Contents

1 Introduction

In 2023, the financial sector's spending on AI worldwide has rapidly grown, amounting to 35.03 billion USD globally, with the European banking sector spending 5.37 billion USD [9]. Recently, the finance sector has found itself at the forefront of AI integration, exhibiting one of the highest adoption rates across industries.

Additionally, pricing software is widely employed by vendors selling on online marketplaces, including airline companies and retail platforms. Over a third of the major online vendors have automated their pricing since 2015. This automation allows vendors to dynamically adjust prices based on factors such as demand, competitor pricing, and inventory levels in real-time [3]. However, there have been problems with the use of these algorithms. Ryanair's ticket pricing algorithm has drawn attention due to the Italian government's intervention to prevent market manipulation and speculation. This dynamic pricing algorithm adjusts ticket prices based on real-time demand, availability, and booking time to maximise profits and aircraft capacity. The practice of overbooking ("bumping"), where airlines sell more tickets than available seats, betting on some passengers not showing up, has been problematic, often resulting in customer dissatisfaction. To address these issues, the Italian government has taken steps to regulate and oversee these pricing practices to ensure fairer market conditions [7].

Another problematic example of airlines' usage of pricing algorithms is the dramatic increase in flight ticket prices to Sicily, with fares surging up to 300 percent during the holiday season. This issue has prompted various responses, including organising bus services, protests, and petitions for a Sicilian airline. The root cause is attributed to pricing algorithms used by airline companies, which potentially engage in collusive behaviour to inflate prices. The Sicilian regional president has called for an antitrust investigation, which noted unusual price alignments possibly due to these algorithms. Although there is a new budget allocation to subsidise flights, its implementation remains unclear, and alternatives like establishing a new airline are being considered [4].

Due to the complex nature of these algorithms, it is difficult to understand exactly how these algorithms arrive at their decisions and, therefore, regulate them. Markets characterised

by oligopolies are at risk of having algorithms learn to charge supracompetitive prices without communicating with one another. This leads us to the topics of collusion and algorithmic pricing, which are both highly relevant given the growing usage of these algorithms. In the literature that will be explored, collusion is defined as "a reward-punishment scheme designed to provide the incentives for firms to consistently price above the competitive level" [2]. Collusion presents significant legal and ethical challenges, necessitating new regulatory frameworks to ensure markets remain fair and competitive.

This thesis aims to explore an area of literature initiated in 2020 by a paper called "Artificial Intelligence, Algorithmic Pricing, and Collusion" [2], which serves as the starting point for understanding the relationship between reinforcement learning algorithms and market outcomes. Emilio Calvano received an ERC (European Research Council) Advanced Grant of 1.9 million for this research, further highlighting the importance of this research [1]. Building upon Calvano's work, subsequent research, such as Eschenbaum's analysis in 2022 [5], will be explored, providing deeper insights into the robustness of algorithmic pricing strategies and the factors influencing collusion. Eschenbaum replicates the Calvano experiments and tests the algorithms in different contexts, delving into the problem of overfitting. Moreover, the Lambin paper in 2023 [8] introduces a novel perspective on algorithmic pricing, challenging the notion of collusion as an intentional strategy and aims to justify why it is not collusion by describing the mechanisms used by the algorithms to learn.

Furthermore, this thesis aims to explore the specific case of memoryless agents: whether they are able to reach collusive outcomes and re-adapt to new environments with different parameters. Robustness is an important property to check due to the dynamic nature of real-world markets. If an agent is not able to quickly adapt to changes in an environment, collusive outcomes identified in static environments do not apply to real-world practical applications. Additionally, considering a memoryless state space, as the literature review will reveal, allows for high levels of collusion to be attained whilst being less computationally expensive to implement. An experiment investigating the adaptability of memoryless agents to new environments, particularly focusing on the transfer of knowledge from low-cost to high-cost contexts in repeated games, will be conducted. By training algorithms in different cost

environments and retraining them in contrasting settings, the experiment aims to reveal how an agent adjusts its strategies in a new environment. The research question at the core of this experiment is: "Can memoryless agents maintain collusion levels when transferred from a low-cost to a high-cost environment, and if so, how do the convergence times and collusion indices change?".

2 Reinforcement learning theory

2.1 The Agent–Environment Interface

We first introduce the notion of an agent-environment interface (AEI) [10]. In an AEI we have an agent and an environment. An agent can select actions in a given state, receive a reward at the next time step and transition to a new state. Formally, the Agent at each step t receives a representation of the environment's *state*, $S_t \in S$ and it selects an action $A_t \in A(s)$. Then, as a consequence of its action the agent receives a *reward*, $R_{t+1} \in R \subseteq \mathbb{R}$. [10]

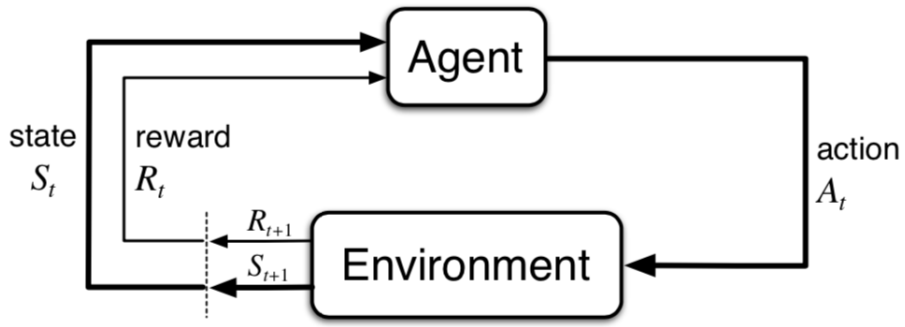


Figure 1: Plot illustrating how an AEI operates. Specifically, depicting how an agent interacts with its environment at each time step. [10]

2.2 Policy

At each given time, an agent has to choose an action. A policy is a probability distribution over the possible actions conditional on a given given state. [10] Formally, a *policy* is a mapping from a state to a distribution of actions: $\pi_t(a|s)$ That is the probability of select an action $A_t = a$ if $S_t = s$. For all the analysis considered in the paper we will only be considering deterministic polices. That is a state maps to a single action with probability 1.

2.3 Reward

The total *reward* is expressed as: $G_t = \sum_{k=0}^{H} \gamma^k r_{t+k+1}$ Where γ is the *discount factor*. Analogous to infinitely repeated games, a small γ implies that an agent is myopic and values rewards in the present significantly more than those in the future. H is the *horizon*, that can

be infinite.

2.4 Markov Decision Process

A **Markov Decision Process**, MDP, is a 5-tuple (S, A, P, R, γ)

$$
\begin{aligned}
&\text{finite set of states:} \\
&s \in S \\
&\text{finite set of actions:} \\
&a \in A \\
&\text{state transition probabilities:} \\
&p(s'|s,a) = Pr\{S_{t+1} = s' | S_t = s, A_t = a\} \\
&\text{expected reward for state-action-nexstate:} \\
&r(s', s, a) = \mathbb{E}[R_{t+1} | S_{t+1} = s', S_t = s, A_t = a]
\end{aligned}
\tag{1}
$$

2.5 Value Function

The value function describes the expected reward given that the agent starts from a given state, under a policy π. Formally: $V_\pi(s) = \mathbb{E}[G_t | S_t = s]$ Moreover we also have the optimal value function, which returns the maximum rewards under the optimal policy, $V_*(s) = \max_\pi V_\pi(s)$

2.6 Action-Value (Q) Function

We can also denoted the expected reward for state, action pairs. $q_\pi(s,a) = \mathbb{E}_\pi \left[G_t | S_t = s, A_t = a \right]$
The optimal value-action function: $q_*(s,a) = \max_\pi q^\pi(s,a)$

2.7 Bellman Equation

An important recursive property emerges for both Value (2.5) and Q (2.6) functions if we expand them.

$$
V_\pi(s) = \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) \left[r + \gamma V_\pi(s') \right] \tag{2}
$$

Similarly, we can do the same for the Q function:

$$q_\pi(s,a) = \sum_{s',r} p(s',r|s,a)\left[r + \gamma V_\pi(s')\right] \tag{3}$$

2.8 Temporal Difference - Q Learning

Temporal Difference (TD) methods assume the dynamics of the system (transition probabilities) are not known. The idea is to obtain samples of trajectories after running some policy through a Markov Decision Process. These type of methods are called model free. Q learning is a type of TD method. Below is the psuedo-code for the algorithm. The procedure outlines the following main steps.

1. Sampling an initial state

2. Sampling a new state + reward

3. Updating the Q-matrix

Algorithm 1: Q Learning

Initialise $Q(s,a)$ arbitrarily and $Q(terminal - state,) = 0$

foreach *episode* $\in$ *episodes* **do**

  while s *is not terminal* **do**

    Choose a from s using policy derived from Q (e.g., ϵ-greedy)

    Take action a, observer r, s'

    $Q(s,a) \leftarrow Q(s,a) + \alpha\left[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right]$

    $s \leftarrow s'$

  end

end

The ϵ-greedy policy selects an action uniformly with probability ϵ, otherwise it selects the $\arg\max_a Q(s,a)$. Introducing this method helps us find the right exploration-exploitation trade off. To discover optimal policies it is necessary to explore all state-actions pairs. However to get high returns we must exploit known high value pairs.

3 Literature Review

In this section, the main literature surrounding the Calvano paper in 2020 [2] to the present research in the field will be covered. This includes extensions of the Calvano paper and critiques of the paper.

3.1 The experiment

The majority of papers consider a game-theoretic environment as a standard repeated Bertrand setting [5]. The setting is as follows: a Bertrand game with $n = 2$ firms is considered. Firms choose prices as actions in each period $t \in \{0, \dots, T\}$. We let $p_{i,t}$ and $p_{j,t}$ for $i, j \in \{1, 2\}, i \neq j$ denote the period-t price of each firm and $q_{i,t}$, $q_{j,t}$ the quantities. The demand function is given by the logit demand function [5]:

$$q_{i,t} = \frac{e^{\frac{\gamma_i - p_{i,t}}{\mu}}}{\sum_{i=1}^{n} e^{\frac{\gamma_i - p_{i,t}}{\mu}} + e^{\frac{\gamma_0}{\mu}}} \tag{4}$$

The profit at the end of each period is given by:

$$\pi_{i,t} = (p_{i,t} - c_i) q_{i,t} \tag{5}$$

The action space for each firm reduces to choosing a price. Formally, the action set is reduced to the following: $[\underline{p}_N - \xi(\bar{p}_C - p_N), \bar{p}_C + \xi(\bar{p}_C - \underline{p}_N)]$, letting m denote the size of this set. This interval is computed by first finding the Bertrand-Nash equilibrium of the one-shot-game and the monopoly prices (i.e., those that maximise aggregate profits). These are denoted by $\underline{p}_N$ and $\bar{p}_C$ respectively [2]. Now we consider how the state space is represented. Since we are studying reinforcement learning in the context of repeated games, we would like the state space to represent the history of both firms' actions. However, Q-learning algorithms assume that the size of the state space is fixed in time. To overcome this limitation, the paper introduces the concept of bounded memory. Formally, we impose a k-bounded memory: $s_k = \{p_{t-1}, \dots, p_{t-k}\}$. When studying the memoryless case, the state space will be the empty

set, meaning the agent only receives a reward from the environment and not a new state. Next, the following epsilon-greedy model is used: $e^{-\beta t}$ with $\beta > 0$. A larger β implies that the agent explores less. The last part of the experimental setup is the assessment of the collusion of algorithms. Collusion is defined as "a reward-punishment scheme designed to provide the incentives for firms to consistently price above the competitive level" [2]. To quantify the collusion, we define the average profit gain:

$$\Delta = \frac{\bar{\pi} - \pi^N}{\pi^C - \pi^N} \tag{6}$$

where $\bar{\pi}$ is the average per-firm profit upon convergence. It is computed by considering the average profit over the time interval where the Q-matrix stabilises. π^C represents the profit under full collusion and π^N the outcome under the Bertrand-Nash static equilibrium. $\Delta = 0$ implies a competitive outcome. Conversely, $\Delta = 1$ implies perfect collusion [2].

3.2 Results: Calvano (2020) and Eschenbaum (2022)

In this section, I will discuss the various results obtained from several papers. Starting with the Calvano (2020) [2] paper, they initially conduct the experiment using the following baseline economic parameters (a symmetric duopoly) ($n = 2$) with $c_i = 1$, $\gamma_i - c_i = 1$, $\gamma_0 = 0$, $\mu = \frac{1}{4}$, $\delta = 0.95$, $m = 15$, $\xi = 0.1$, and a one-period memory ($k = 1$). To perform the experiment, they initialise two Q-learning agents and let them interact (they are interacting in this setting because the demand is a function of the prices both firms set). Then, they study the strategies of the sessions that converged (for every firm, the optimal strategy doesn't change for 100,000 consecutive periods). It is important to note that the optimal strategy can be represented as a mapping from $f : A \times A \to A$ for $k = 1$ memory. The main conclusion of the experiment is that trained algorithms consistently set higher prices than competitive levels, leading to supra-competitive levels. These elevated prices persist even when algorithms play optimally and near-optimally. The paper makes two more observations from the experiment. Firstly, algorithms that converge to a limit strategy may not always achieve optimal responses when faced with a rival's strategy. Secondly, equilibrium play is not always achieved, particularly

when exploration is limited. However even when equilibrium play is not achieved, the algorithms typically do not deviate significantly from optimal responses.

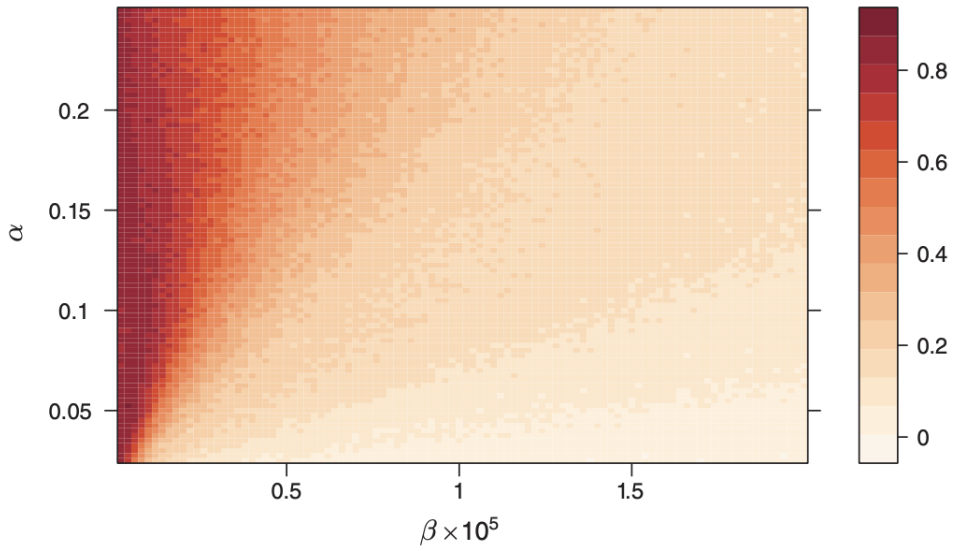


Figure 2: Fraction of sessions that converge to a Nash Equilibrium as a function of the parameters α and β. The parameter α represents the learning rate and β represents the exploration rate, The color intensity indicating the fraction of converging sessions.[2]

Figure 2 illustrates that with the right amount of exploration, collusive outcomes are attainable; low values of β lead to more prevalent equilibrium play. Across the grid, Δ varies from 70% to 90%. The profit gain appears to be less sensitive to changes in the learning and experimentation parameters. It tends to be highest when both α and β are low, indicating extensive exploration and persistent learning. However, reducing either α or β too much can lead to negative consequences. The largest attainable Δ is slightly lower than 1. [2]

It is important to dissect the main part of the paper: understanding why firms do not cut their prices when their prices exceed the Nash-Bertrand level. The results reveal that upon convergence, behaviour either ends up in cycles or in Nash equilibrium (of the repeated game). The cycles refer to a recurring pattern in the pricing behaviour. The most interesting part of the section is the discussion on deviations and punishments. The paper, in an attempt to demonstrate that the algorithms are colluding, notes that supra-competitive prices do not necessarily imply collusion (both firms may arbitrarily adopt a strategy where they always set high prices regardless of their profits). Figure 3 illustrates how following a price cut from one

agent, the other one follows, and they gradually readjust their prices to the collusive ones.

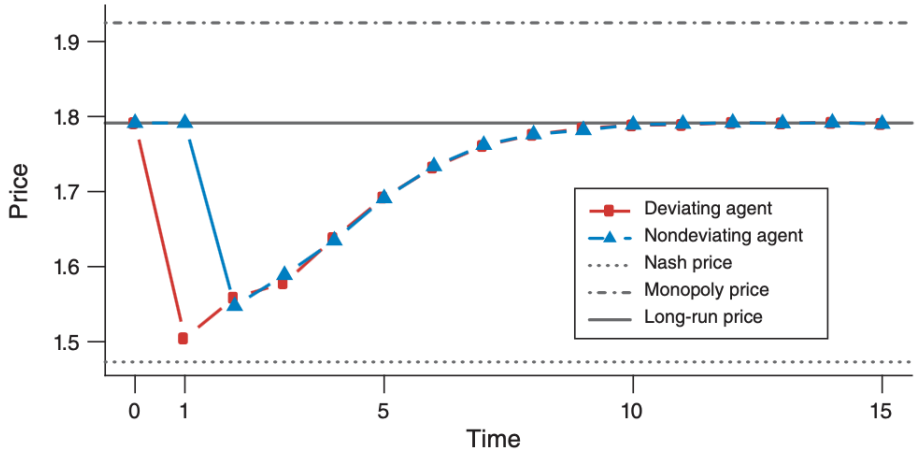
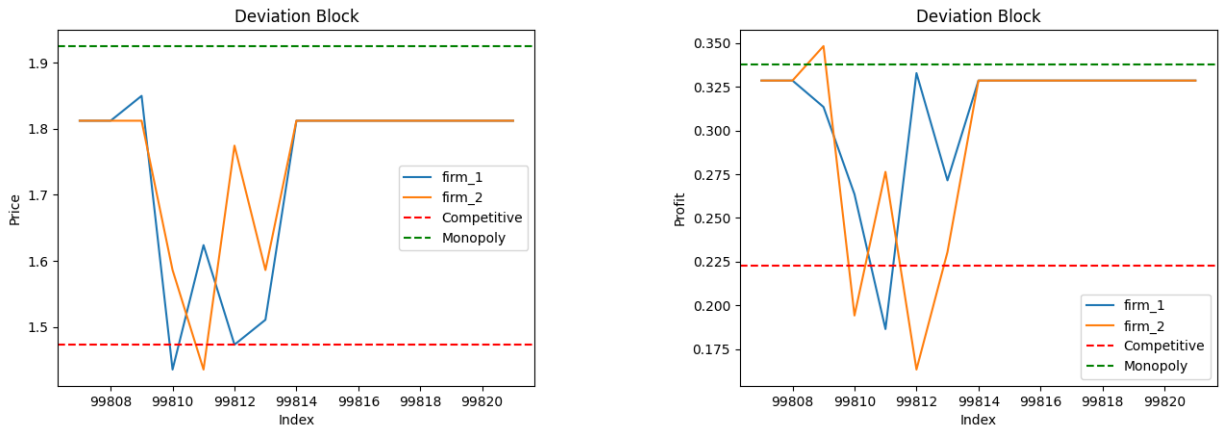


FIGURE 4

Figure 3: Price Dynamics Following a Forced Deviation from Either Agent. Specifically an illustration of the average price charged over an interval following a forced deviation from either agent. [2]

It is calculated by considering the price charged over an interval following a forced deviation from either agent. However, this plot is an average across multiple sessions and doesn't entirely capture the types of deviations that are occurring. The paper notes that this behaviour is not in line with grim-trigger strategies, which involve maintaining a cooperative stance and punishing deviations indefinitely. Instead, they exhibit a pattern of temporary punishment followed by a return to normal behaviour. As shown in the graph, there is a pattern of returning to pre-deviation prices after 5–7 periods. The paper attributes this behaviour to the fact that the algorithms have memory because it gives them the ability to penalise deviations in maintaining cooperative behaviour and preventing individual agents from pursuing self-interested strategies that could undermine collective outcomes. To show the limitations of this plot, I will consider the average behaviour when both agents deviate shown in Figure 4: this was computed by considering the 100,000 iterations for a given session where the Q-matrix has converged. Within these last iterations, because of the epsilon-greedy model, there are some forced simultaneous deviations.

The two illustrations show the behaviour of the agents after a deviation. For firm one, they have the following trajectory: $S_0 = (1.811967, 1.811967), A_0 = 1.849639, R_1 = 0.328551, S_1 =$



(a) Price Deviation

(b) Profit Deviation

Figure 4: Comparison of agent behavior following a deviation. (a) Fluctuations in agent prices after a deviation. (b) Profit deviation illustrates the fluctuations in agent rewards after a deviation

$(1.849639, 1.811967), A_1 = 1.435255, R_1 = 0.313445, \ldots$ The following observations can be made. Firstly, the first deviation results in a lower reward for firm 1 and a higher reward for firm 2. This is then followed by a sharp drop in rewards for both firms after they lower their prices to competitive levels. The rewards and prices continue to oscillate around the competitive levels and then jump back up to the pre-deviation levels. This is not consistent with the graph presented in the paper which shows a gradual return to pre-deviation prices.

The Eschenbaum [5] paper further analyses this experiment. Their goal is to confirm the robustness of the findings from the Calvano paper. They also introduce the problem of over-fitting the training environment, i.e., that when placed in a new environment (changing some parameters of the algorithm being faced), it is no longer able to collude. When replicating the Calvano experiment with the same parameters, they observe the high values of Δ. However, they note that changing the marginal costs for the firms results in different levels of collusion. In a low-cost environment, players benefit from randomised play, achieving profits above the Nash equilibrium. This implies that they yield higher returns on average. Conversely, in a high-cost context, players experience losses from randomising compared to playing Nash equilibria, resulting in lower collusion. The discussion then transitions to the main discussion of the paper: testing the algorithms in a new context. This entails training algorithms separately in

different contexts and then evaluating their performance in the contexts they were not trained. To clarify, consider the following example. Suppose we have two different contexts k and k'. We first let two algorithms learn in each context respectively. These algorithms will all learn a policy which we denote as: $b_1^k(\cdot)$, $b_2^k(\cdot)$, $b_1^{k'}(\cdot)$, $b_2^{k'}(\cdot)$. The performance of each player in their own context is measured in terms of Δ and outside of their context. We formalise this with $M_i(b_k(\cdot), b_k(\cdot), G_k)$ and $M_i(b_k(\cdot), b_{k'}(\cdot), G_{k'})$ where G_k encodes the parameters for a specific context. By comparing the performance in different contexts, one can evaluate how well the algorithms can extrapolate policies learned [5]. Surprisingly, in the new context, collusion breaks down entirely as shown in Figure 5. This implies that performance during training is no indication of the outcome in the market.

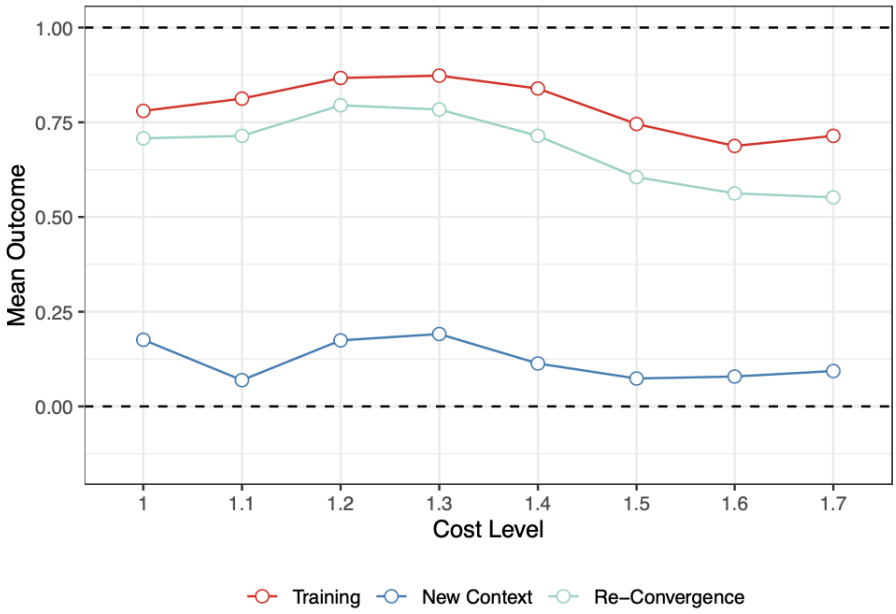


Figure 5: Comparison of collusion indices between trained and new testing contexts across different scenarios: (red line) convergence in training contexts, (blue line) initial 1000 iterations in new testing contexts, and (light blue line) convergence in new contexts. Variables are defined as follows: M_i denotes the collusion index, $b_k(\cdot)$ and $b_{k'}(\cdot)$ are behavior strategies, and G_k represents the context. [5].

Specifically, Figure 5 illustrates how collusion behaviour changes over time, transitioning from high collusion during training to a breakdown of collusion in the new context, leading to Nash equilibrium behaviour on average (of the stage game). The re-convergence line represents the outcomes after allowing the algorithms to converge again in the new environment. The paper notes that changes to any parameters, even just changing the initial seed used to

initialise the learning process breaks down collusion.

Both the Calvano and Eschenbaum papers further try to characterise the strategy profile by considering the transitive closure of strategy pairs (analysing the limiting strategies and the sequence of transitions between states which lead to stable/unstable nodes). To illustrate the closure, they adopt a directed graph to represent the combined behaviour of the interacting algorithms.

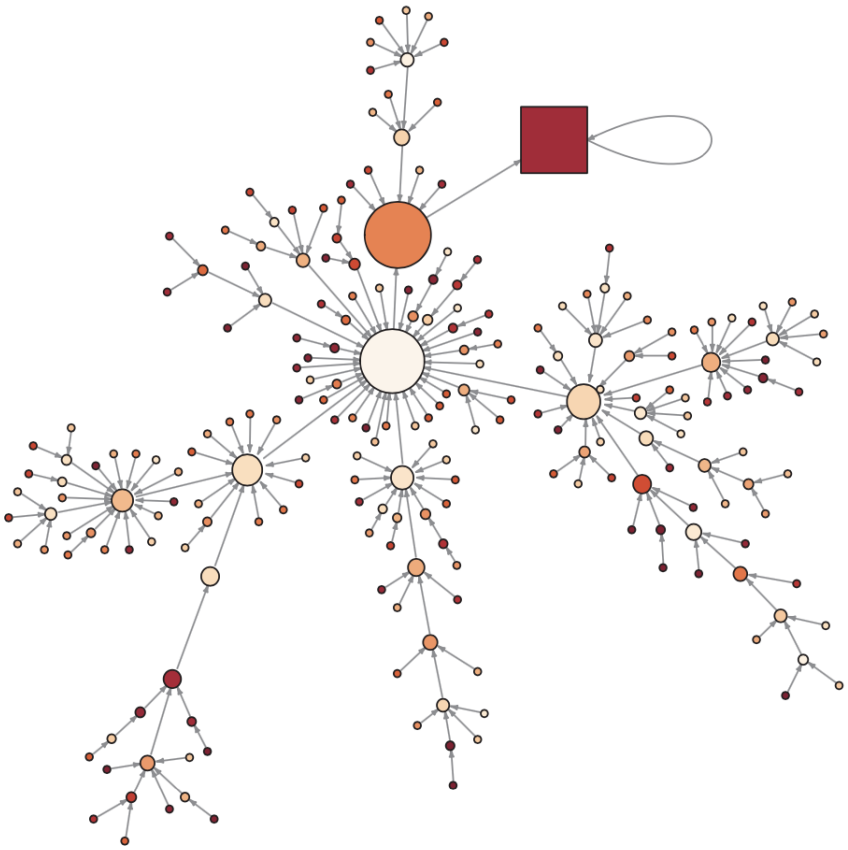


Figure 6: Fully connected directed graph of strategies. [2]

Variable	**Description**
Node	Represents a state defined by the prices of Firm 1 and Firm 2
Horizontal Axis	Price charged by Firm 1
Vertical Axis	Price charged by Firm 2
Brightness	Profit gain at each state
Edge	Possible transition between states
Square Node	Absorbing state, indicating long-run prices or stable equilibrium

Table 1: Description of variables in the fully connected directed graph of strategies.

The Calvano paper makes the following comments. Firstly, all the nodes lead to the absorbing node. This implies that if the agents start in an arbitrary state and continually choose prices according to their optimal policy, they will end up in the absorbing state. The figure provides evidence for the deviation-punishment behaviour. Starting from any node other than the absorbing one, the system initially moves toward the low part of the main diagonal and then climbs up to the long-run prices. This suggests that cooperation does not restart immediately but only after a punishment phase [2]. However, the Eschenbaum paper reveals slightly different results. When they consider an instance of algorithms converging to a high symmetric price, they observe that the graph is not fully connected. This is because depending on the action selected, the strategies do not lead back to the collusive outcome [5] as shown in Figure 7.

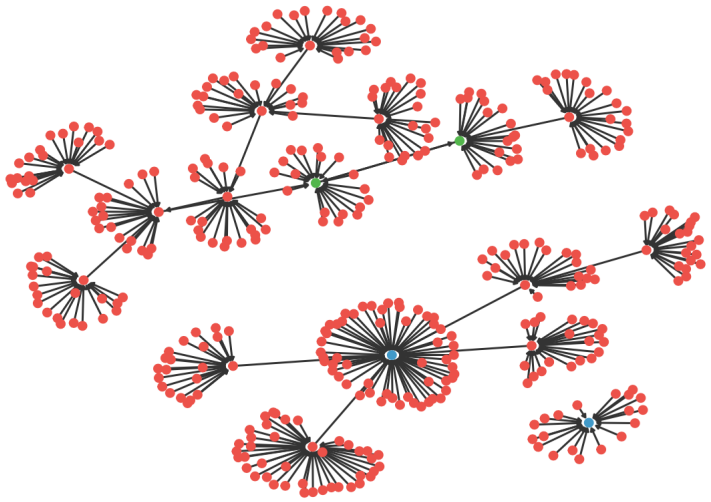


Figure 7: Disconnected graph of strategies illustrating the transitive closure of two pairs of policies with identical outcome in training contexts evaluated in testing contexts. [5]

Variable	**Description**
Node	Strategy profile representing a state defined by the prices of Firm 1 and Firm 2
Edge	Transition between states occurring with the specific strategy-pair
Blue Node	Stable end-node where the action-pair is played repeatedly forever
Green Node	Unstable end-node forming a cycle among action-pairs played forever once reached

Table 2: Description of variables in the disconnected graph of strategies.

3.3 Results: Lambin (2023)

The next paper to be discussed is the Lambin (2023) [8]. Unlike the Calvano and Eschenbaum paper, the Lambin paper refutes the idea that the algorithms reach supra-competitive prices because of the deviation-punishment schemes. Instead, they argue that high prices result directly from simultaneous experimentation and the learning inertia inherent in most reinforcement learning techniques. Specifically, how initial action valuations are based on simultaneous experimentation by multiple agents. This can lead to persistent erroneous valuations, causing agents to fail in identifying profitable strategies and potentially converge to prices significantly higher than Nash equilibrium when the exploration rate decreases or agents learn slowly. Unlike the Calvano and Eschenbaum papers, they support their hypothesis with some theoretical results. To construct their argument, they first consider a Q-learning approach with a 2-steps exploration policy, where agents alternate between exploration and exploitation phases based on the ϵ-greedy strategy. Initially, agents explore with $\epsilon_t = 1$, then transition to exploitation with $\epsilon_t = 0$ after a period T_1. Without loss of generality, the Q-matrices are defined considering a stateless agent-environment interface. We slightly reformulate the notation for the update procedure:

$$Q(a) \leftarrow Q(a) + \alpha \left[\tilde{r}(a) + \gamma \max_{a'} Q(a') - Q(a) \right]$$

where $\tilde{r}(a_i) = \mathbb{E}[R_{t+1}|S_t = s, A_t = a, B_t = b]$, and B_t is a random action selected by the other agent, and $\tilde{r} = \mathbb{E}[R_{t+1}|S_t = s, B_t = b]$. Before introducing Lemma 1 [8], it is important to keep in mind that when using Q-learning methods, which are an adjustment of Monte Carlo evaluation methods, we are initially approximating expectations with averages in hopes that they converge to the true expectations. The lemma states the following:

Lemma 1. *When* $\epsilon_t = \epsilon_1 = 1$, $\mathrm{Q}^{\mathrm{t_i}}(a_i) \xrightarrow{T_1 \to \infty} \tilde{r}(a_i) + \frac{\delta}{1-\delta}\tilde{r}$

The provided interpretation suggests that within a system, agents eventually reach a stage where each agent selects a unique action with the highest Q-value, denoted as $a_1 = \arg\max r$. This occurs when agents have explored sufficiently and the Q-values of their actions approx-

imate their true values closely. The paper then introduces Lemma 2. They build upon the assumption that at time T_1 when the Q-matrix values for $\epsilon_t = 1$ have converged to the true expectations and for the next T_2 periods, they consider $\epsilon_t = 0$. They state Lemma 2 as follows:

Lemma 2. *With $\epsilon_2 = 0$ and assuming a_1 remains the greedy action throughout the exploration phase, let $T = T_1 + T_2$. Then $\mathrm{Q^T}(a_1) \xrightarrow{T\to\infty} \frac{1}{1-\delta} r(a_1, a_1)$*

The interpretation of this lemma is that if both agents play their greedy optimal action, their expected profit converges to the discounted profit of the action profile. All Q-values for the other actions remain as the ones stated in Lemma 1 since they are no longer being selected and hence updated. Now, Lemma 2 is built on the assumption that a_1 remains greedy. Therefore, there may exist some time period $\tilde{T}$ where $Q^{\tilde{T}(a_1)} < Q^{\tilde{T}(a_i)}$ for some i. Suppose it holds for $i = 2$, Then a_2 will become the preferred action. Next, re-applying the lemma, it is observed that $\mathrm{Q^T}(a_2) \xrightarrow{T\to\infty} \frac{1}{1-\delta} r(a_2, a_2)$. Next, the $\max\{Q^T(a_1), Q^T(a_2)\}$ is compared to $Q^T(a_3)$ for some 3 and the process is iterated for the remaining i. The following provides insight into this procedure:

Theorem 3.1. *Define J as the smallest j such that:*

$$\max_{i \leq j} r(a_i, a_i) > (1-\delta)\tilde{r}(a_{j+1}) + \delta\tilde{r} \tag{7}$$

With a two-step learning procedure, the agents converge on:

$$I \equiv \arg\max_{i \leq J} r(a_i, a_i) \tag{8}$$

Suppose we reach a point in the iterative procedure described above where there is no $j \notin \{1, 2, .., i\}$ such that $\max\{Q^T(a_1), Q^T(a_2), .., Q^T(a_i)\} < Q^T(a_j)$. Equation 7 characterises this type by providing a condition for the first time in the iterative procedure where this occurs, meaning that it can be terminated prematurely. As a consequence, finding the optimal policy reduces to a maximisation over the indices before the occurrence as shown in equation 8. Naturally, if the process never terminates, the problem reduces to maximising over all the indices. How does this argument prove that there is no deviation-punishment scheme? The

paper considers a setting with a one-period memory. They define the "reaction function" $g : A \times A \to A$ as follows:

$$g(a^{t-1,1}, a^{t-1,2}) = \begin{cases} a_{i+1} & \text{if } a^{t-1,1} = a^{t-1,2} = a_i \text{ with } i < I \\ a_I & \text{if } a^{t-1,1} = a^{t-1,2} = a_i \text{ with } i = I \\ a_1 & \text{if } a^{t-1,1} = a^{t-1,2} = a_i \text{ with } i > I \text{ or } a^{t-1,1} \neq a^{t-1,2} \end{cases} \tag{9}$$

The first two cases follow from Theorem 3.1. In the first case, since $i < I$, it means that the iterative procedure described before has not yet terminated (since $Q^T((a^{t-1,1}, a^{t-1,2}), a_i) < Q^T((a^{t-1,1}, a^{t-1,2}), a_I)$). Therefore, the next action is selected. The second case instead marks the end of the procedure; hence, a_I is selected. To break down the last case, we first consider the scenario where $i > I$. In this scenario, it means that the procedure terminates prematurely, and therefore the Q-matrix values for a_j for $i \leq j$ have not been updated since the exploratory phase. Therefore, the greedy action a_1 is played, followed by a series of readjustments to return to a_I. In the second case where $a^{t-1,1} \neq a^{t-1,2}$, we may have a deviation to a visited or unvisited state. If the state is unvisited, the same reasoning as before applies. Otherwise, we may have a deviation to an already visited state, meaning that the number of steps to return is reduced. The paper concludes the section with the following comments: "This rationalizes what has sometimes been described as a reward-punishment scheme, whereby a deviation is followed by several periods of bad outcomes until the system slowly returns to the pre-deviation prices. Instead of being a sign of agents' sophisticated understanding of collusive strategies, we showed that this is a simple and mechanical consequence of the learning process, which keeps a memory of past decisions" [8].

To support this argument, the Lambin performs several numerical simulations. In the appendix of the paper they illustrate several plots for the 1-period memory case.

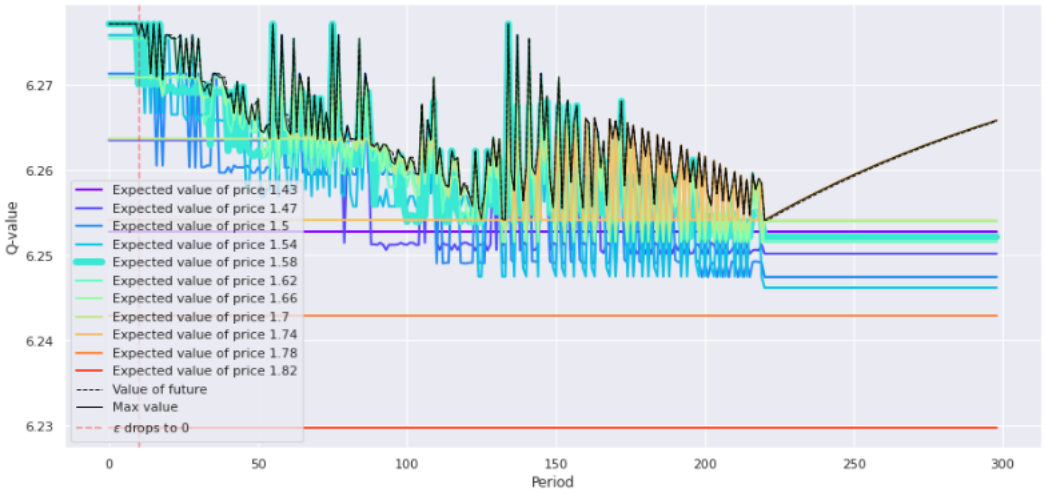


Figure 8: One-period-memory plot of Q-values of agents over time. The exploration rate is set to 1 for the first 1000 steps and is subsequently dropped to 0 for the remaining steps. Each line represents the Q-value for a different price, averaged over all states. [8]

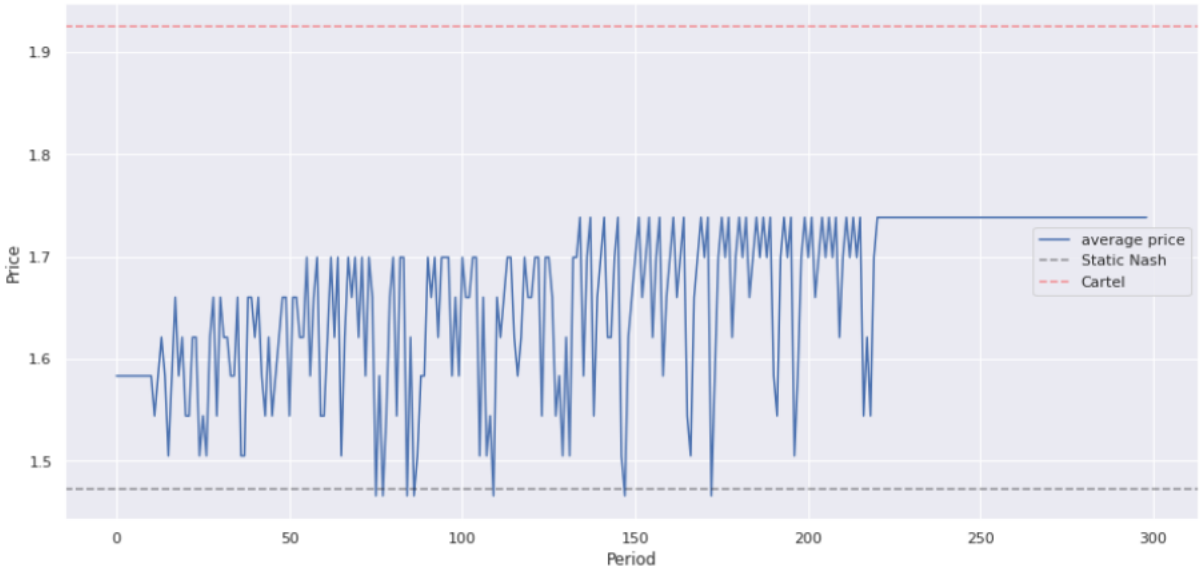


Figure 9: Q-value of actions over learning, with a 2 steps exploration policy. Agents switch from full exploration to full exploitation at time t = 10. [8]

In figure 8, as soon as the exploitation phase commences it can be seen that initially the Q-values start to frantically update and then converge to a fixed value according to lemma 2. Figure 9 illustrates the price set at each iteration. It is evident that as soon as the exploitation period commences there is a re-adjustment of the Q-values and subsequently a sequence of price changes over the next iterations. To extend the previous argument discussed from only considering a two step epsilon-greedy model to the $e^{-\beta t}$ model the paper gradually increases

the number of steps from two to three and so fourth, until a suitable approximation of $e^{-\beta t}$ is obtained.

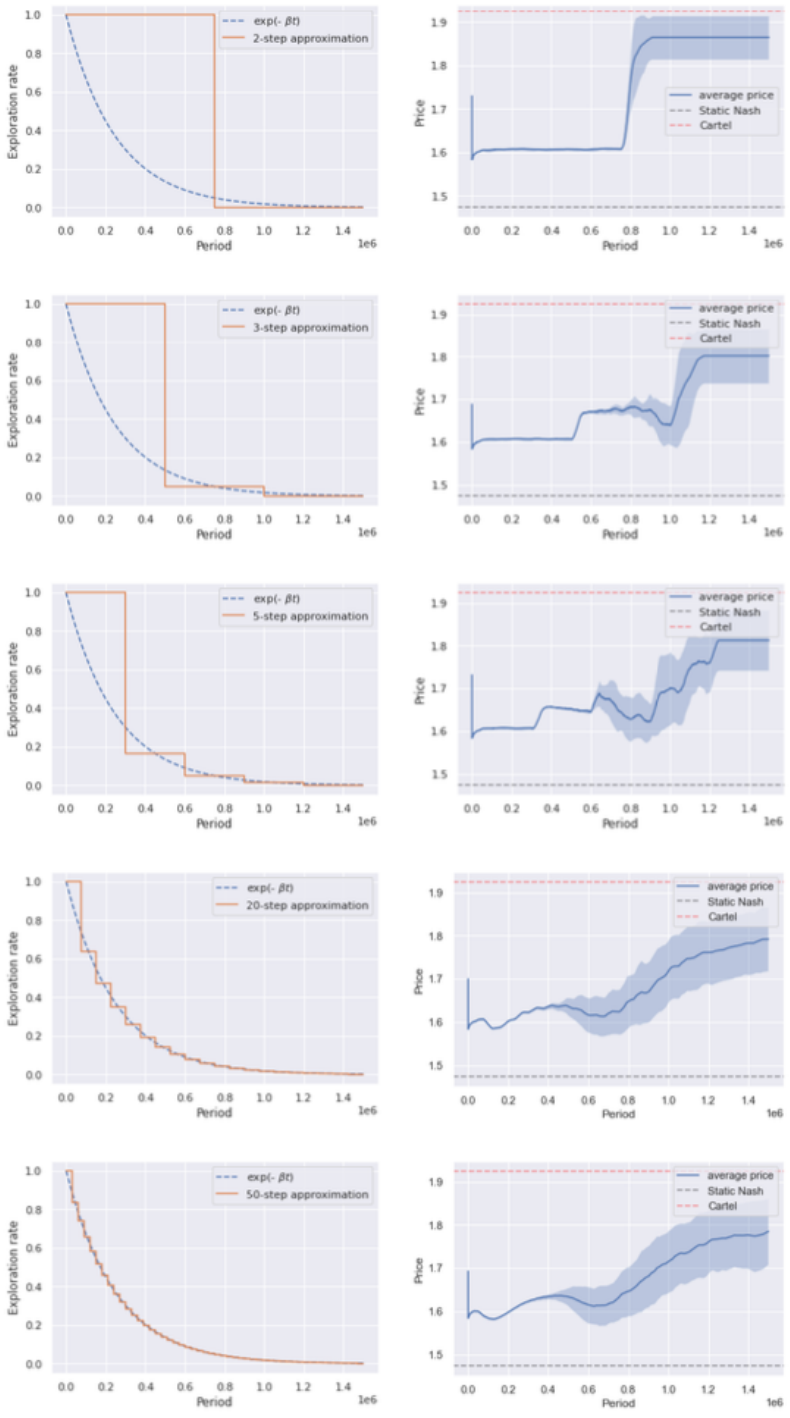


Figure 10: Average preferred price of agents over the course of learning with an approximated exponential decline of the exploration rate. The average is calculated over 1000 simulations. [8]

Figure 10 further illustrates the argument presented because each time the "n-step policy"

is smoothed, we see a less noisy adjustment of prices. This is due to the fact that the exponential decay ensures gradually diminishing of the exploration rate as the agent. This allows the agent to start exploiting the knowledge obtained from an earlier point in time, and gradually increase how much they are exploiting.

3.4 Summary

While the Calvano and Eschenbaum papers shed light on the tendency of algorithms to converge to supra-competitive prices and support the idea of collusion, Lambin's work challenges the interpretation of this phenomenon as evidence of sophisticated collusion strategies. Instead, Lambin suggests that high prices can emerge due to the intrinsic learning process, particularly when agents exhibit exploration-exploitation behavior. This implies that removing the bounded memory doesn't result in reduced collusion. While these papers offer valuable insights they also present certain limitations. The focus on specific parameter settings and simplified models may not fully capture the complexity of real-world markets. Additionally, the experiments primarily investigate collusion among symmetric agents, potentially overlooking asymmetries in market power or strategic behavior. Moreover, the results may not readily translate into actionable regulatory or policy measures without further validation or refinement. The Calvano papers suggests that agencies and courts could subpoena and test pricing algorithms in simulated environments to detect collusive behavior. However, as addressed before the complex nature of these markets makes them difficult to simulate accurately. As described in the Eschenbaum paper for one-period-memory algorithms, they were unusually sensitive to any sort of changes in the parameters, even just the initial seed [5]. For low-cost environments, players benefit from randomised play, achieving profits above the Nash equilibrium and in a high-cost context, players experience losses from randomising compared to playing Nash equilibrium, resulting in lower collusion. This thesis proposes an experiment aims to contribute by testing the adaptability of memoryless agents in varying cost environments, addressing some of the gaps related to testing the algorithms in a real-world context. By examining whether agents can 'collude' again after being re-trained in a different context, the experiment seeks

to provide deeper insights whether the proposed solutions of testing algorithms is feasible and even necessary. In the next section, the research question : "Can memoryless agents maintain collusion levels when transferred from a low-cost to a high-cost environment, and if so, how do the convergence times and collusion indices change?" will be addressed.

4 Experiment

4.1 Setup

In this section, the ability of memoryless agents to adapt to new environments will be explored. Specifically, referring to the notation used in section 3.2 to describe the Eschenbaum paper, two memoryless agents will be trained, one in $b_1^k(\cdot)$ and the other $b_2^k(\cdot)$. Then the algorithm trained in $b_1^k(\cdot)$ is placed in $b_2^k(\cdot)$ and re-trained. For the experiment, we will consider two contexts: one with a higher cost and one with a lower cost. In the original Calvano experiment, the Q-matrix is initialised with the Q-values that would arise if both agents randomised uniformly. However, in our experiment, we will:

1. Train 2 algorithms in each context.
2. From each context, select one of the Q-matrices out of two.
3. Copy the converged Q-matrix values from each of the selected algorithms and stack them into a single Q-matrix. This implies that the first row of the Q-matrix will contain the Q-values of the converged agent in the low-cost context, and analogously the second one in the higher-cost context.
4. We then copy the converged Q-matrix values (for one of the two firms in each of the environments) for both agents into the higher-cost context. This means that instead of initialising the Q-matrices randomly, they are now initialised with their converged values.
5. During the re-training process, we update only the Q-matrix of the agent trained in the lower-cost context. The goal is to verify whether the agents are able to 'collude' again and how many iterations it takes to reach collusive outcomes, if any.

4.2 Results - General Statistics

The procedure described in section 4.1 was repeated 126 times. In the low-cost context, we use the default $c = 1$ and for the high one $c = 1.7$. Figure 11 illustrates a table of samples

sessions. Figure 12 shows the correlations between the collusion indices and convergence times across context.

	time_convergence_low	time_convergence_high	time_convergence_low_in_high	collusion_index_low	collusion_index_high	collusion_index_low_in_high
0	1407590	1557035	2718511	0.931798	0.931822	0.611441
1	1958122	1514244	2797877	0.900328	0.928276	0.697438
2	1478935	1472506	2788012	0.972539	0.925163	0.687648
3	1542389	1835384	2884675	0.821975	0.885463	0.665304
4	1499246	1883798	2593611	0.921482	0.953418	0.819508

Figure 11: The dataframe presents a sample of the simulations conducted. Variables include time_convergence representing the number of iterations for convergence and collusion_index representing the collusion indices. Variables are defined for each context

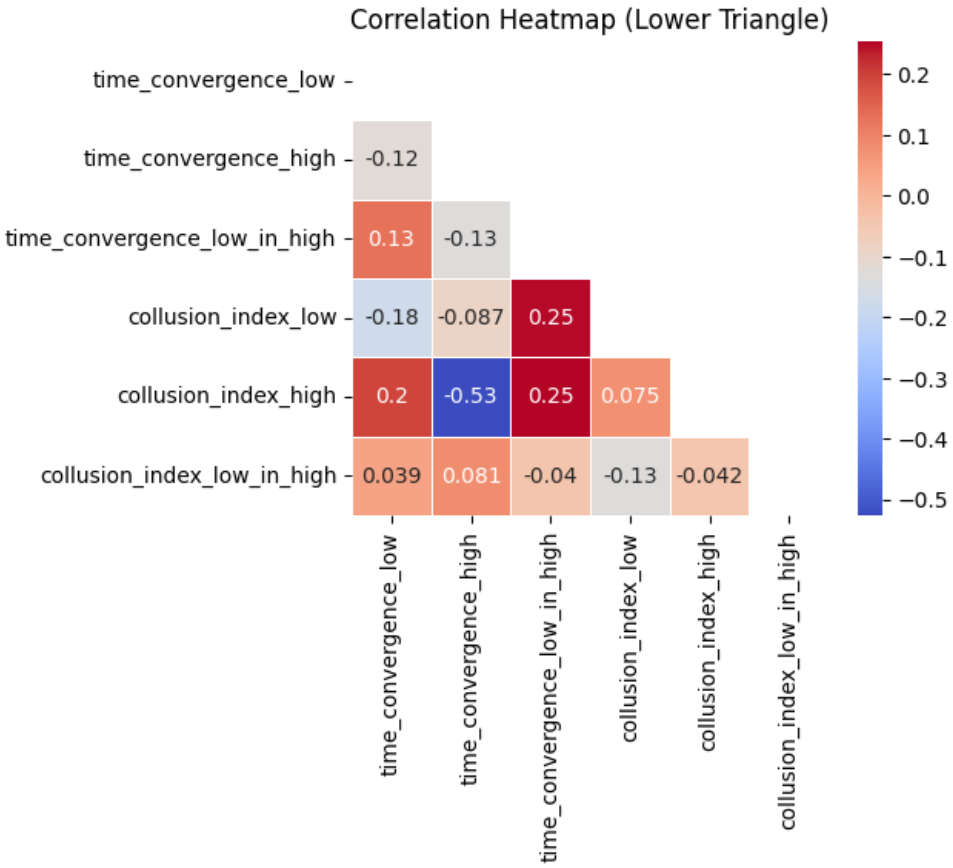


Figure 12: Correlation heatmap illustrating the relationships among different variables in the simulation results. The heatmap displays correlations between convergence times and collusion indices in the 3 contexts.

There is a negative correlation between time_convergence_high and collusion_index_high (-0.526). This might indicate that quicker convergence times at high levels are associated with higher levels of collusion. We also note that there is a positive correlation between time_convergence_low_in_high and collusion_index_low (0.249), and time_convergence_low_in_high

and collusion_index_high (0.254). This may indicate that the collusion levels attained previously impact the time it takes for re-training. In this case, contexts with higher collusion require more iterations.

	time_convergence_low	time_convergence_high	time_convergence_low_in_high	collusion_index_low	collusion_index_high	collusion_index_low_in_high
count	1.260000e+02	1.260000e+02	1.260000e+02	126.000000	126.000000	126.000000
mean	1.562267e+06	1.781602e+06	2.662635e+06	0.853285	0.858347	0.681650
std	2.258998e+05	3.397535e+05	2.104934e+05	0.095447	0.100184	0.175802
min	1.077309e+06	1.320025e+06	2.005295e+06	0.591781	0.429553	0.263721
25%	1.416074e+06	1.554930e+06	2.532451e+06	0.785509	0.793214	0.560893
50%	1.526637e+06	1.662913e+06	2.690010e+06	0.868382	0.881412	0.673755
75%	1.670714e+06	1.945738e+06	2.790596e+06	0.933228	0.931288	0.818656
max	2.413860e+06	2.763098e+06	3.184193e+06	0.990218	0.992108	0.993455

Figure 13: Summary statistics of convergence times and collusion indices across 126 sessions

The summary statistics of the convergence times and collusion indices provide insights into the performance and behavior of Q-learning algorithms in different cost environments. The mean convergence times for low-cost, high-cost, and low-in-high environments are approximately 1.56 million, 1.78 million, and 2.66 million iterations, respectively, indicating that the algorithms converge fastest in the low-cost environment and slowest when re-trained in the high-cost environment with initial low-cost values. The standard deviations indicate higher variability in the high-cost environment (339,753) compared to the low-cost environment (225,900).

Examining the collusion indices, we observe that the mean collusion index is quite similar for the low-cost (0.853) and high-cost (0.858) environments, suggesting that both settings show similar levels of collusion. However, when the low-cost trained agent is re-trained in the high-cost environment, the collusion index drops to 0.682, with a higher standard deviation (0.176), indicating more variability and a lower tendency to collude compared to the original contexts.

From this analysis, it is revealed that transferring knowledge from a low-cost to a high-cost setting reduces the propensity to collude and increases the variability of outcomes.

4.3 Results - Specific session

Below is an example of a given session and the results revealed. In the first simulation, the following results were obtained:

Table 3: Comparison of Results in Different Contexts for Session 1

Context	Collusion Index	Time for Convergence	Type of Convergence
Low Cost	0.84	1,500,000	1-Asym
High Cost	0.92	1,600,000	1-Sym
High Cost (merged)	0.93	2,700,000	1-Sym

In this session, it can be observed that retraining the low-cost algorithm in a high-cost context required nearly double the amount of iterations. However, high levels of collision are attained. Figure 14 illustrates a moving average of the profits for both firms across the entire training. For the first 1,500,000 iterations, the algorithm trained in the high context consistently achieves higher profits. This is due to the fact that the Q-values for firm 1 were already trained in the environment. Moreover, the Q-values of the agent trained in the low-context environment are continuously being updated as the agent is exploring and exploiting. Eventually, there is a sharp increase in the reward for the retrained agent, indicative of the retrained agent learning a new strategy to price more effectively than the agent in the market. However, approximately 1.5 million iterations were required to reach this point, indicating that algorithms are not effective at responding to changes in an environment.

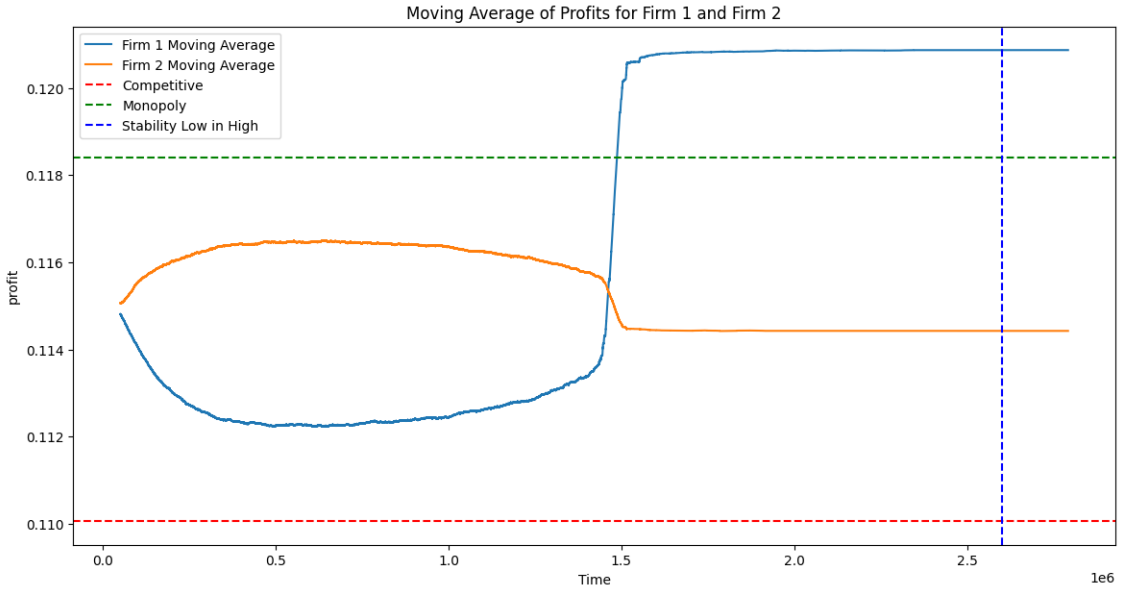


Figure 14: Moving average of profits for both firms over 2,500,000 iterations. Context: high cost (merged), with a moving average window of 50,000 iterations

Figures 15 and 16, on the other hand, show the moving averages for the agents trained in

their own contexts. The reward the agents receive changes frequently over time, due to the fact that both Q-matrices are updating and the agents are exploring / exploiting. However, what is interesting to note is that despite the fact that more updates are taking place, the convergence in both contexts takes significantly fewer iterations compared to the merged one.

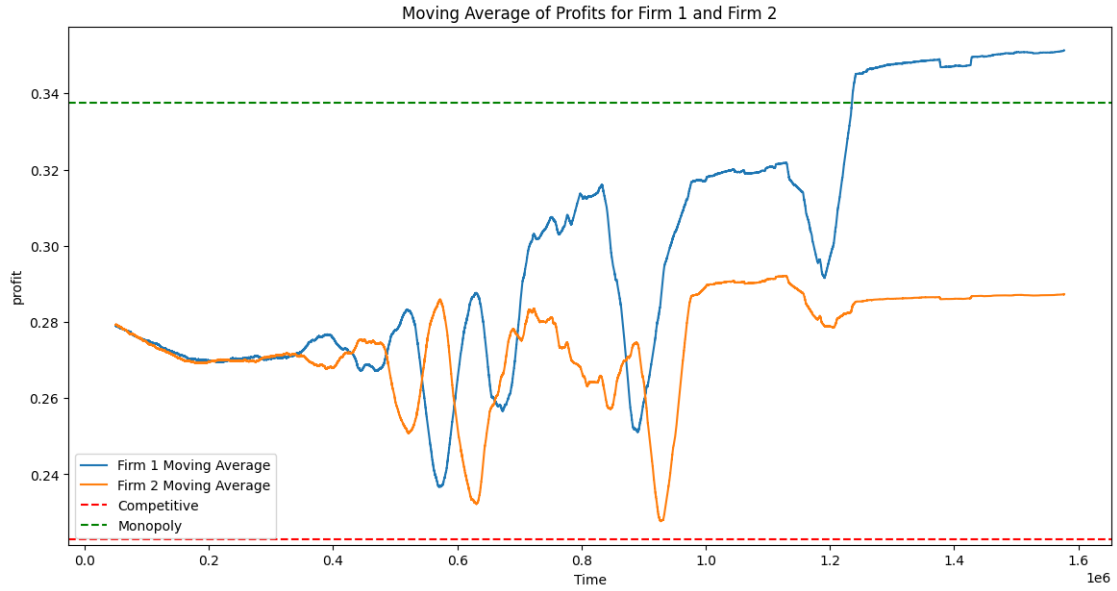


Figure 15: Moving average of profits for both firms over 1,500,000 iterations. Context: low cost, with a moving average window of 50,000 iterations

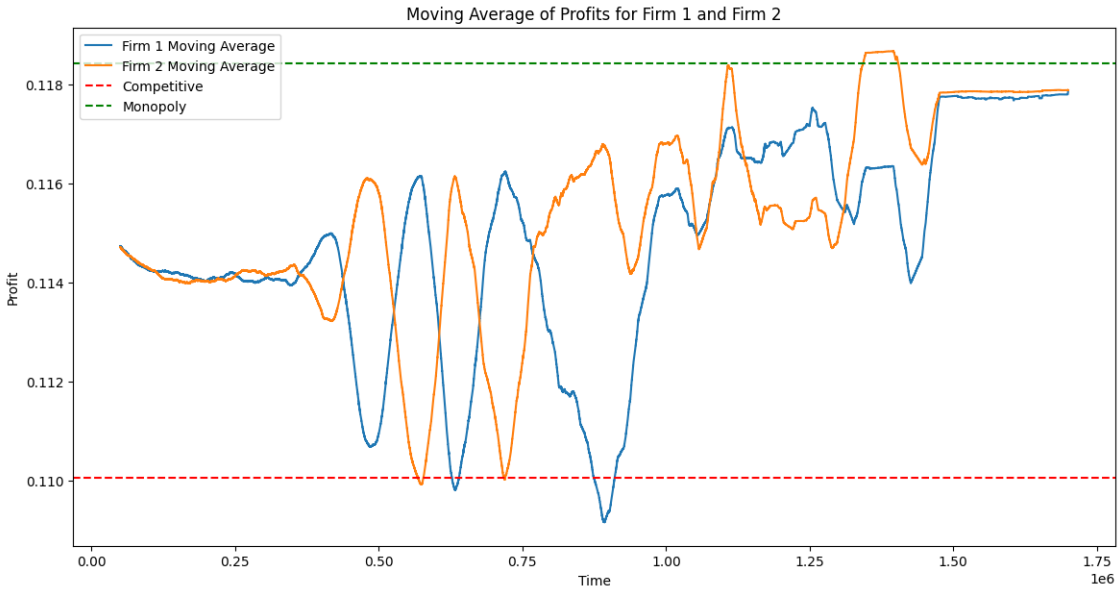


Figure 16: Moving average of profits for both firms over 1,600,000 iterations. Context: high cost, with a moving average window of 50,000 iterations

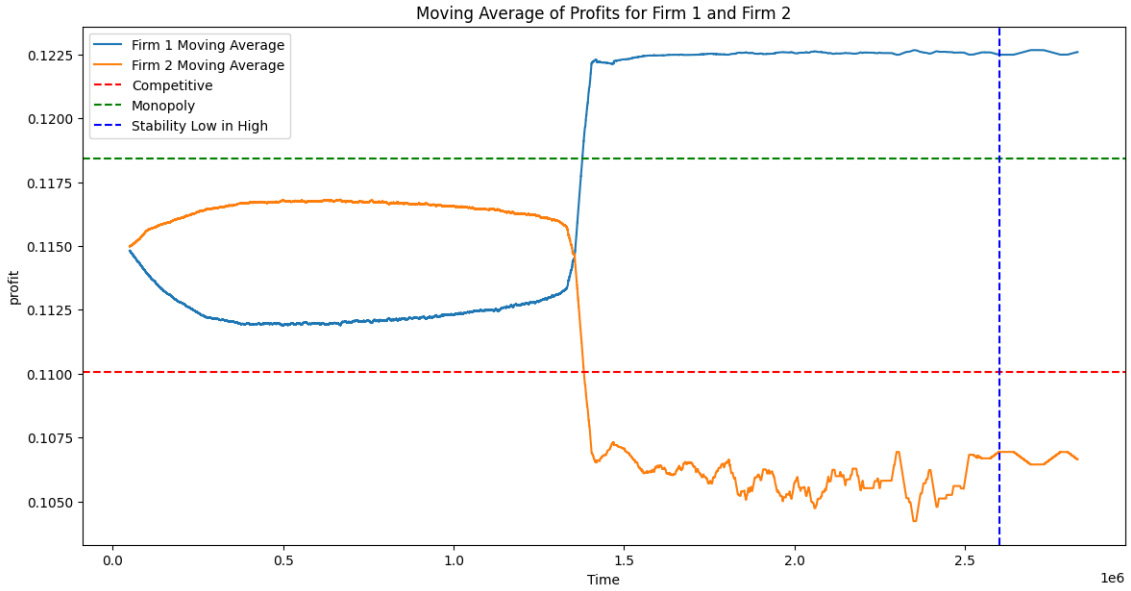


Figure 17: Moving average of profits for both firms over 2,500,000 iterations. Context: high cost (merged), with a moving average window of 50,000 iterations

Other sessions did reveal similar moving average plots for the retraining, as shown in Figure 17 for another session.

5 Conclusion

The literature review revealed several key findings regarding the behaviour of reinforcement learning algorithms in repeated game environments. The Calvano (2020) and Eschenbaum (2022) papers initially demonstrated that trained algorithms consistently set prices higher than competitive levels, even when playing optimally. They highlighted the presence of deviation-punishment schemes, where agents temporarily deviate from collusive prices before returning to them.

However, Lambin (2023) challenged this interpretation, arguing that high prices result from simultaneous experimentation and learning inertia inherent in reinforcement learning techniques. He provided theoretical insights and numerical simulations supporting the notion that supra-competitive prices emerge as a mechanical consequence of the learning process rather than sophisticated collusion strategies.

To answer the original research question: "Can memoryless agents maintain collusion levels

when transferred from a low-cost to a high-cost environment, and if so, how do the convergence times and collusion indices change?" the following can be said. From the Eschenbaum paper, it was revealed that the algorithms are very sensitive to small perturbations in the training environment and that retraining two converged algorithms in a new context can result in collusive outcomes. The experiment revealed that transferring knowledge from a low-cost to a high-cost setting resulted in reduced collusion propensity and increased outcome variability. While collusion levels were similar between low-cost and high-cost environments individually, retraining a low-cost-trained agent in a high-cost context led to lower collusion and longer convergence times. The moving average plots illustrated how retraining in a high-cost environment resulted in a more stable and less frantic change in reward over time. Moreover, for any real-world application, the number of iterations required to converge in a new context was excessively large. This result was surprising given that the new context considered Q-matrices which had already converged in their own contexts and only one was being updated.

Keeping in mind the scope of this field of research is providing agencies with a way to test pricing algorithms in simulated environments, the experiment revealed that in practice, it is infeasible to train an algorithm in a static context and have successful results in a dynamic environment given the excessively large number of iterations required to reach a collusive index. However, it does not necessarily render algorithms trained in static environments useless. Firms may have multiple algorithms trained in different contexts that they use in response to shocks in the current environment. If this is the case, then firms can overcome the problem of having to retrain a single algorithm.

Moreover, it is highly possible that in practice firms are training their algorithms in more complex environments, taking into account more factors. For example, for a specific product on Amazon, they may also consider the number of product visits, seller performance, and competitor prices. Unfortunately, creating a single general test setting is difficult since different algorithms are trained considering different factors and environments.

Another limitation of the experiment presented is that it only considers the classification of environments as low-cost or high-cost, which oversimplifies the complexity of real-world scenarios that may have many different changing variables by different scales. Future research

could explore a more nuanced categorisation of environments, considering varying degrees of cost, incorporate additional parameters that affect agent behaviour, and consider training environments with changing market conditions.

To conclude, it is very important that more research is done to further understand the behaviour of the algorithms and how they make decisions. Although our experiment revealed that algorithms are very sensitive to perturbations in parameters, it is still not clear why this is the case. More theoretical explanations of the behaviour of the algorithms are required. Even though this thesis focused more on the mechanisms of collusion, recent literature also focuses on developing tools and regulations to detect and prevent collusion in markets. An example is the detection of collusive behaviour through an empirical audit framework that checks data logged by pricing algorithms while they are deployed. This approach uses statistical testing of the algorithms' behaviour using collected data. This framework ensures regulators do not require inspecting source code or pre-approving algorithmic designs, which is infeasible in practice [6]. As stated in the introduction of Calvano, to directly detect collusion by only studying the outcomes of the simulations without considering the underlying mechanisms is not feasible.

References

[1] Stefania Bartoletti and Emilio Calvano. *Dallo Starting all'Advanced: gli ERC Grant sono di casa a Ingegneria ed Economia*. Published on 08/05/2023, Last modified on 28/08/2023. Università degli Studi di Roma Tor Vergata. 2023. URL: https://web.uniroma2.it/it/contenuto/dallo-starting-alladvanced-gli-erc-grant-sono-di-casa-a-ingegneria-ed-economia.

[2] Emilio Calvano et al. "Artificial Intelligence, Algorithmic Pricing, and Collusion". In: *American Economic Review* 110.10 (2020), pp. 3267–3297. DOI: 10.1257/aer.20190623.

[3] Le Chen, Alan Mislove, and Christo Wilson. "An Empirical Analysis of Algorithmic Pricing on Amazon Marketplace". In: Apr. 2016. DOI: 10.1145/2872427.2883089.

[4] Giacomo Di Girolamo. "Gli aerei con i prezzi più pazzi del mondo sono quelli che volano verso la Sicilia". In: *Linkiesta* (2023). URL: https://www.linkiesta.it/2023/01/aerei-prezzi-sicilia-italia-feste/.

[5] Nicolas Eschenbaum, Filip Mellgren, and Philipp Zahn. "Robust Algorithmic Collusion". In: *arXiv* (2022). Submitted on 2 Jan 2022 (v1), last revised 5 Jan 2022 (this version, v2). DOI: 10.48550/arXiv.2201.00345. arXiv: 2201.00345 [econ.GN]. URL: https://arxiv.org/abs/2201.00345.

[6] Jason D. Hartline, Sheng Long, and Chenhao Zhang. "Regulation of Algorithmic Collusion". In: *CSLAW '24: Symposium on Computer Science and Law*. Northwestern University, USA, 2024. DOI: 10.1145/3614407.3643706.

[7] Innovaformazione. "Algoritmo prezzi Ryanair". In: *Innovaformazione Blog* (2023). URL: https://innovaformazione.net/algoritmo-prezzi-ryanair/.

[8] Xavier Lambin. "Less Than Meets the Eye: Simultaneous Experiments as a Source of Algorithmic Seeming Collusion". In: *SSRN* (2024). DOI: 10.2139/ssrn.4498926.

[9] Statista Research Department. "Artificial intelligence (AI) in finance - statistics & facts". In: *Statista* (2024). URL: https://www.statista.com/topics/7083/artificial-intelligence-ai-in-finance/.

[10] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. Second. Adaptive Computation and Machine Learning Series. Cambridge, Massachusetts; London, England: The MIT Press, 2018. ISBN: 9780262039246. URL: https://lccn.loc.gov/2018023826.