; Noel,

Figure 3: Impulse response after one-period cost shock



Notes: Marginal cost changes in period $\tau = 1$. The shock lasts for one period only. The figure plots the average difference between no-shock and post-shock prices. The top two panels depict the results for $\rho = 0.5$. The bottom two panels depict the results of $\rho = 0.9$.

2009). We also observe that the price series tends to oscillate around zero. The oscillation occurs because prices are constantly changing along the cycle under both scenarios, so the pass-through fluctuates cyclically. Given this remark, the most critical period for observing the response of prices to cost shocks is period $\tau = 1$. See Appendix C.3 for more information on the distribution of the impulse responses.

4.4 The impact of cost uncertainty on collusion

The fact that the algorithms are still able to autonomously collude does not mean that cost uncertainty has no impact on collusion. To evaluate the impact of cost uncertainty on collusion, I consider a no-uncertainty benchmark. I conduct two numerical simulations by fixing the marginal cost at $c = c_L$ and $c = c_H$ separately and then taking the average. The normalized profit gain here is 58%, about 6% higher than in our baseline experiment. Thus cost uncertainty reduces profit gains in absolute terms.

5 Robustness

In this section, I briefly report how robust the baseline results are to changes in the economic environment. Appendix D provides more details.

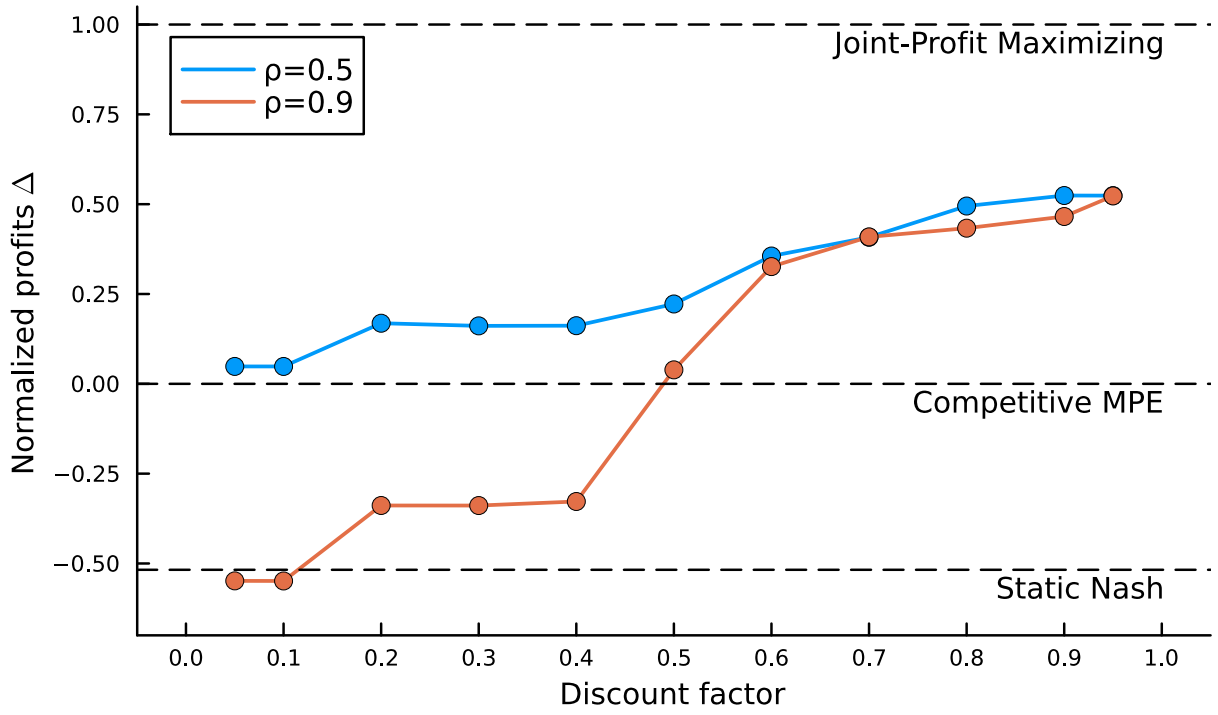
5.1 Discount factor

In the baseline calibration, I set the discount factor $\delta = 0.95$. In this subsection, I analyze how the results change as the discount factor takes lower values. Figure 4 summarizes how normalized profits vary. Consistent with theoretical literature and experimental evidence, lower discount factors make collusion difficult to sustain (Bruttel, 2009; Ivaldi et al., 2003). When the discount factor is close to zero, algorithms coordinate around the competitive outcome. Notably, algorithms consistently converge to the competitive MPE when the marginal cost follows a Bernoulli process (i.e., $\rho = 0.5$), but they coordinate on the static Nash equilibrium outcome under high correlation (i.e., $\rho = 0.9$).

5.2 Conditioning on own past price

In the baseline specification, algorithms condition their pricing decisions on the rival's price from the previous period, as well as the current and previous marginal costs. I examine the effects of allowing for conditioning on own past price. Relative to the baseline

Figure 4: Normalized profits as a function of the discount factor



Notes: The static Nash outcome is defined as a situation in which both algorithms set their prices at one increment above the marginal cost. The competitive MPE and joint-profit maximizing benchmark are defined as in Section 4.

specification, this results in a reduction of the profit gain. The profit gain is about 38.50% under the Bernoulli distribution and 44.20% under high persistence, respectively. This is due to the fact that the Q-matrix is much larger than in the baseline model. However, since β is held constant, the algorithms do not achieve the same level of learning. To do so, more experimentation would be required.

5.3 Higher uncertainty

In the baseline model, marginal costs can take only two values. In this subsection, I consider three cost levels (c_L, c_M, c_H) with transition matrix given by

$$\begin{bmatrix} \rho & \frac{1-\rho}{2} & \frac{1-\rho}{2} \\ \frac{1-\rho}{2} & \rho & \frac{1-\rho}{2} \\ \frac{1-\rho}{2} & \frac{1-\rho}{2} & \rho \end{bmatrix}. \tag{5}$$

As in the main analysis, I examine two relevant cases. In the high-persistence case, the persistence parameter is set $\rho = 0.9$, while in the Bernoulli case, it is $\rho = \frac{1}{3}$. The cost levels are defined as $(c_L, c_M, c_H)=\left(0, \frac{1}{6}, \frac{1}{3}\right)$. Again, these values are chosen such that the static Nash and monopoly prices are in the price space. I find that the normalized profit gained is 57%, which is a bit larger than in the baseline experiment. This confirms that Q-learning still yields supra-competitive profits under cost uncertainty.

5.4 Action set

I explore the consequences of enlarging the discrete price grid by setting $k = 24$, $k = 48$, and $k = 108$. Profit gains range from 60.82% to 10.84% when marginal costs follow a Bernoulli process, and from 53.31% to 31.90% when high persistence is introduced. In all cases, profits remain well above the competitive benchmark.

5.5 Large cost shocks

In the baseline setup, I set the high marginal cost $c_H = \frac{1}{6}$. In this subsection, I examine whether the magnitude of the cost shocks affects outcomes. Specifically, I increase the high-cost state to $c_H = \frac{1}{3}$ while keeping $c_L = 0$. I find that the normalized profit gained is around 60%.

6 Discussion

This paper presents a proof-of-concept demonstration that autonomous algorithmic collusion can be sustained in the presence of common and observable cost uncertainty. Specifically, in a simulation study using Q-learning pricing algorithms, I find that collusion is sustained by strategies that involve reward-punishment schemes. The behavior of the algorithms is remarkably consistent with the Markov perfect equilibria identified by Eckert (2004), Maskin and Tirole (1988), and Noel (2008). These findings contribute by addressing skepticism about the possibility of autonomous algorithmic collusion emerging in practice. Furthermore, by shedding light on when and how algorithms learn to collude, these findings can inform discussions on the development of legal frameworks aimed at protecting consumers.

This study is not without limitations. First, I have only considered self-trained Q-learning algorithms to facilitate comparisons with previous results in the algorithmic collusion literature. However, Q-learning has some practical limitations. Particularly, it requires a long and costly training period (Abada et al., 2024a; Frick, 2024). Such extensive learning might be acceptable in markets with very high-frequency price adjustments, but it seems impractical in other settings. Additionally, Q-learning requires a finite action and state space, which limits its applicability in environments with continuous decision variables or more complex state representations.

Second, I have focused on a single source of uncertainty, namely, stochastic marginal costs. Additionally, I have not considered firm-specific cost shocks. Extending the analysis to such shocks lies beyond the scope of the present paper because they fundamentally alter the information structure of the game and require a different equilibrium concept. Such shocks challenge the sustainability of collusive agreements since algorithms face an identification problem. That is, a price cut may be driven by exogenous shocks or a unilateral deviation from the implicit agreement. I leave extending the analysis to more diverse algorithms, other sources of uncertainty, and firm-specific shocks for future research.

References

Abada, I., Harrington Jr, J. E., Lambin, X., and Meylahn, J. M. (2024a). Algorithmic collusion: Where are we and where should we be going? Available at SSRN 4891033.

Abada, I. and Lambin, X. (2023). Artificial intelligence: Can seemingly collusive outcomes be avoided? Management Science, 69(9):5042–5065.

Abada, I., Lambin, X., and Tchakarov, N. (2024b). Collusion by mistake: Does algorithmic sophistication drive supra-competitive profits? European Journal of Operational Research.

Arulkumaran, K., Deisenroth, M. P., Brundage, M., and Bharath, A. A. (2017). A brief survey of deep reinforcement learning. arXiv preprint arXiv:1708.05866.

Arunachaleswaran, E. R., Collina, N., Kannan, S., Roth, A., and Ziani, J. (2024). Algorithmic collusion without threats. arXiv preprint arXiv:2409.03956.

Asker, J., Fershtman, C., and Pakes, A. (2021). Artificial intelligence and pricing: The impact of algorithm design. Technical report, National Bureau of Economic Research.

Asker, J., Fershtman, C., and Pakes, A. (2024). The impact of artificial intelligence design on pricing. Journal of Economics & Management Strategy, 33(2):276–304.

Assad, S., Calvano, E., Calzolari, G., Clark, R., Denicolò, V., Ershov, D., Johnson, J., Pastorello, S., Rhodes, A., Xu, L., et al. (2021). Autonomous algorithmic collusion: Economic research and policy implications. Oxford Review of Economic Policy, 37(3):459–478.

Assad, S., Clark, R., Ershov, D., and Xu, L. (2024). Algorithmic pricing and competition: Empirical evidence from the german retail gasoline market. Journal of Political Economy, 132(3):000–000.

Banchio, M. and Mantegazza, G. (2023). Adaptive algorithms and collusion via coupling. In EC, page 208.

Banchio, M. and Skrzypacz, A. (2022). Artificial intelligence and auction design. In Proceedings of the 23rd ACM Conference on Economics and Computation, pages 30–31.

Brown, Z. Y. and MacKay, A. (2021). Competition in pricing algorithms. Technical report, National Bureau of Economic Research.

Bruttel, L. V. (2009). The critical discount factor as a measure for cartel stability? Journal of Economics, 96(2):113–136.

Calvano, E., Calzolari, G., Denicolò, V., Harrington, J. E., and Pastorello, S. (2020a). Protecting consumers from collusive prices due to ai. Science, 370(6520):1040–1042.

Calvano, E., Calzolari, G., Denicolo, V., and Pastorello, S. (2020b). Artificial intelligence, algorithmic pricing, and collusion. American Economic Review, 110(10):3267–97.

Calvano, E., Calzolari, G., Denicolò, V., and Pastorello, S. (2021). Algorithmic collusion with imperfect monitoring. International Journal of Industrial Organization, page 102712.

Calvano, E., Calzolari, G., Denicolò, V., and Pastorello, S. (2023). Algorithmic collusion: Genuine or spurious? International Journal of Industrial Organization, 90:102973.

Calzolari, G. and Hanspach, P. (2024). Pricing algorithms out of the box: A study of the repricing industry. Available at SSRN 4871394.

Cartea, A., Chang, P., Penalva, J., and Waldon, H. (2022). The algorithmic learning equations: Evolving strategies in dynamic games. Available at SSRN 4175239.

Chen, L., Mislove, A., and Wilson, C. (2016). An empirical analysis of algorithmic pricing on amazon marketplace. In Proceedings of the 25th international conference on World Wide Web, pages 1339–1349.

Chesnes, M. (2016). Asymmetric pass-through in us gasoline prices. The Energy Journal, 37(1):153–180.

Cho, I.-K. and Williams, N. (2024). Collusive outcomes without collusion: Algorithmic pricing in a duopoly model. Available at SSRN 4753617.

den Boer, A. V., Meylahn, J. M., and Schinkel, M. P. (2022). Artificial collusion: Examining supracompetitive pricing by q-learning algorithms. Amsterdam Law School Research Paper, (2022-25).

Deng, A. (2018). What do we know about algorithmic tacit collusion. Antitrust, 33:88.

Descamps, A., Klein, T., and Shier, G. (2021). Algorithms and competition: The latest theory and evidence. Competition Law Journal, 20(1):32–39.

Doraszelski, U. and Pakes, A. (2007). A framework for applied dynamic analysis in IO. Handbook of industrial organization, 3:1887–1966.

Dorner, F. E. (2021). Algorithmic collusion: A critical review. arXiv preprint arXiv:2110.04740.

Dou, W. W., Goldstein, I., and Ji, Y. (2024). Ai-powered trading, algorithmic collusion, and price efficiency. Jacobs Levy Equity Management Center for Quantitative Financial Research Paper.

Eckert, A. (2004). An alternating-move price-setting duopoly model with stochastic costs. International Journal of Industrial Organization, 22(7):997–1015.

Eckert, A. (2013). Empirical studies of gasoline retailing: A guide to the literature. Journal of economic surveys, 27(1):140–166.

Edelman, B. and Ostrovsky, M. (2007). Strategic bidder behavior in sponsored search auctions. Decision support systems, 43(1):192–198.

Epivent, A. and Lambin, X. (2024). On algorithmic collusion and reward–punishment schemes. Economics Letters, 237:111661.

Ericson, R. and Pakes, A. (1995). Markov-perfect industry dynamics: A framework for empirical work. The Review of economic studies, 62(1):53–82.

Eschenbaum, N., Mellgren, F., and Zahn, P. (2022). Robust algorithmic collusion. arXiv preprint arXiv:2201.00345.

Ezrachi, A. and Stucke, M. E. (2017). Artificial intelligence & collusion: When computers inhibit competition. U. Ill. L. Rev., page 1775.

Farias, V., Saure, D., and Weintraub, G. Y. (2012). An approximate dynamic programming approach to solving dynamic oligopoly models. The RAND Journal of Economics, 43(2):253–282.

Fish, S., Gonczarowski, Y. A., and Shorrer, R. I. (2024). Algorithmic collusion by large language models. arXiv preprint arXiv:2404.00806.

Frick, K. M. (2024). Convergence to collusion in algorithmic pricing. Available at SSRN 4527452.

Hansen, K. T., Misra, K., and Pai, M. M. (2021). Frontiers: Algorithmic collusion: Supra-competitive prices via independent algorithms. Marketing Science, 40(1):1–12.

Harrington, J. E. (2018). Developing competition law for collusion by autonomous artificial agents. Journal of Competition Law & Economics, 14(3):331–363.

Hartline, J. D., Long, S., and Zhang, C. (2024). Regulation of algorithmic collusion. In Proceedings of the Symposium on Computer Science and Law, pages 98–108.

Hettich, M. (2021). Algorithmic collusion: Insights from deep learning. Available at SSRN 3785966.

Ivaldi, M., Jullien, B., Rey, P., Seabright, P., and Tirole, J. (2003). The economics of tacit collusion. Technical report, Report for DG Competition, European Commission.

Johnson, J. P., Rhodes, A., and Wildenbeest, M. (2023). Platform design when sellers use pricing algorithms. Econometrica, 91(5):1841–1879.

Kimbrough, S. O. and Murphy, F. H. (2009). Learning to collude tacitly on production levels by oligopolistic agents. Computational Economics, 33:47–78.

Klein, T. (2021). Autonomous algorithmic collusion: Q-learning under sequential pricing. RAND Journal of Economics, pages 1–21.

Kühn, K.-U. and Tadelis, S. (2018). The economics of algorithmic pricing: Is collusion really inevitable. Unpublished Manuscript.

Lamba, R. and Zhuk, S. (2022). Pricing with algorithms. arXiv preprint arXiv:2205.04661.

Lambin, X. (2024). Less than meets the eye: Simultaneous experiments as a source of algorithmic seeming collusion. Available at SSRN 4498926.

Leisten, M. (2024). Algorithmic competition, with humans. Available at SSRN 4733318.

Martin, S. and Rasch, A. (2024). Demand forecasting, signal precision, and collusion with hidden actions. International Journal of Industrial Organization, 92:103036.

Maskin, E. and Tirole, J. (1988). A theory of dynamic oligopoly II: Price competition, kinked demand curves, and Edgeworth cycles. Econometrica: Journal of the Econometric Society, pages 571–599.

Maskin, E. and Tirole, J. (2001). Markov perfect equilibrium: I. Observable actions. Journal of Economic Theory, 100(2):191–219.

Miklós-Thal, J. and Tucker, C. (2019). Collusion by algorithm: Does better demand prediction facilitate coordination between sellers? Management Science, 65(4):1552–1561.

Musolff, L. (2022). Algorithmic pricing facilitates tacit collusion: Evidence from e-commerce. In Proceedings of the 23rd ACM Conference on Economics and Computation, pages 32–33.

Noel, M. (2009). Do retail gasoline prices respond asymmetrically to cost shocks? The influence of Edgeworth cycles. The RAND Journal of Economics, 40(3):582–595.

Noel, M. D. (2008). Edgeworth price cycles and focal prices: Computational dynamic Markov equilibria. Journal of Economics & Management Strategy, 17(2):345–377.

OECD (2017). Algorithms and collusion: Competition policy in the digital age. OECD.

O'Connor, J. and Wilson, N. E. (2021). Reduced demand uncertainty and the sustainability of collusion: How AI could affect competition. Information Economics and Policy, 54:100882.

Pakes, A. and McGuire, P. (2001). Stochastic algorithms, symmetric Markov perfect equilibrium, and the 'curse'of dimensionality. Econometrica, 69(5):1261–1281.

Possnig, C. (2023). Reinforcement learning and collusion. Department of Economics, University of Waterloo.

Salcedo, B. (2015). Pricing algorithms and tacit collusion. Manuscript, Pennsylvania State University.

Stokey, N., Lucas, R., and Prescott, E. (1989). Recursive methods in economic dynamics. Harvard University Press.

Sutton, R. S. and Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.

Waltman, L. and Kaymak, U. (2008). Q-learning agents in a cournot oligopoly model. Journal of Economic Dynamics and Control, 32(10):3275–3293.

Watkins, C. and Dayan, P. (1992). Q-learning. Machine learning, 8(3-4):279–292.

Werner, T. (2026). Algorithmic and human collusion. The Economic Journal.

Xie, M. and Chen, J. (2004). Studies on horizontal competition among homogenous retailers through agent-based simulation. Journal of Systems Science and Systems Engineering, 13(4):490–505.

Appendix

A Q-Learning

The objective of the algorithm i is to maximize

$$\mathbb{E}\left[\sum_{t=0}^{\infty}\delta^t\pi_i(p_{it},p_{jt},c_t)\right] \qquad \text{s.t} \qquad \begin{array}{l} s_{it+1}=g(p_{it},s_{it}) \\ s_0 \ \text{ given.} \end{array} \tag{A.1}$$

I assume that the algorithms move alternately. Only in odd-numbered periods t, algorithm 1 observes its state $s_{1t} \in S$ and then chooses its price $p_{1t} \in P$. Similarly, only in even-numbered periods, algorithm 2 observes its state $s_{2t} \in S$ and then chooses its price $p_{2t} \in P$. Given the state and the action chosen, the algorithm i gets an immediate profit π_i that depends on the algorithm's own action p_{it} and on the state s_{it}.[15] The future state is determined by the function g, which is the state transition function. Under the one-period memory assumption, the state for player i is a tuple $s_{it}=(p_{jt-1},c_{t-1},c_t)$. A finite memory is necessary for the state space to be finite, given that the action space is finite as well. We can rewrite the sequential problem (A.1) as a recursive problem by using Bellman's principle of optimality

$$V_i(s_{it})=\max_p\Big\{\pi_i(p,s_{it})+\mathbb{E}\left[\delta\pi_i(p,s_{it+1})+\delta^2V_i\left(s_{it+1}\right)\mid p,s_{it}\right]\Big\}. \tag{A.2}$$

The solution of (A.2) is given by the policy function $h_i : S \to P$, which is defined as

$$h_i(s_{it})=\operatorname*{argmax}_p\Big\{\pi_i(p,s_{it})+\mathbb{E}\left[\delta\pi_i(p,s_{it+1})+\delta^2V_i\left(s_{it+1}\right)\mid p,s_{it}\right]\Big\}. \tag{A.3}$$

Assuming that the agent has perfect knowledge of the immediate reward function π_i and the state transition function g, the recursive problem (A.2) can be solved using standard

[15] It is assumed that the profit function π_i is bounded.

dynamic programming techniques (i.e., the value function iteration method).[16] However, in this paper, it is assumed that π_i and g are unknown. Thus, instead of using standard dynamic programming methods, our algorithms use an alternative method called Q-learning.

For our purposes, define the function $Q_i(p_{it}, s_{it})$, which gives the maximum discounted cumulative reward that can be achieved starting from state s_{it} and applying the action p_{it} as the first action.[17] Because the action space is discrete, Q_i is a $|P| \times |S|$ matrix. The value of the cell $Q_i(p_{it}, s_{it})$ is the reward received immediately upon executing action p_{it} from state s_{it}, plus the value of following the optimal policy thereafter:

$$Q_i(p_{it}, s_{it}) \equiv \pi_i(p_{it}, s_{it}) + \mathbb{E}\left[\delta \pi_i(p_{it}, s_{it+1}) + \delta^2 V_i(s_{it+1}) \mid p_{it}, s_{it}\right]. \tag{A.4}$$

By taking the maximum of both sides and using equation (A.2) yields

$$\max_p Q_i(p, s_{it}) = V_i(s_{it}). \tag{A.5}$$

Therefore, we can rewrite the equation (A.3) in terms of Q as

$$h_i(s_{it}) = \operatorname*{argmax}_p Q_i(p, s_{it}). \tag{A.6}$$

It shows that if the agent learns the Q_i function instead of the V_i function, it will be able to select optimal actions even when it has no knowledge of the functions π_i and g. As equation (A.6) makes clear, the algorithm needs only consider each available action p_{it} in its current state s_{it} and choose the action that maximizes $Q_i(p_{it}, s_{it})$. Hence, if the agent knew the Q-matrix, it could then easily calculate the optimal action for any given state.

[16] The perfect knowledge assumption implies that the agents can predict in advance the exact outcome of setting a certain price in any state. See Stokey et al. (1989) for further details.

[17] The term Q-function derives from the fact that the Q-value can be thought of as an index of the "Quality" of price p in state s.

Learning. Note that equation (A.5) allows rewriting equation (A.4) as

$$Q_i(p_{it}, s_{it}) = \pi_i(p_{it}, s_{it}) + \mathbb{E}\left[\delta\pi_i(p_{it}, s_{it+1}) + \delta^2 \max_p Q_i\left(p, s_{it+1}\right) \mid p_{it}, s_{it}\right]. \tag{A.7}$$

This recursive definition of Q_i provides the basis for estimating the Q-matrix by an iterative approximation procedure. Starting from an arbitrary initial matrix Q_0, after choosing action p_{it} in state s_{it}, the algorithm observes π_i and s_{it+1} and updates the corresponding cell of the matrix $Q_{it}(p_{it}, s_{it})$ according to the following equation:

$$Q_{it+1}(p_{it}, s_{it}) = (1-\alpha)Q_{it}(p_{it}, s_{it}) + \alpha\left[\pi_i(p_{it}, s_{it}) + \delta\pi_i(p_{it}, s_{it+1}) + \delta^2 \max_{p_{it+1}} Q_{it}(p_{it+1}, s_{it+1})\right], \tag{A.8}$$

where $\alpha \in [0, 1]$ is a learning parameter that regulates how quickly new information replaces old information. When α is equal to 0, then there is no learning process since the agent does not take into account the new information acquired. Analogously, when α is equal to 1, the algorithm would immediately forget what it has learned in the past.

According to equation (A.8), the new estimate of the optimal long-run value given state s_{it} consists of three components: direct profit $\pi_i(p_{it}, s_{it})$, next period profit $\pi_i(p_{it}, s_{it+1})$ when new state s_{it+1} realizes but the price has not changed (discounted for one period), and the highest possible Q-value $\max_{p_{it+1}} Q_{it}(p_{it+1}, s_{it+1})$ in this new state s_{it+1} (discounted for two periods). This enables an iterative approximation in which initially the Q-values are imprecise, but over time, they become better estimates of the long-run consequences of choosing p_{it} in state s_{it}. It is worth noticing that Q-learning is slow because it updates only one cell of the Q-matrix at a time, and approximating the true matrix generally requires that each cell be visited many times.[18] The larger the state or action space, the more iterations will be needed.

[18] Another learning protocol involves updating the estimates of all Q-values simultaneously rather than one at a time. Asker et al. (2021) refer to these algorithms as *synchronous*. However, synchronous Q-learning algorithms need to be provided with information about the underlying economic environment, which allows them to compute counterfactual (Calvano et al., 2023).

The action selection. As in Calvano et al. (2020b) and Klein (2021), I use a procedure called ε-greedy exploration, with probability ε_t it chooses a price randomly at period t and with probability $1 - \varepsilon_t$ it chooses the currently optimal action (i.e., the one with the highest Q-value in the relevant state). During the exploration phase, the algorithm explores the state-action space in order to discover the payoffs associated with a certain action in a given state. Following Calvano et al. (2020b), the exploration rate is defined as $\varepsilon_t = \exp(-\beta t)$ with $\beta > 0$. This implies that initially, the algorithm chooses an action randomly, but then the probability of selecting the greedy choice increases as time goes by. In case of ties, the algorithm randomizes over all currently optimal actions.

Convergence. In a single-agent setting Q-learning converges to the optimal strategy under certain conditions (Watkins and Dayan, 1992). Intuitively, a sufficient condition for such convergence is that each action-state pair should be visited infinitely often. However, in a multi-agent setting, when two or more algorithms interact repeatedly with each other, the problem becomes non-stationary and history-dependent and there is no theoretical result that guarantees ex-ante that the agents will learn an optimal policy.[19] However, this does not imply that Q-learning is expected to behave badly. It simply means that theory is not able to say how well Q-learning is expected to work. In my simulations, I used the same criterion as in Calvano et al. (2020b) to verify convergence. In each session, convergence is achieved if for each agent i and each state s, the price $p_{it} = \text{argmax}_p\big[\hat{Q}_{it}(p, s)\big]$ remains constant for 100,000 consecutive periods.

[19]Some recent papers provide approaches based on stochastic approximation results to characterize the convergence of reinforcement learning algorithms to study the emergence of autonomous algorithmic collusion (Cartea et al., 2022; Banchio and Mantegazza, 2023; Cho and Williams, 2024; Possnig, 2023).

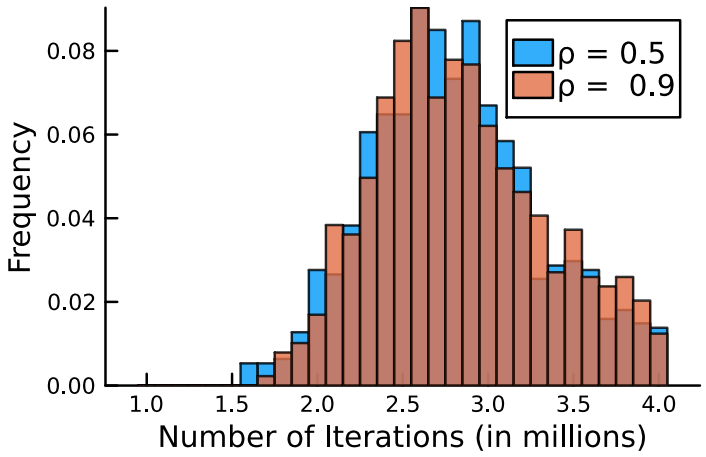
B Outcomes

This Appendix contains additional material for Section 4.

B.1 Convergence

Figure Figure B.1 shows the number of iterations required to achieve convergence, as defined in Section 4. Convergence typically takes millions of iterations.

Figure B.1: Time to converge

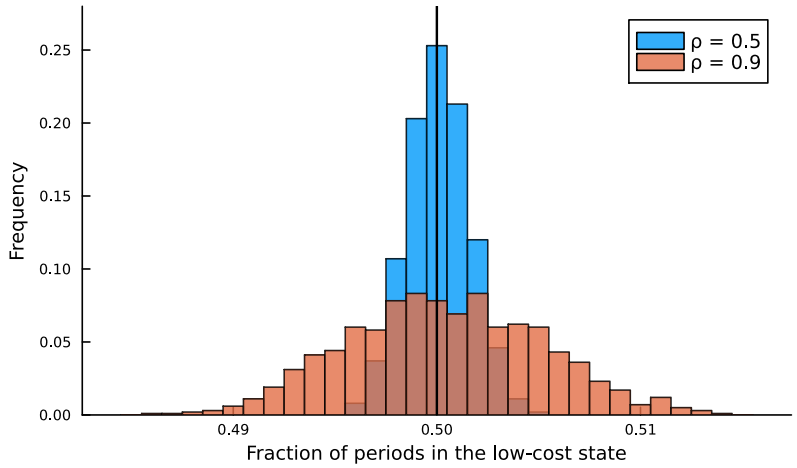


Notes: The figure shows a histogram of the number of iterations until algorithms converge to a limit strategy for different values of the persistence parameter.

B.2 Cost distribution, pricing dynamics, and cycles

Figure B.2 depicts the empirical distribution of cost states upon convergence. Consistent with theoretical predictions, it is centered at 0.50 for any value of the persistence parameter.

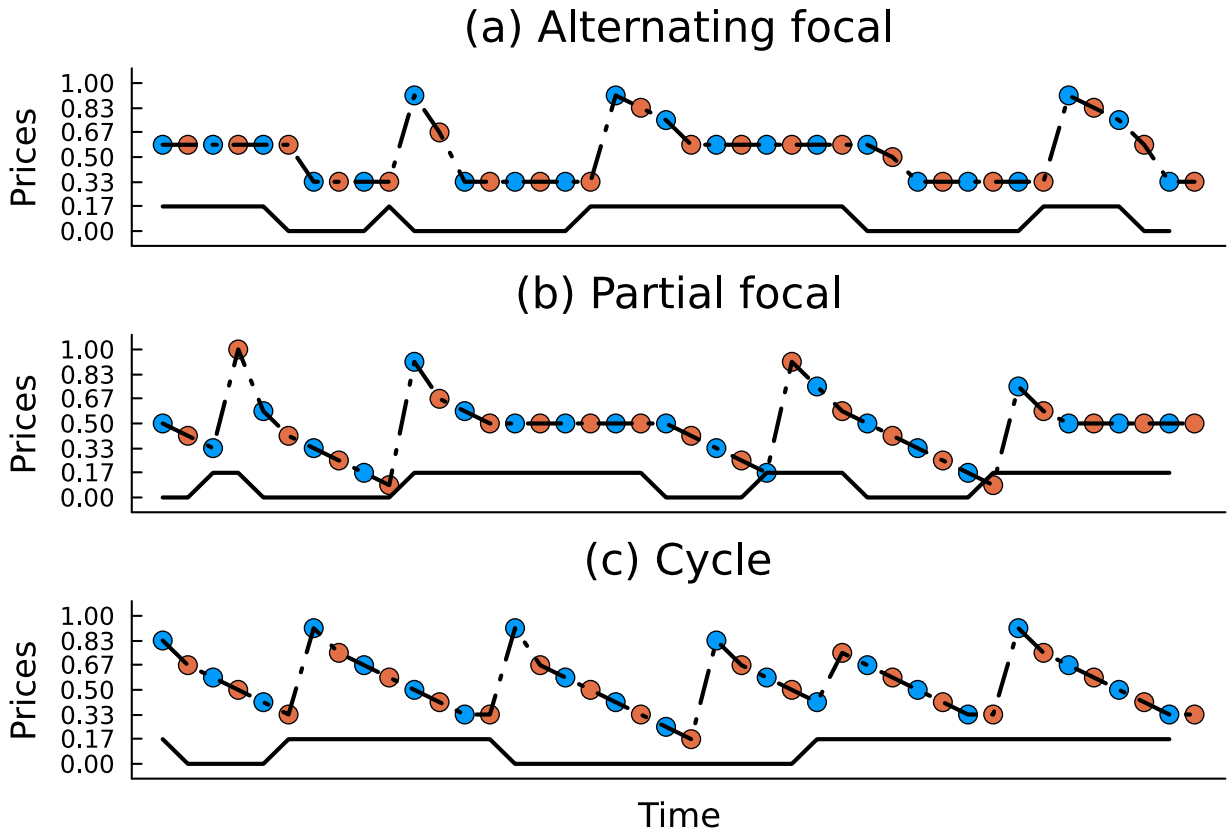
Figure B.2: Empirical distribution of cost state



Notes: This figure shows the fraction of time the marginal cost spends in the low state upon convergence for different values of the persistence parameter ρ. The solid line represents the stationary distribution.

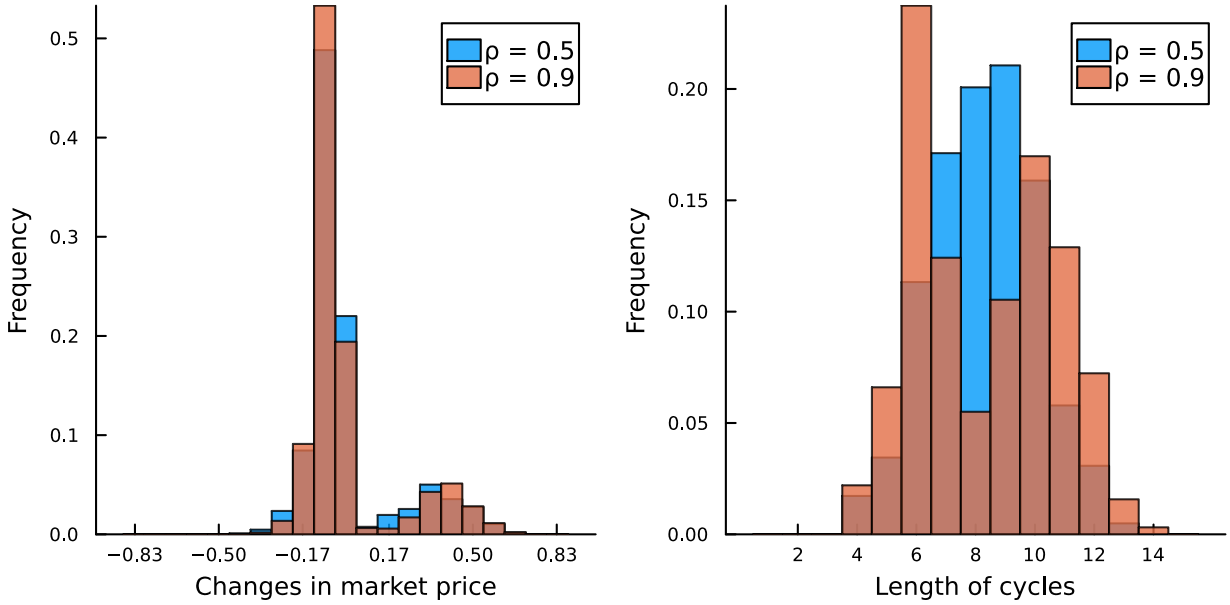
Figure B.3 illustrates the three different pricing patterns observed upon convergence. It depicts the prices charged by algorithms during the final 40 periods of individual sessions that have converged to different patterns. One notable difference between the price cycles observed in panels (b) and (c) and those developed in Maskin and Tirole (1988) is that our algorithms display deterministic cycles. That is, it is always the same algorithm that resets the cycle. This difference comes from the fact that Q-learning plays a pure strategy by design. Figure B.4 reports more information about the price cycles. The left panel shows that price decreases occur much more often but are smaller in magnitude than price increases. The right panel illustrates the length of the cycles.

Figure B.3: Pricing dynamics



Notes: Prices observed upon convergence for three different pricing patterns. The solid black lines represent marginal costs. Just for the sake of illustration, the figure depicts the pricing dynamics for $\rho = 0.9$. A similar pattern is observed when marginal cost follows a Bernoulli process.

Figure B.4: Distribution of changes in market price and length of cycles



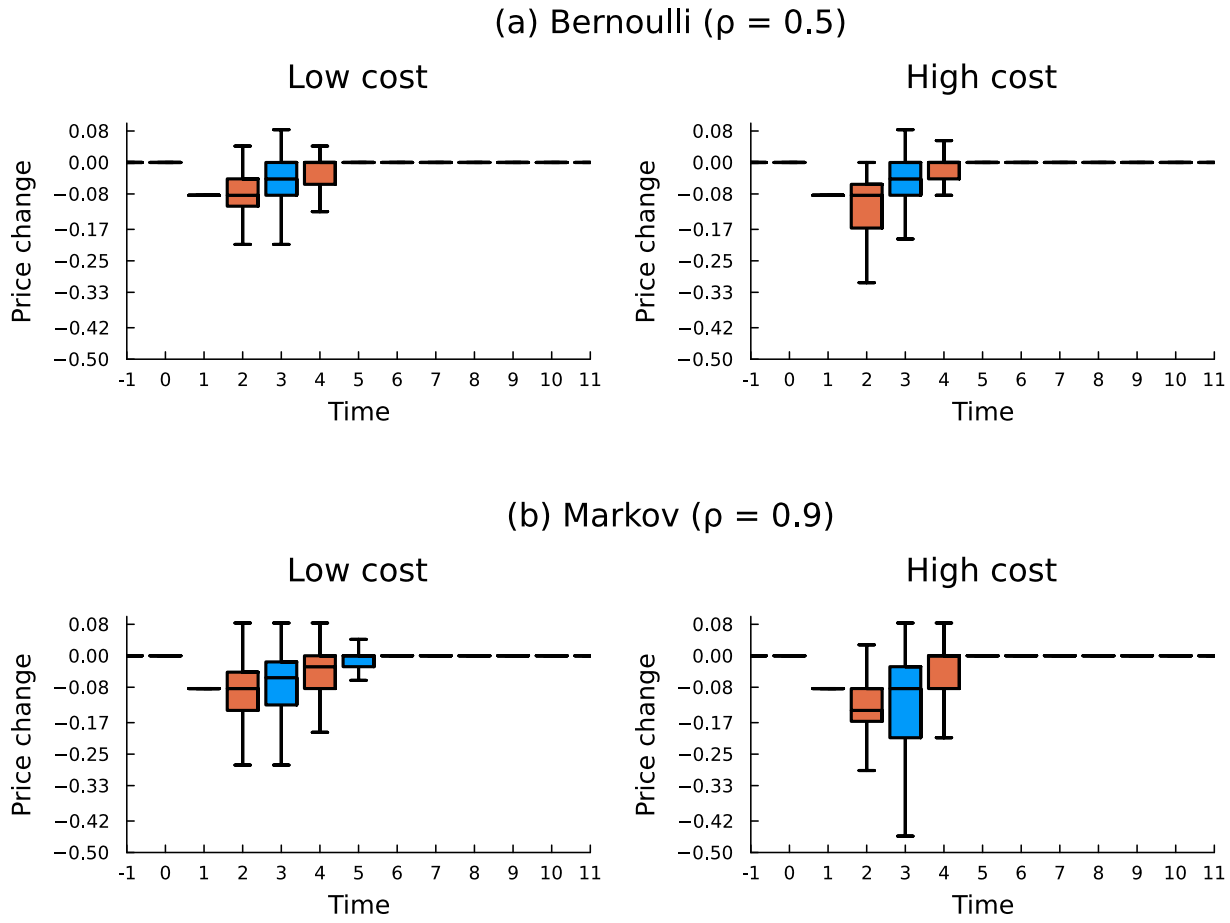
Notes: The left panel illustrates all the changes in market price that occur during the final 1,000 periods for those sessions that converge to a cycle. The right panel depicts the length of those price cycles. Results are reported for different values of the persistence parameter.

C Deviations and Punishment

C.1 Downward price deviation

Figure C.1 reports more information on the distribution of impulse-response functions after a unilateral downward price deviation.

Figure C.1: Impulse response by cost state

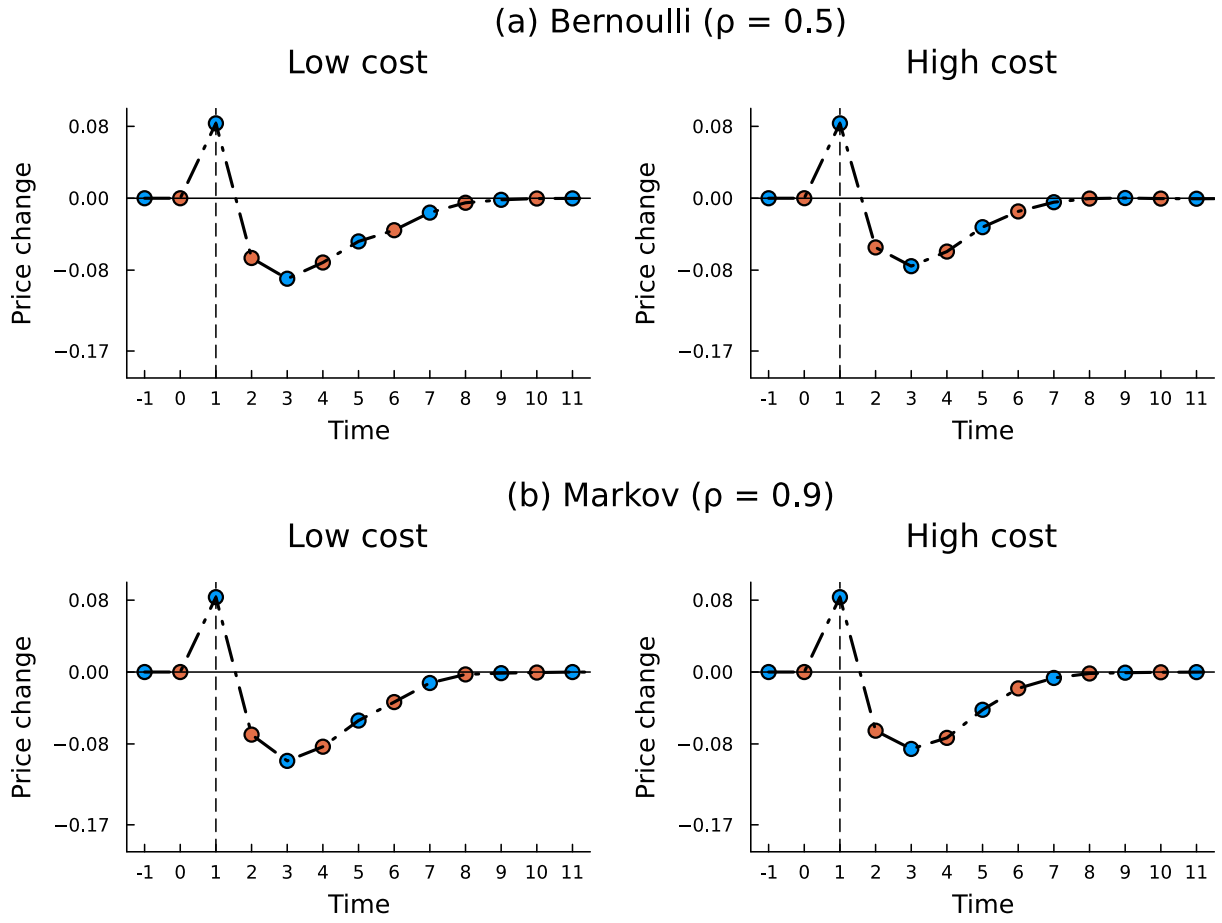


Notes: In each panel, the solid black lines represent the median value, the boxes represent the 25th and 75th percentiles, and the solid intervals represent the ranges of the price charged after an exogenous price reduction in period $\tau = 1$. or sessions leading to a price cycle, I consider deviations starting from every point of the cycle and take the average of all of them. The variable on the vertical axis is the difference between the current and the long-run price. The top two panels depict the results for $\rho = 0.5$. The bottom two panels depict the results of $\rho = 0.9$.

C.2 Upward price deviation

In line with Epivent and Lambin (2024), Figure C.2 shows that price hikes are systematically followed by price wars.

Figure C.2: Price hikes are followed by price wars

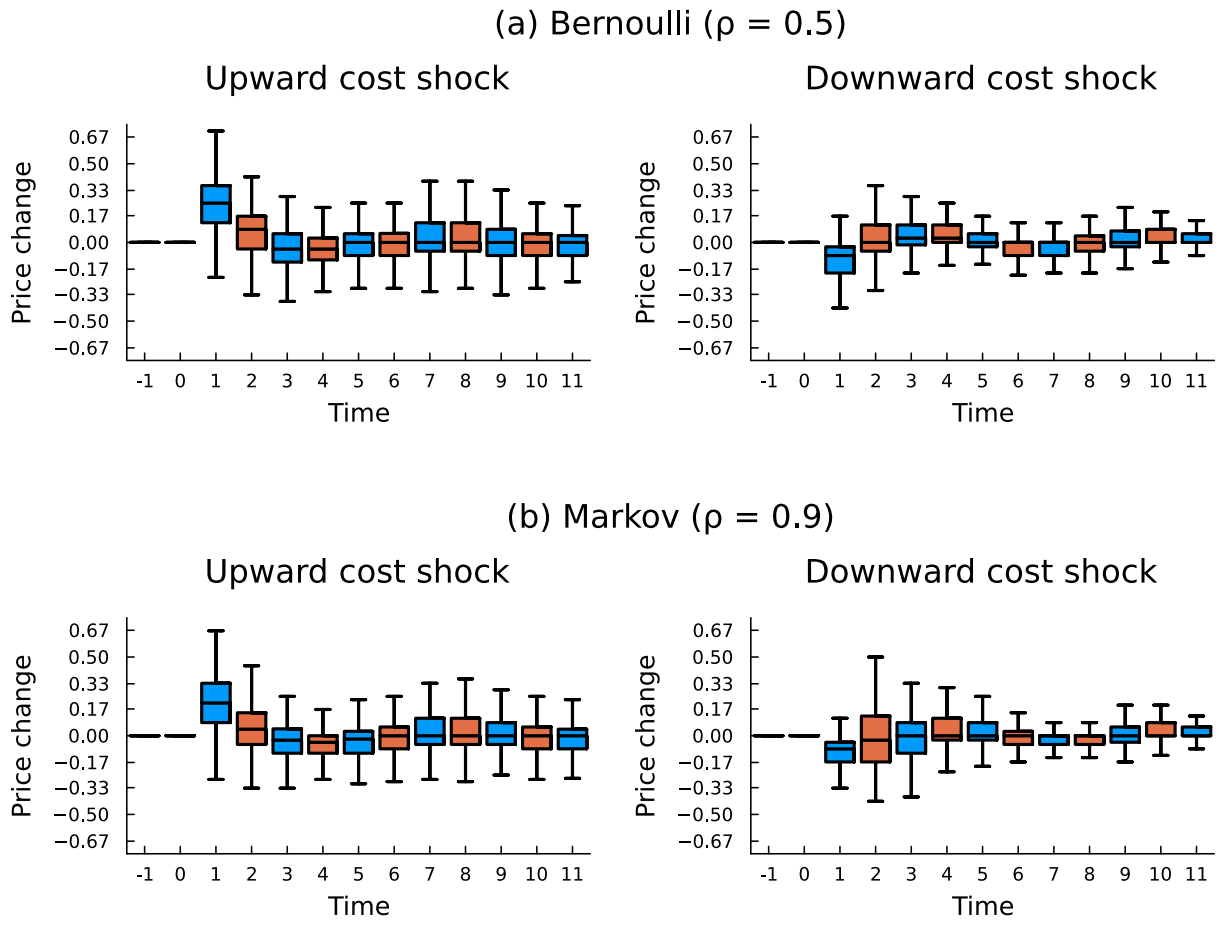


Notes: One algorithm is forced to deviate in period $\tau = 1$. The upward deviation lasts for one period only. To avoid confounding effects, the marginal cost is fixed. The figure plots the average price. The variable on the vertical axis is the difference between the current and the long-run price. The top two panels depict the results for $\rho = 0.5$. The bottom two panels depict the results of $\rho = 0.9$.

C.3 Asymmetric price responses to cost shocks

Figure C.3 shows the distribution of the price responses to cost shocks. Upward cost shocks have a large effect on prices, while downward cost shocks have a small effect in the period $\tau = 1$.

Figure C.3: Impulse response after one-period cost shock



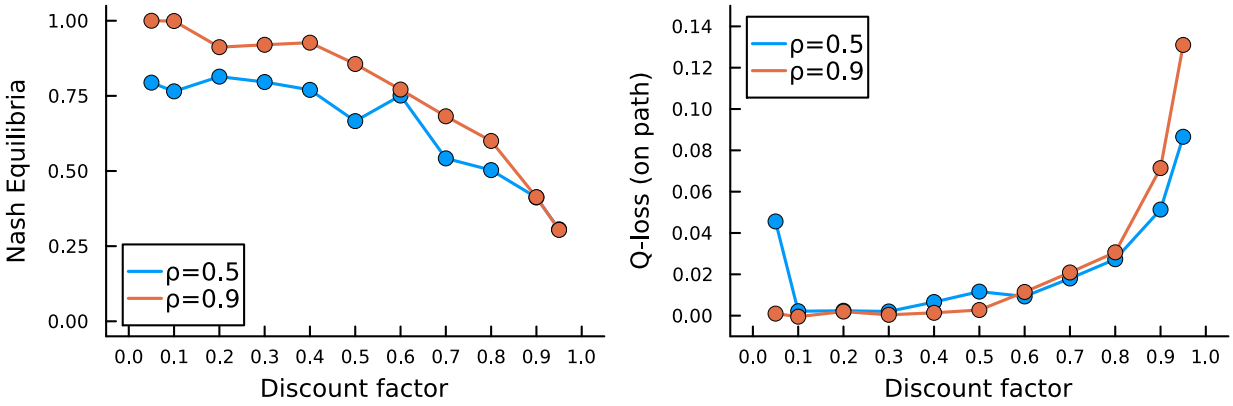
Notes: Marginal cost changes in period $\tau = 1$. The shock lasts for one period only. In each panel, the solid black lines represent the median value, the boxes represent the 25th and 75th percentiles, and the solid intervals represent the ranges of the price charged after an exogenous price reduction in period $\tau = 1$. The variable on the vertical axis is the difference between the current and the long-run price. The top two panels depict the results for $\rho = 0.5$. The bottom two panels depict the results of $\rho = 0.9$.

D Robustness

D.1 Discount factor

Figure D.1 plots the frequency of Nash equilibria and the average Q-loss (on path) as a function of the discount factor for different values of the persistence parameter. In line with Klein (2021), the share of Nash equilibria decreases, and the average Q-loss (on path) increases as the discount factor increases.

Figure D.1: Frequency of Nash equilibria and average Q-loss (on path)



D.2 Conditioning on own past price

Table D.1 reports the results allowing for conditioning on own past prices. The Q-matrix is substantially larger than in the baseline case. As learning is more limited, the profit gain decreases, along with the equilibrium play on path.

Table D.1: Conditioning on own past price

	Bernoulli ($\rho = 0.5$)	Markov ($\rho = 0.9$)
Average number of iterations until convergence (in millions)	3.267	4.021
Average market price	0.342	0.357
Average normalized profit	0.385	0.442
Standard deviation normalized profits	0.103	0.114
Frequency of Nash equilibria	0.180	0.172
Average Q-loss (on path)	0.177	0.243
Standard deviation Q-loss (on path)	0.053	0.082
Average Q-loss (all states)	0.558	0.647
Standard deviation Q-loss (all states)	0.030	0.035

D.3 Higher uncertainty

Table D.2 reports the results of experiments with three cost states. The average profit is a bit larger than in the baseline experiment, but algorithms have increasing difficulty in learning strategies that are the best responses to its competitor.

Table D.2: Higher uncertainty

	Bernoulli ($\rho = 0.5$)	Markov ($\rho = 0.9$)
Average number of iterations until convergence (in millions)	4.273	4.093
Average market price	0.455	0.451
Average normalized profit	0.579	0.570
Standard deviation normalized profits	0.043	0.079
Frequency of Nash equilibria	0.140	0.125
Average Q-loss (on path)	0.075	0.177
Standard deviation Q-loss (on path)	0.035	0.084
Average Q-loss (all states)	0.063	0.223
Standard deviation Q-loss (all states)	0.024	0.046

Notes: For comparison, normalized profit gained under random pricing is -0.572

D.4 Action set

Table D.3 reports the results when the action is enlarged by setting $k = 24$, $k = 48$, and $k = 108$. As the action set is enlarged, average profits decline yet remain well above the competitive benchmark, the frequency of Nash equilibria decreases, Q-losses (both on path and over all states) increase, and the number of iterations required for convergence falls. This reflects the fact that enlarging the action set substantially increases the size of the Q-matrix, but the experimentation parameter β is held constant. As a result, each state–action cell is visited less frequently, and Q-values are updated less often, leading to earlier stabilization and limited learning. Achieving the same level of learning as in the baseline specification would require more experimentation.

Table D.3: Action set

	Bernoulli ($\rho = 0.5$)			Markov ($\rho = 0.9$)		
	$k = 24$	$k = 48$	$k = 108$	$k = 24$	$k = 48$	$k = 108$
Number of iterations until convergence (in millions)	3.099	2.690	0.288	3.482	3.449	2.298
Average market price	0.437	0.390	0.322	0.410	0.408	0.339
Average normalized profit	0.608	0.542	0.106	0.533	0.568	0.319
Standard deviation normalized profits	0.061	0.067	0.020	0.081	0.064	0.068
Frequency of Nash equilibria	0.154	0.052	0.001	0.139	0.034	0.002
Average Q-loss (on path)	0.103	0.183	0.611	0.167	0.252	0.469
Standard deviation Q-loss (on path)	0.043	0.040	0.141	0.078	0.081	0.051
Average Q-loss (all states)	0.131	0.469	0.933	0.297	0.594	0.894
Standard deviation Q-loss (all states)	0.046	0.026	0.005	0.070	0.041	0.009

Notes: For comparison, the normalized profit gain under random pricing is -0.282 for $k = 24$, -0.237 for $k = 48$, and -0.213 for $k = 108$.

D.5 Large cost shocks

Table D.4 reports the results when the high marginal cost is set equal to $c_H = \frac{1}{3}$. The average market price and normalized profits are a bit larger than in the baseline experiment.

Table D.4: Large cost shocks

	Bernoulli ($\rho = 0.5$)	Markov ($\rho = 0.9$)
Number of iterations until convergence (in millions)	3.058	3.156
Average market price	0.453	0.457
Average normalized profit	0.598	0.606
Standard deviation normalized profits	0.072	0.105
Frequency of Nash equilibria	0.232	0.211
Average Q-loss (on path)	0.112	0.174
Standard deviation Q-loss (on path)	0.055	0.097
Average Q-loss (all states)	0.150	0.262
Standard deviation Q-loss (all states)	0.058	0.102

Notes: For comparison, normalized profit gained under random pricing is -0.557