DISCUSSION PAPER SERIES

DP13405
(v. 2)

ARTIFICIAL INTELLIGENCE, ALGORITHMIC PRICING AND COLLUSION

Emilio Calvano, Giacomo Calzolari, Vincenzo Denicolò and Sergio Pastorello

INDUSTRIAL ORGANIZATION

ISSN 0265-8003

ARTIFICIAL INTELLIGENCE, ALGORITHMIC PRICING AND COLLUSION

Emilio Calvano, Giacomo Calzolari, Vincenzo Denicolò and Sergio Pastorello

Discussion Paper DP13405
First Published 20 December 2018
This Revision 24 January 2019

Centre for Economic Policy Research
33 Great Sutton Street, London EC1V 0DX, UK
Tel: +44 (0)20 7183 8801
www.cepr.org

ARTIFICIAL INTELLIGENCE, ALGORITHMIC PRICING AND COLLUSION

Abstract

Increasingly, pricing algorithms are supplanting human decision making in real marketplaces. To inform the competition policy debate on the possible consequences of this development, we experiment with pricing algorithms powered by Artificial Intelligence (AI) in controlled environments (computer simulations), studying the interaction among a number of Q-learning algorithms in a workhorse oligopoly model of price competition with Logit demand and constant marginal costs. In this setting the algorithms consistently learn to charge supra-competitive prices, without communicating with one another. The high prices are sustained by classical collusive strategies with a finite phase of punishment followed by a gradual return to cooperation. This finding is robust to asymmetries in cost or demand and to changes in the number of players.

JEL Classification: L41, L13, D43, D83

Keywords: artificial intelligence, Pricing-Algorithms, Collusion, Reinforcement Learning, Q-Learning

Emilio Calvano - emilio.calvano@unibo.it
University of Bologna, Toulouse School of Economics

Giacomo Calzolari - giacomo.calzolari@unibo.it
European University Institute, University of Bologna, CEPR and CEPR

Vincenzo Denicolò - vincenzo.denicolo@unibo.it
University of Bologna and CEPR and CEPR

Sergio Pastorello - sergio.pastorello@unibo.it
University of Bologna

Acknowledgements
We thank without implicating Areil Ezrachi, Joshua Gans, Joe Harrington, Kai-Uwe Kuhn, Patrick Legros, Yossi Spiegel, Steve Tadelis, Emanuele Tarantino and participants at the 2018 NBER Economics of Artificial Intelligence Conference, Cambridge University, Universidad Carlos III, Universidad Autonoma de Barcelona, the 2$^\text{nd}$ Bergamo IO workshop, EAGP workshop at the European Commission (DG Competition), University of Napoli, EIEF (Rome), the 2018 Toulouse biannual postal conference, the Mannheim MaCCi workshop on Big Data and 12$^\text{th}$ Toulouse Digital Conference for useful comments. Financial support from the Digital Chair initiative at the Toulouse School of Economics is gratefully acknowledged.

ARTIFICIAL INTELLIGENCE, ALGORITHMIC PRICING AND COLLUSION[†]

EMILIO CALVANO[*‡], GIACOMO CALZOLARI[&‡§], VINCENZO DENICOLÒ[*§], SERGIO PASTORELLO[*]

DECEMBER 2018

Click here for the latest version

Abstract

Increasingly, pricing algorithms are supplanting human decision making in real marketplaces. To inform the competition policy debate on the possible consequences of this development, we experiment with pricing algorithms powered by Artificial Intelligence (AI) in controlled environments (computer simulations), studying the interaction among a number of Q-learning algorithms in a workhorse oligopoly model of price competition with Logit demand and constant marginal costs. In this setting the algorithms consistently learn to charge supra-competitive prices, without communicating with one another. The high prices are sustained by classical collusive strategies with a finite phase of punishment followed by a gradual return to cooperation. This finding is robust to asymmetries in cost or demand and to changes in the number of players.

Keywords: Artificial Intelligence, Pricing-Algorithms, Collusion, Reinforcement Learning, Q-Learning.

J.E.L. codes: L41, L13, D43, D83.

[†]We thank without implicating Areil Ezrachi, Joshua Gans, Joe Harrington, Kai-Uwe Kuhn, Patrick Legros, Yossi Spiegel, Steve Tadelis, Emanuele Tarantino and participants at the 2018 NBER Economics of Artificial Intelligence Conference, Cambridge University, Universidad Carlos III, Universidad Autonoma de Barcelona, the 2nd Bergamo IO workshop, EAGP workshop at the European Commission (DG Competition), University of Napoli, EIEF (Rome), the 2018 Toulouse biannual postal conference, the Mannheim MaCCi workshop on Big Data and 12th Toulouse Digital Conference for useful comments. Financial support from the Digital Chair initiative at the Toulouse School of Economics is gratefully acknowledged.

[*]Bologna University; [‡]Toulouse School of Economics; [&]European University Institute; [§]CEPR

1. INTRODUCTION

Firms are increasingly adopting software algorithms to price their goods and services. In a sample of over 1,600 best-selling items listed on Amazon, Chen et al. (2016) find that in 2015 more than a third of the vendors had already automated their pricing. Large firms often develop their own programs in-house, but a repricing-software industry has arisen, supplying turnkey pricing systems to smaller vendors.[1] As this industry is developing steadily, algorithmic pricing is likely to become even more prevalent in the future.

Actually, algorithmic pricing is not entirely new: airline companies, for instance, have been using revenue management software for decades now. But this software would set prices mechanically following the instructions of the programmer, who thus remains effectively in charge of the strategic choices.[2] The new vintage of pricing algorithms now emerging is radically different. Powered by Artificial Intelligence, these algorithms are much more "autonomous" than their precursors. They develop their pricing strategies from scratch, engaging in active experimentation and adapting to the evolving environment. In this learning process, they require little or no external guidance.

These developments raise various issues for competition policy. In particular, legal scholars and policy-makers alike have voiced concern that algorithmic pricing may facilitate tacit collusion.[3] While economic theory would not appear to lend unambiguous support to this thesis,[4] antitrust agencies are discussing the problem

[1] https://www.techemergence.com/ai-for-pricing-comparing-5-current-applications/ last accessed October 4, 2018. Algorithmic pricing is now so widely diffused that Amazon advertises its MarketPlace WebService API stressing that the API facilitates pricing automation. See https://developer.amazonservices.com/, last accessed October 4, 2018.

[2] Revenue management software typically comprises an estimation module and an optimization module. The estimation module estimates a structural model of the industry (which is often taken to be a monopoly, however, abstracting from strategic interactions). The optimization module then calculates the optimal prices.

[3] See, for instance, Ezrachi and Stucke (2016, 2017), the remarks of M. Vestager, European Commissioner, at the Bundeskarellamt 18th Conference on Competition, Berlin, March 16, 2017 ("Algorithms and Competition"), and the speech of M. Ohlhausen, Acting Chairman of the FTC , at the Concurrences Antitrust in the Financial Sector Conference, New York, May 23, 2017 ("Should We Fear the Things That Go Beep in the Night? Some Initial Thoughts on the Intersection of Antitrust Law and Algorithmic Pricing").

[4] Salcedo (2015) shows theoretically that optimized algorithms will inevitably reach a collusive outcome, but this requires that each algorithm periodically "decode" the others, which in the meantime have not changed. Such an assumption is somewhat far-fetched, so the practical relevance

seriously.[5]

To inform this policy debate, this paper takes an experimental approach. We construct Artificial Intelligence pricing agents and let them interact repeatedly in controlled environments (computer-simulated marketplaces). The results indicate that even relatively simple pricing algorithms systematically learn to play sophisticated collusive strategies. The strategies involve punishments that are proportional to the extent of the deviations and are finite in duration, with a gradual return to the pre-deviation prices. The algorithms learn these strategies purely by trial and error. They are not designed or instructed to collude, they do not communicate with one another, and they have no prior knowledge of the environment in which they operate.

To our knowledge, this is the first paper that clearly documents the emergence of collusive strategies among pricing algorithms. The previous literature in both computer science and economics has focused more on outcomes than strategies. But the observation of supra-competitive prices is not, per se, genuine proof of collusion, as high prices could be due to the algorithms' failure to learn the competitive equilibrium.[6] If this were so, there would be little ground for concern, in that the problem would presumably fade away as Artificial Intelligence developed further. If instead pricing algorithms even now can learn to collude, then with the spread of "smarter" programs the problem is likely, if anything, to worsen.

Our analysis not only shows that pricing algorithms do learn to collude but further suggests they may actually be better than humans at colluding tacitly. The exper-

of the result remains controversial.

[5]Most recently, the seventh session of the FTC Hearings on competition and consumer protection, November 13-14, 2018, centered on the "impact of algorithms and Artificial Intelligence." In fact, agencies have been actively engaged on the issue for a couple of years now. For example, the OECD sponsored a Roundtable on *Algorithms and Collusion* in June 2017, and in September 2017 the Canadian Competition Bureau released a discussion paper on the ability of algorithms to collude as a major question for antitrust enforcement (*Big data and Innovation: Implications for Competition Policy in Canada*). Lastly, the British CMA published a white paper on *Pricing Algorithms* on October 8, 2018.

[6]For example, Waltman and Kaymak (2008) study repeated Cournot competition among Q-algorithms both where the algorithms cannot condition current choices on past actions (where tacit collusion is by definition impossible) and where they can (where collusion becomes a possibility). They find that the extent to which firms manage to reduce output with respect to the Cournot-Nash static equilibrium "seems to be somewhat lower for firms with a memory [...] than for firms without a memory." (p. 3283) This suggests, however, that what they observe is not collusion but a failure to learn Nash.

imental literature has consistently found that human subjects are hardly able to coordinate without explicit communication. In a laboratory setting with no communication possible two agents sometimes manage to converge to slightly supra-competitive prices, but three agents typically set prices close to the static Nash equilibrium, and four or more actually tend to be more aggressive than Nash – a "rivalry" effect that is often attributed to experimental subjects' tendency to behave as if they were involved in a contest.[7] Our algorithms, by contrast, display a stubborn propensity to collude. Even though, in keeping with theory, the degree of collusion decreases as the number of competitors rises, substantial collusion continues to prevail when the active firms are three or four in number, when they are asymmetric, and when they operate in stochastic environments.

These results would appear to suggest that current policy on tacit collusion may no longer be adequate in the age of Artificial Intelligence (AI). In most countries (including the U.S. and Europe), tacit collusion is not now regarded as illegal.[8] The rationale is twofold. First, tacit collusion is held to be a chimera: illusory and practically impossible to achieve. And second, the position is that even if tacit collusion nevertheless occurred, it would be hard to detect. These assumptions imply that aggressive antitrust enforcement risks producing many false positives, while tolerant policy would result in relatively few false negatives. Our findings, however, suggest that the advent of AI pricing could well alter the balance between the two types of errors. More research is certainly needed before policy conclusions can be drawn, but there is good reason to hold that the issue deserves closer scrutiny.

The rest of the paper is organized as follows. The next section provides a self-contained description of the class of "Q-learning"algorithms, which we use in our simulations. These are relatively simple Reinforcement Learning algorithms with a finite state and action space, but the more sophisticated algorithms that have been developed recently in computer science share the same basic architecture. Section 3 reviews the related economics and computer science literature. Section 4 then gives a detailed description of the economic environments where the simulations are performed. Section 5 reports the main results, and Section 6 reports on a number of

[7]See for instance Huck et al. (2004) and the literature cited therein.

[8]In fact, the treatment of tacit collusion has long been one of the most controversial issues in competition policy, and although the prevalent approach is tolerant, some jurisdictions take a more aggressive stance.

robustness checks. Section 7 concludes with a discussion of some major open issues.

2. Q-LEARNING: A PRIMER

Artificial Intelligence is largely based on Reinforcement Learning (RL) algorithms. Such algorithms have a structure that naturally appeals to economists: they adapt their behavior to past experience, taking actions that have proven successful more frequently and unsuccessful ones less frequently.[9] In this way, the algorithms may learn an optimal policy, or a policy that approximates the optimum, with little or no prior knowledge of the particular problem at hand.

Within the set of Reinforcement Learning models, a highly popular class among computer scientists is Q-learning algorithms. These are designed expressly to maximize the present value of a flow of rewards in problems of repeated choice. In non-strategic, stationary settings, under mild regularity conditions, they are certain to eventually deliver the optimal policy. In strategic settings, they form the basis for the software that has recently obtained spectacular successes, achieving superhuman performances, in games such as Go,[10] the Atari video-games,[11] and, lastly, chess.[12]

2.1. *Single-agent problems*

Q-learning was originally proposed by Watkins (1989) to tackle Markov decision processes.[13] In a Markov decision process, in each period $t = 0, 1, 2, ...$ an agent observes a state variable $s_t \in S$ and then chooses an action $a_t \in A(s_t)$. For any s_t and a_t, the agent obtains a reward π_t, and the system moves on to the next state s_{t+1}, according to a (time invariant) probability distribution $F(\pi_t, s_{t+1}|s_t, a_t)$. In the simplest settings, the distribution may be degenerate, so π_t and s_{t+1} become deterministic functions of s_t and a_t.

The decision maker's problem is to maximize the expected present value of the

[9]For a comprehensive discussions of Reinforcement Learning in the computer science literature, see Sutton and Barto (2018). Erev and Roth (1995) is one of the first applications to economics.

[10]See Silver et al. (2017).

[11]See Mnih et al. (2015).

[12]See https://thenewstack.io/deepminds-new-milestones-on-the-road-to-artificial-general-intelligence/, last accessed December 17, 2018.

[13]The reader is referred to Sutton and Barto (2018) for an in-depth discussion.

reward stream:

$$(1) \qquad E\left[\sum_{t=0}^{\infty} \delta^t \pi_t\right],$$

where $\delta < 1$ represents the discount factor. This dynamic programming problem is usually attacked by means of Bellman's value function

$$(2) \qquad V(s) = \max_{a \in A} \{E\left[\pi|s,a\right] + \delta E[V(s')|s,a]\},$$

where s' is a shorthand for s_{t+1}. For our purposes it is convenient to consider instead a precursor of the value function, namely the Q-function representing the cumulative discounted payoff of taking action a in state s.[14] It is implicitly defined as:

$$(3) \qquad Q(s,a) = E(\pi|s,a) + \delta E[\max_{a' \in A} Q(s',a')|s,a].$$

The Q-function is related to the value function by the simple identity $V(s) \equiv \max_{a \in A} Q(s,a)$. If the agent knew the Q-function, he could then calculate the optimal action for any given state, i.e. the optimal strategy.

2.1.1. *Estimating the Q-matrix*

Q-learning is essentially a method for estimating the Q-function without knowing the underlying model, i.e. the distribution function $F(\pi, s'|s,a)$. This task is performed under the assumption that S and A are finite, and that A is not state-dependent. On these assumptions, the Q-function becomes an $|S| \times |A|$ matrix.

Q-learning algorithms estimate this matrix by an iterative procedure. Starting from an arbitrary initial matrix $\mathbf{Q}_0$, after choosing action a_t in state s_t, the algorithm observes π_t and s_{t+1} and updates the corresponding cell of the matrix $Q_t(s,a)$ for $s = s_t, a = a_t$, according to the learning equation:

$$(4) \qquad Q_{t+1}(s,a) = (1-\alpha)Q_t(s,a) + \alpha\left[\pi_t + \delta \max_{a \in A} Q_t(s',a)\right].$$

For all other cells $s \neq s_t$ and $a \neq a_t$, the Q-value does not change: $Q_{t+1}(s,a) =$

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

$Q_t(s,a)$. Equation (4) tells us that for the cell visited the new value $Q_{t+1}(s,a)$ is a convex combination of the previous value and the current reward plus the discounted value of the state that is reached next. The weight $\alpha \in [0,1]$ is called the learning rate.

If the iterative process converges to the true Q-matrix, the algorithm will learn the optimal policy $a(s) = \arg\max[Q(s,a)]$ with absolutely no prior knowledge of the function $F(\pi, s'|s,a)$. If there is to be a chance of really approximating the true matrix, however, all actions must be tried in all states. This means that the algorithm has to be instructed to experiment, i.e. to gather new information by selecting actions that may appear sub-optimal in the light of the knowledge acquired in the past.

Plainly, such experimentation is costly and thus entails a trade-off. The benefit of exploration is better learning, which in turn improves future decisions. The downside is that by picking an action randomly instead of choosing the one with the largest current Q-value, the algorithm does not fully exploit its stock of acquired knowledge.

Finding the optimal resolution to this trade-off is problematic. In the economic theory of optimal experimentation, progress has been made only in simplified settings (see e.g. Bergeman and Valimaki, 2006). For frameworks as general as those considered here, we do not yet know what an optimal exploration policy might look like. At any rate, Q-learning algorithms do not even try to optimize in this respect: the mode and intensity of the exploration are specified exogenously.

The simplest possible exploration policy – sometimes called the ε-greedy model of exploration – is to choose the currently optimal action (i.e., the one with the highest Q-value in the relevant state, also known as the "greedy" action) with a fixed probability ε and to randomize uniformly across all other actions with probability $1-\varepsilon$. Thus, ε is the fraction of times the algorithm is in *exploitation mode*, while $1-\varepsilon$ is the fraction of times it is in *exploration mode*. More sophisticated exploration policies can also be designed: for example, one may let the probability ε vary over time, as we do in what follows, or let the probability with which sub-optimal actions are tried depend on their respective Q-values.[15]

[15]For example, in the so-called Boltzman experimentation model actions are chosen with prob-

2.2. *Convergence*

One reason Q-learning is so popular is that for this model convergence to the optimal policy is guaranteed under quite general conditions (see Dayan and Watkins, 1992). It requires only that the algorithm's exploration policy belong to a class known as *Greedy in the Limit with Infinite Exploration* (GLIE). Loosely speaking, these policies satisfy three intuitive requirements: that exploration decrease over time; that if a state is visited infinitely often, the probability of choosing any feasible action in that state be always positive (albeit arbitrarily small); and that the probability of choosing the greedy action go to one as $t \to \infty$.

Under these conditions, convergence is always achieved, but completing the learning process may take quite a long time. 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. The larger the state or action space, the more iterations will be needed.

2.3. *Q-learning in repeated games*

Although Q-learning was originally designed to deal with Markov decision processes, it can also be applied to repeated games. In this case, however, stationarity is inevitably lost even if the stage game does not change from one period to the next.

One source of non-stationarity is that in repeated games with perfect information, where strategies consist of mappings from the history of past play to the set of actions, the state s_t must be defined so as to include players' actions in previous periods. But if the actions of all previous periods are included, the set of states increases exponentially with time.[16]

This problem can be fixed by bounding players' memory. With bounded recall, a

---

abilities

$$\Pr(a_t = a) = \frac{e^{Q_t(s_t,a)/\tau}}{\sum_{a' \in A} e^{Q_t(s_t,a')/\tau}}$$

where the parameter τ is often called the system's "temperature." As long as $\tau > 0$, all actions are chosen with positive probability. When $\tau = 0$, however, the algorithm chooses the action with the highest Q-value with probability 1, and thus ceases to experiment.

[16]Furthermore, there would be no hope of visiting the same state twice.

state s will include only the actions chosen in the last k stages, implying that the state space may be finite and time-invariant.

A more serious problem is that in repeated games the per-period payoff and the transition to the next state generally depend on the actions of all the players. If player i's rivals change their actions over time – because they are experimenting or learning, or both – player i's optimization problem becomes inherently non-stationary.

Such non-stationarity is the ultimate reason why there are no general convergence results for Q-learning in repeated games. That is, there is no guarantee that several Q-learning algorithms interacting in the same environment will learn an optimal policy (that is, collectively, a Nash equilibrium of the repeated game with bounded memory). Convergence and equilibrium play can be verified ex-post, however, as we shall do in what follows.

Note that even if the algorithms are playing a game, they may continue to learn in a non-strategic way. That is, algorithm i regards its rivals as part of the environment and continues to update its Q-matrix according to (4). It does not have to be aware that it is interacting strategically with other players: from its standpoint, the past actions of rivals are treated just like any other possibly relevant state variable. In the computer science literature, this approach to Q-learning in repeated games is called *independent learning*. This is the approach we use in our experiments.

2.4. *Beyond Q-learning*

These simple Q-learning algorithms, which we use in our simulations, are adopted widely in computer science and indeed form the basis of the most recent developments in the field. Such algorithms have two significant drawbacks, however. First, they are unable to deal with continuous state and action spaces. The spaces must be discretized, and the finer the discretization, the slower the learning. Second, the algorithms do not realize that they are playing a game, so their learning is non-strategic. The recent computer science literature has tried to address these problems.

2.4.1. *Deep learning*

With continuous state and action spaces, the Q-matrix becomes a smooth function. Deep learning algorithms estimate this function by means of iterative methods sim-

ilar in spirit to (4). When the state and action spaces are finite but large, a smooth approximation of the Q-matrix may also serve to speed up the learning process. In this case, at each period a deep learning algorithm would update not only the last cell of the matrix that has been visited but also a number of neighboring cells.

Deep learning has long suffered from failures to achieve convergence, but significant progress on this front has been made recently. For example, in 2015 Google patented a new method for updating the Q-function that has proved quite successful (even if theoretical proof of convergence is still lacking). Loosely speaking, the method uncouples the updating of the two Q-functions that appear on the right-hand side of (4). This is the method that has enabled the algorithms to attain superhuman performance in the complex strategic tasks mentioned above.

2.4.2. *Joint learning*

Independent learning implies that the competing algorithms do not realize they are playing a game. In computer science, this issue has been addressed in the so-called *joint learning* literature. A number of different approaches have been proposed. For example, with equilibrium-based joint learning, other players' actions are predicted by means of some sort of equilibrium notion.[17]

Generally speaking, joint learning faces three main challenges: (*i*) the observability of rivals' payoffs, necessary in order to anticipate their behavior but much more demanding, as a requirement, than the observability of actions; (*ii*) a curse of dimensionality, which emerges because tracing rivals' optimization increases the amount of information that each algorithm has to store and process; and (*iii*) the treatment of multiple equilibria: when different algorithms focus on different equilibria, problems of mis-coordination can arise.[18]

2.4.3. *State of the art*

The computer science literature has not yet developed a consensus approach to joint learning. And the more sophisticated "reasoning" of which joint-learning algorithms are capable does not seem to offset the advantage of independent learning in terms

[17]Different notions of equilibrium may be used: for example, Hu and Wellman (1998) use the Nash equilibrium, Greenwald and Hall (2003) the correlated equilibrium.

[18]Shoham et al. (2007) provide an excellent survey of these methods and their limits.

of lower dimensionality. In fact, the most spectacular successes in complex multi-agent environments have come through independent-learning algorithms (e.g. Silver et al. 2016, 2017, Tampuu et al. 2017).[19]

The deep learning approach, by contrast, seems both more settled and more promising. While the best way to guarantee convergence is still an unsettled question, considerable progress has unquestionably been made. Deep learning is currently being applied to the most diverse strategic settings, and there is little doubt that it will have a major impact on pricing software as well.

Nevertheless, here we have chosen to focus on simple, "tabular" Q-learning algorithms. In addition to being well understood, these simple algorithms can keep possibly arbitrary modeling choices to a minimum. In fact, our algorithms are fully described by just two parameters, the learning rate α and an experimentation parameter β, which will be defined precisely in a moment. This allows us to conduct extensive comparative statics with respect to the structure of the algorithms. More sophisticated algorithms have a more complex architecture requiring further parametrization.[20]

The downside of such relatively simple AI technology is the slowness of learning, but our research strategy focuses on the behavior of the algorithms once they have completed the training process. This "off-line" approach (i.e. with agents learning off the market) would appear to be legitimate as long as the economic environment is stationary, in that the algorithms may then be trained off-line before being put to work (as is often the case in real life).

In non-stationary environments, of course, the issue is more delicate. As is discussed in greater detail in the concluding section, in this case it seems likely that some learning must take place also on-line (i.e. with actual market play). The speed of learning then becomes important. Here, deep learning algorithms would appear today to be the elective choice. Extending our experiments to this framework is therefore an important task for future research.[21]

[19]Perhaps for these reasons, today's most popular software libraries, such as Google TRFL, do not include joint-learning algorithms.

[20]For example, with deep learning algorithms, in addition to the learning and experimentation parameters one must specify a functional form for the Q-function, the number of estimation layers, and the structure of the neural network in each layer.

[21]It should be noted, though, that the spectacular successes cited in Section 2 rely on off-line

3. THE LITERATURE

A number of papers have analyzed how Q-learning algorithms perform in oligopolies, competing either against fixed-strategy opponents or against other Q-learners. In the former case, theory predicts that the algorithms will learn to play a best response, and this is what the literature in fact finds. The latter case, on which we focus in what follows, is more interesting.

Before proceeding, it must be said that so far this literature has been dominated by computer scientists, who naturally pursue their own research agenda. As a result, the primary focus has been on algorithms' performance in terms of ability to converge, speed of convergence, and payoff. Relatively few studies have addressed the issue of whether Q-learning algorithms can learn to collude, and none provides clear evidence on this question.

3.1. *Staggered prices*

One group of studies specifically analyzes pricing games. To the best of our knowledge, all these papers consider a model of staggered pricing where firms commit to a price level for two periods, as in Maskin and Tirole (1988). Like Maskin and Tirole, these papers posit Markov behavior, meaning that a firm can condition its price only on rivals' current prices and not on past history (including its own previous prices).

In this setting, several works have found supra-competitive prices, at least to some extent. Sometimes, however, either the models or the findings exhibit anomalies that raise doubts about the interpretation of the results. For example, the demand function posited by Greenwald et al. (1999) and Tesauro and Kephart (2002) for the case of differentiated products is quite special. In addition, both these studies find that the more efficient firm makes lower profits than its less efficient competitor. As another example, Kononen (2006) finds that prices are already substantially supra-competitive when $\delta = 0$ (a case in which firms should effectively be playing a one-shot game) and do not increase significantly with the discount factor δ. One wonders whether this may be evidence not so much of collusion as of a failure to

learning as we do here, and it is likely that repricing algorithms currently available in the market rely on the same approach.

learn the optimal strategy.

The best paper along these lines is perhaps Klein (2018), which considers Maskin and Tirole's baseline example with homogeneous products and linear demand. In accordance with theory, he finds that two Q-learning algorithms converge either to a constant price or to a pattern that resembles an Edgeworth cycle. In his baseline case with 6 possible price levels, profits are roughly midway between the static Bertrand-Nash level and the collusive level: in the terminology of this paper, the *average profit gain*[22] is around 50%. Enlarging the set of actions to 12 or 100 possible prices increases the average profit gain and makes it much more likely that the algorithms will converge to an Edgeworth cycle rather than a constant price.

These results are interesting and in broad agreement with our own findings. However, the postulate of price commitment is controversial, especially given that software algorithms can adjust prices very quickly. The assumption is probably consequential, as both theory and experimental evidence suggest that commitment may facilitate coordination. In point of theory, Maskin and Tirole (1988) have shown that the set of equilibrium profits that can be supported given commitment is bounded away from the static, one-shot level of profit. By contrast, the folk theorem for repeated games says that there exist subgame perfect equilibria in which profits are lower than in the static equilibrium. As for the experimental evidence, Leufkens and Peeters (2011) have shown that in a model of staggered prices, substantial cooperation is easily attained even among human subjects. In fact, as Klein recognizes, Q-learning algorithms appear to achieve less cooperation than humans in similar settings. And in any event Klein does not analyze the strategies that support these collusive-seeming outcomes.

We differ from this literature in using the canonical model of collusion, i.e. an infinitely repeated Bertrand game in which all firms act simultaneously and condition their actions on past history. We depart from the canonical model only in assuming a bounded memory, for the reasons explained in the previous section. Also, we make an explicit analysis of the punishment strategies that sustain the collusive outcomes we observe. It turns out that these strategies hinge on the possibility of conditioning one's current price not only on competitors' but also on one's own past prices.

[22]For a formal definition, see equation (9) below.

3.2. *Cournot competition*

Another group of papers, including most notably Waltman and Kaymak (2008), has studied repeated Cournot games. Here all players act simultaneously and have bounded recall, as in our model. They, too, find outcomes that are far from fully competitive: specifically, the average profit gain in their experiment is around 60%.

In principle, the only difference with respect to our analysis is that the stage game is one of strategic substitutes rather than complements. It is not clear, a priori, whether this impedes or facilitates collusion. But in practice Waltman and Kaymak's findings do not seem to be based on equilibrium behavior. As noted above, they find less competitive outcomes when firms are myopic and have no memory – conditions under which collusion is either unfeasible or cannot emerge in equilibrium – than when they are more far-sighted and can condition current on past prices. This suggests that what Waltman and Kaymak actually find is a failure to learn to play Nash.

3.3. *Repeated prisoner dilemma*

When the action space is reduced to two feasible actions, both Bertrand and Cournot games collapse to some sort of prisoner dilemma. There is a substantial literature on Q-learning in the repeated prisoner dilemma, and we have also conducted extensive experimentation with this model. In a companion paper (Calvano et al., 2018), we survey the relevant literature and report our own results.

For our current purposes, suffice it here to say that the literature sometimes finds a significant amount of cooperation. But the prisoner dilemma is a special type of stage game, in that there is only one way to cooperate. It is often argued that tacit collusion is hard to achieve in practice because there are many supra-competitive prices, and players may find it difficult to coordinate in the absence of explicit communication.[23] Analyses based on the prisoner dilemma cannot address this concern. That is, even if Q-learning algorithms managed to cooperate in the repeated prisoner dilemma, this would not constitute conclusive evidence that they can learn to collude in more realistic settings.

[23]Khun and Tadelis (2018).

3.4. *Other literature*

Our analysis also relates to the economic literatures on market experimentation, on learning in games, and on evolutionary game theory. The main difference from the market experimentation literature (see Bergeman and Valimaki (2006) for a survey) is that Q-learning algorithms take the exploration policy as exogenously given and do not experiment optimally. As for the theory of learning in games (see Fudenberg and Levine (2016) and Dal Bò and Frechette (2018) for excellent recent surveys), the theory has considered models of Reinforcement Learning but mostly for passive learning, not active experimentation.[24]

Some scholars have also noted the analogy between Q-learning and evolutionary game theory,[25] in that both approaches study some sort of adaptive behavior. However, the evolutionary approach assumes that each agent follows a fixed strategy, and "learning" is through a replicatory dynamics that propagates the best-performing strategies. Our algorithms, by contrast, learn actively, exploring the environment and adapting their own behavior to take advantage of past experience.

4. EXPERIMENT DESIGN

We construct a number of Q-learning algorithms and let them interact in a repeated Bertrand oligopoly setting. Here we describe the environment in which the algorithms operate, the exploration strategy they follow, and the criterion we apply to end each computer simulation.

[24]In particular, this literature examines the ability of Reinforcement Learning models to replicate the experimental behavior of human subjects (see e.g. Erev and Roth (1995, 1998, 2014) and Ho et al. (2007)). The working hypothesis is that a good part of the difference between behavior in the laboratory and the model of the perfectly rational *Homo Oeconomicus* stems from learning. This is undoubtedly a promising approach, but our own focus is on markets in which prices are set by algorithms rather than human beings.

[25]See for instance Tuyls et al. (2006).

4.1. *Economic environment*

Our workhorse is a standard model of price competition with logit demand and constant marginal costs. In each period t, the demand for product $i = 1, 2, ..., n$ is:

$$q_{it} = \frac{e^{\frac{a_i - p_{it}}{\mu}}}{\sum_{j=1}^{n} e^{\frac{a_j - p_{jt}}{\mu}} + e^{\frac{a_0}{\mu}}}. \tag{5}$$

The parameters a_i may be taken as product quality indexes, which as such capture vertical differentiation, whereas μ is an index of horizontal differentiation. Product 0 is an outside good, so a_0 is an inverse index of aggregate demand.

Each product is supplied by a different firm, so n is also the number of firms. The per-period reward accruing to firm i is then $\pi_{it} = (p_{it} - c_i)q_{it}$, where c_i is the marginal cost. As usual, fixed costs are irrelevant as long as firms stay active.

We use the logit model because it has been widely applied in empirical work for its flexibility. As noted, it accommodates both vertical and horizontal differentiation, and it can also easily accommodate the case of stochastic demand. As a robustness check, however, we also considered the case of linear demand functions derived by quadratic preferences *à la* Singh and Vives (1984).

Although this benchmark model is deterministic, we also investigated stochastic demand and stochastic entry and exit.

4.1.1. *Discretization of the action space*

In a Bertrand game, the algorithms choose prices. Since Q-learning requires a finite action space, we discretize the model as follows. For each value of the parameters, we compute both the Bertrand-Nash equilibrium of the one-shot game and the monopoly prices (i.e., those that maximize aggregate profits). These are denoted by $\mathbf{p}^N$ and $\mathbf{p}^M$, respectively. Then, we take the set A of the feasible prices to be given by m equally spaced points in the interval $[(1-\xi)\mathbf{p}^N, (1+\xi)\mathbf{p}^M]$, where $\xi > 0$ is a parameter. For example, in our baseline specification we set $\xi = 0.1$, so prices range from 10% below Bertrand to 10% above monopoly.[26]

[26] As a robustness check, we have tried values of ξ large enough that the price interval also includes the unit costs. This turns out to be immaterial, as in fact observed prices are generally higher than Bertrand and lower than monopoly.

Our way of discretizing the strategy space implies that the exact Bertrand and monopoly prices may not be feasible, so there may be mixed-strategy equilibria both in the stage and in the repeated game. Since by design our algorithms generically end up playing pure strategies, they might then cycle around a "target" that is not feasible.

4.1.2. *Memory*

To ensure that the state space is finite, we posit a bounded memory. Specifically, the state is defined simply as the set of all past prices in the last k periods:

$$s_t = \{\mathbf{p}_{t-1}, ..., \mathbf{p}_{t-k}\}, \tag{6}$$

where k is the length of the memory. The assumption here is perfect monitoring, which is reasonable for markets where algorithmic pricing tends to be used. In these markets, prices are typically posted on the Internet and so are easily observed by competitors.[27]

Our assumptions imply that for each player i we have $|A| = m$ and $|S| = m^{nk}$.

4.2. *Algorithm settings*

As is pointed out in Section 2, given the set of states S and the set of actions A, a Q-learning algorithm is fully specified by the learning rate α , the discount factor δ, and an exploration strategy.

4.2.1. *Exploration*

We use one of the simplest exploration policies in the GLIE class, namely, an ε-greedy model with a time-declining exploration rate. More specifically, we set

$$\varepsilon_t = 1 - e^{-\beta t}, \tag{7}$$

where $\beta > 0$ is a parameter. This means that initially the algorithms choose in purely random fashion, but as time passes they make the greedy choice more and

[27]For example, the Amazon.com API allows sellers to recover current and past prices of any product with a simple query.

more frequently. The greater β, the faster exploration fades away.

4.2.2. *Initialization*

Q-learning algorithms may well start with a clean slate, such as $\mathbf{Q}_0 = \mathbf{0}$. However, to speed up the learning process the initial Q-matrices may be set in accordance with certain hypotheses. For example, our baseline choice is to set $Q_{i,0}(s, a)$ at the discounted payoff that would accrue to player i if opponents randomized uniformly:

$$Q_{i,0}(s, a_i) = \frac{\sum_{a_{-i} \in A^{n-1}} \pi_i(s, a_i, a_{-i})}{(1-\delta) \prod_{j \neq i} |A_j|}. \tag{8}$$

This is in keeping with our assumption that at first the choices are purely random. In a similar spirit, the initial state s_0 is drawn randomly at the beginning of each session.

However, we have run robustness checks with alternative initializations, corresponding to rivals playing, say, the static Bertrand-Nash equilibrium. With enough exploration, the results are insensitive to the way the matrix is initialized.

4.3. *Convergence*

As mentioned, for non-stationary problems like ours there are no general convergence results: we do not know whether the algorithms converge at all or, if they do, whether they converge to a Nash equilibrium.

But convergence can be verified in practice. We use the following criterion: convergence is deemed to be achieved if for each player the optimal strategy does not change for 25,000 consecutive periods. That is, if for each player i and each state s the action $a_{i,t}(s) = \arg\max [Q_{i,t}(a, s)]$ stays constant for 25,000 repetitions, we assume that the algorithms have completed the learning process and attained stable behavior.[28] We stop the simulation when this occurs, and in any case after one billion repetitions.

[28]Note that even if $\arg\max [Q_{i,t}(a, s)]$ does not change, the Q-matrices themselves might keep changing. An alternative standard of convergence would require that the entries of Q-matrices not change by more than some x% for a certain number of periods.

4.4. *Outcomes*

For each set of parameters, an "experiment." consists of 1,000 sessions. In each session, agents play against the same opponents for a large number of periods: up to a billion, or until convergence as defined above.

We can observe all variables of interest (prices, profits, market shares, etc), but we often elect to concentrate on one economic indicator, namely average profit gain Δ, defined as:

$$\Delta \equiv \frac{\overline{\pi} - \pi^N}{\pi^M - \pi^N}, \tag{9}$$

where π^N is the per-firm profit in the Bertrand-Nash static equilibrium, π^M is the profit under full collusion (monopoly), and $\overline{\pi}$ is the average profit in the last 25,000 repetitions. Thus, $\Delta = 0$ corresponds to the competitive outcome and $\Delta = 1$ to the perfectly collusive outcome. The main reason for focusing on Δ is that this index can be compared directly across different economic settings.

For sessions that converge, we take the outcome to be the average for the last 25,000 repetitions.[29] As discussed, we focus on long-run outcomes since in practice the algorithms are trained off-line before being put to work. In each experiment, we then calculate the mean and standard error for the 1,000 sessions.

5. RESULTS

Here the baseline economic environment consists of a symmetric duopoly ($n = 2$) with $c_i = 1$, $a_i = 2$, $a_0 = 1$, $\mu = \frac{1}{2}$, $\delta = 0.95$ and a one-period memory ($k = 1$). Holding this environment constant, we vary the learning and experimentation parameters α and β, which for now are taken to be the same for both competing algorithms. In the next section, we run extensive comparative statics on the economic parameters, including length of memory. But it is worth noting at the outset that the assumption of a one-period memory is less restrictive than it might seem, because, as is noted by Barlo et al. (2016), the richness of the state space may substitute for the length of the memory. Even with $k = 1$, the algorithms can effectively replicate strategies that, strictly speaking, would require a longer memory.

[29]Since the algorithms' asymptotic behavior is very stable, as we shall see below, we could actually average over a much shorter sub-period.

To illustrate, consider the simple case of a stage game with only two feasible actions, of the prisoner dilemma type. With $m = 2$, there are four possible states, namely (with obvious notation) CC, CD, DC and DD. In this setting, cooperating in state CC and defecting in all other states is a strategy whose incentive compatibility constraint is exactly the same as that of a true grim-trigger strategy, which in principle requires an infinitely long memory. Likewise, cooperating in states CC and DD and defecting otherwise has the same incentive compatibility constraint as a "one-period punishment" strategy. Loosely speaking, a unilateral defection is interpreted as a genuine deviation, which as such is punished. In contrast, joint defection is interpreted as punishment for a previous deviation and is accordingly followed by return to cooperation.[30]

When the state and action spaces are richer, it is possible to mimic more complex strategies as well. In fact, we find that our algorithms are able to learn punishment strategies that have finite but long duration, with a slow return to the pre-deviation prices.

5.1. *Learning and experimentation*

The learning parameter α ranges from 0 to 1. Generally speaking, extreme values of α tend to disrupt learning: when $\alpha = 0$, the algorithm does not learn at all; when $\alpha = 1$, it immediately forgets what it has learned in the past. So we expect learning to be most effective for intermediate values of α.

As for the experimentation parameter β, the trade-off is as follows. On the one hand, the algorithms need to explore extensively, as the only way to learn is multiple visits to every state-action cell (of which there are 3375 in our baseline experiments with $n = 2$, $m = 15$ and $k = 1$, and many more in more complex environments). On the other hand, in the short run exploration is costly. Here we abstract from this cost to consider only long-run outcomes (i.e., after the convergence of the algorithms). But exploration entails another cost as well, in that if one algorithm experiments more extensively, this creates noise in the environment, which makes it harder for

[30]Grim trigger strategies turn out not to be an effective way for Q-learning algorithms to cooperate. This is because with experimentation one algorithm would sooner or later defect, and when this happened both would be trapped in a protracted punishment phase that would last until further (joint) experimentation drove the firms out of the trap. In fact, cooperation in the repeated prisoner dilemma is typically achieved by means of one-period punishment strategies.

the other to learn. This externality means that in principle experimentation may be excessive even discounting the short-term cost.

Initially, we explore a grid of 50×50 equally spaced values in the intervals $\alpha \in (0, 1)$ and $\beta \in (0, 4 \times 10^{-5})$. While for α we explored the entire unit segment, for β we discarded *a priori* values that do not allow for sufficient experimentation. To get a sense of the pattern implied, when β is at its upper bound, the probability of choosing an action randomly drops to two thirds after just 10,000 repetitions, and to under 2% after 100,000 repetitions. As a result, many of the cells in the matrix, which corresponds to prices perceived initially as sub-optimal, would be visited no more than 4 or 5 times in a session.[31] This seems barely sufficient to guarantee decent learning.

5.2. *Convergence*

In spite of the absence of theoretical guarantees, in more than 99.9 % of the sessions we have obtained convergence by our definition.

Typically a great many repetitions are needed for convergence. For example, with $\alpha = \frac{1}{2}$ and $\beta = 2 \times 10^{-5}$ (the mid-point of our grid) convergence is achieved on average after 500,000 periods, for the simple reason that With that value of β, the probability of choosing an action randomly is still 14% after 100,000 repetitions. If the rival is experimenting at this rate, the environment is still too non-stationary for the algorithm to converge. In practice, convergence is achieved only when experimentation is nearly terminated. The long time to convergence is not surprising given the way the algorithms learn, i.e. mechanically but persistently. For our purposes, however, this is not a problem, as the algorithms are supposed to be trained off-line.[32]

[31] If the probability of choosing randomly were always equal to 1, each state-action cell would have been visited just 3 times after the first 10,000 repetitions. But in fact at that point ε would already be in the range of 2/3, meaning that cells that correspond to currently sub-optimal prices would be visited on average less than twice in all subsequent repetitions.

[32] From this viewpoint, convergence is actually quite fast, requiring on average less than a minute of CPU time, compared with several weeks for the algorithm that beat humans at Go: see Silver et al. (2017).

5.3. *Consistency*

Outcomes are quite stable across sessions. This is not a foregone conclusion, given that learning requires extensive experimentation which, being random, might well create considerable variation. Yet the standard error of the profit gain is typically less than 1 percentage point. Another sign of the consistency of the learning process is that the two algorithms, which in the baseline model are ex-ante symmetric, perform in a very similar manner: the difference between the two firms in profit gain is never statistically significant. This suggests that what the algorithms eventually do is not casual.

These results do not depend on the averaging across a large number of repetitions (25,000). In fact, upon convergence prices are quite stable. In almost half the sessions both algorithms keep charging the same price period after period. The other sessions display price cycles, but more than three quarters of them have a period of two, and all involve adjacent prices. We interpret these cycles not as Edgeworth cycles but rather as an artifact of our discretization, with actual prices orbiting around constant targets.[33]

5.4. *Equilibrium play*

Even if the algorithms learn to behave consistently, this might not ensure that they eventually play an optimal strategy. Again, this is guaranteed theoretically for single-agent decision making but not when different algorithms are involved. In this case, if both algorithms converged to an optimal strategy in the presence of the rival, they would converge to a Nash equilibrium of the repeated game with bounded recall.

Although not guaranteed *ex ante*, this property can be verified *ex post*, but to do so we cannot rely on a complete characterization of the set of Nash equilibria, which is totally unmanageable.[34] Instead, we perform a direct check of whether an algorithm's strategy is a best response to the one the rival has converged to. If not, we then calculate the distance from best response.[35] This check can be performed

[33]In theory the algorithms might also learn to split the market dynamically by taking turns, but apparently they do not.

[34]With 15 feasible prices and a one-period memory, there are 225 states, 225^{15} strategies, and 225^{30} strategy profiles.

[35]That is, we compare the actual Q-matrix with the theoretical matrix corresponding to the

either on path (verifying whether a Nash equilibrium is played), or both on and off path (verifying subgame perfection).

The extent to which the algorithms converge to a Nash equilibrium depends on the values of the learning and exploration parameters. Intuitively, learning is more difficult when experimentation is extensive (β low) and the algorithms quickly forget what they learned in the past (α high). Even in these cases, however, we observe a considerable amount of equilibrium play. For example, for $\alpha = 0.9$ and $\beta = 8 \times 10^{-6}$, (a point in the far south-east of our grid), in about one third of the sessions both algorithms play a best response to the strategy eventually adopted by the rival. For the remaining sessions, the observed Q-values differ from those associated with best response by only 5% on average.

With more persistent learning, equilibrium play becomes prevalent. For example, when α is lowered to 0.1 with the same level of experimentation, a Nash equilibrium is attained in 55% of cases, and each algorithm plays an individual best response more than 60% of the times. Moreover, even when the algorithms do not play best responses, they come quite close: the potential profit gain from a true best response is, on average, a mere 1%.[36]

These numbers suggest that the algorithms do learn to play Nash equilibria. A key implication is that once the learning process is complete, the algorithms cannot be exploited, no matter how smart the opponent is.[37]

Off the equilibrium path, things are somewhat different. Fewer than 2% of the sessions produce a subgame perfect Nash equilibrium. This should not come as a surprise, however, as sub-optimal cells are visited only few times. As a result, the algorithms cannot learn perfectly how to respond off path, especially for states corresponding to past actions far from the optimal. Still, we document that they are able to learn quite sophisticated punishment strategies.

---

rival's strategy upon convergence.

[36]That is, excluding the best responses for which the distance is nil. To some extent, equilibrium play is more common with even more extensive experimentation. For example, with $\alpha = 0.1$ and $\beta = 2 \times 10^{-7}$, the algorithms play a Nash equilibrium 80% of the times. Yet increasing experimentation even further eventually backfires, reducing the profit gain, probably because of the exploration externality mentioned above.

[37]Whether Q-learning algorithms could be exploited during the learning phase, if it were conducted on line, is an interesting and difficult question for future study.

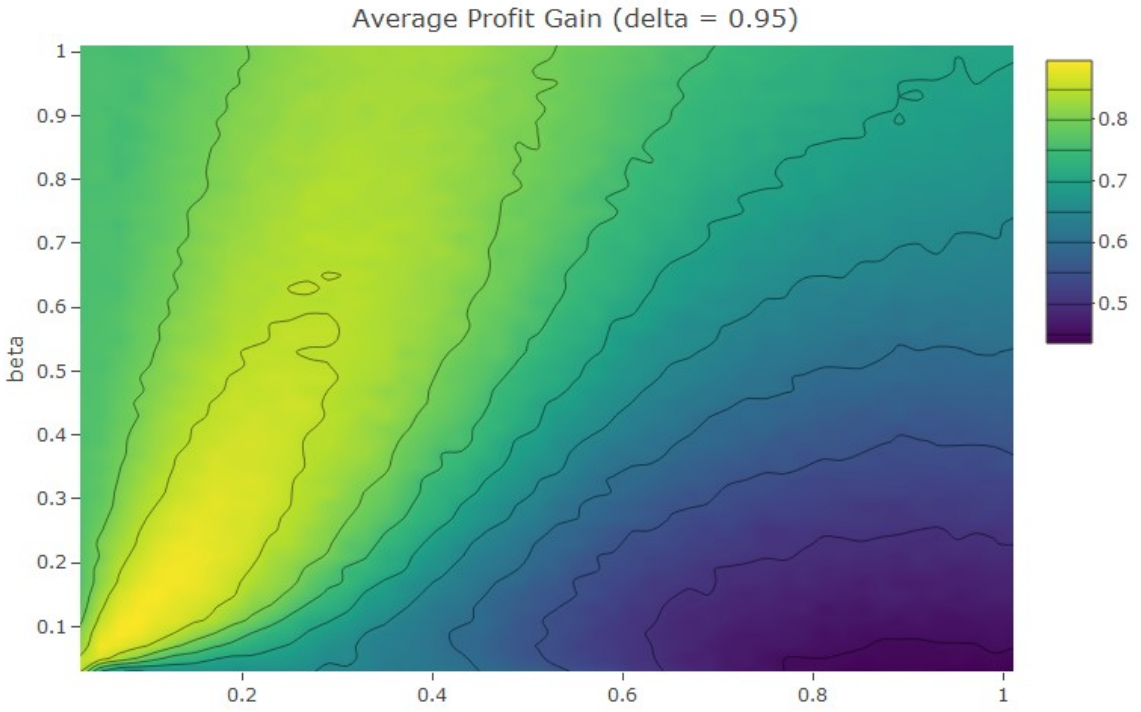

Figure 1: Average profit gain Δ for a grid of values of α and β. (Note that β has been rescaled to vary from 0 to 1.)

5.5. *Outcomes*

What do the algorithms do once they are fully trained? In this subsection, we show that they systematically charge supra-competitive prices and earn supra-competitive profits. In the next subsection we inquire into the strategies that underpin these collusive-looking outcomes.

The average profit gain Δ is represented in Figure 1 as a function of α and β. In the interval we have considered, Δ ranges from a minimum of 40% to a maximum of 99%. Although the profit gain is sensitive to the learning and experimentation parameters, non-competitive outcomes are quite common, not obtained at just a few selected points.

Generally speaking, the profit gain tends to be relatively small when there is little exploration, but it is actually smallest when exploration is extensive and α is very high. For this range of parameters, the algorithms explore widely but forget rapidly. This does not look like a particularly effective way of learning, and in fact it leads to

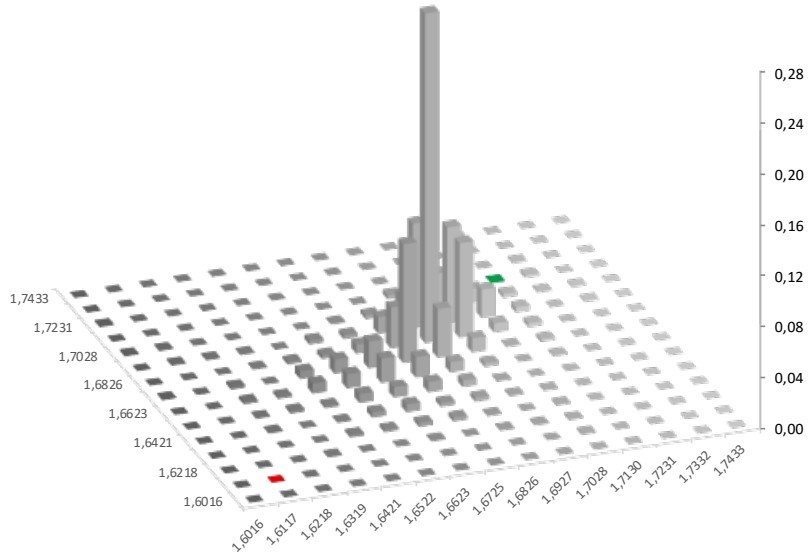

Figure 2: Histogram of states visited in the last 25,000 iterations pooling all data from 1000 sessions with $\alpha = 0.05$, $\beta = 8 \times 10^{-6}$. Green=(p^M, p^M), red=(p^N, p^N).

relatively limited equilibrium play, as we have already observed. Yet not even this stolid approach leads to cutthroat competition.

A more sensible method is to combine a moderate rate of learning with extensive exploration. In this case, we find not only more equilibrium play but also higher profits. For example, when α ranges between 0 and 0.2, and β between 0 and 2×10^{-5}, the average profit gain is systematically around 80% or more.

Given the consistency across sessions documented above, the observed values of Δ arise not because the algorithms sometimes learn to collude and sometimes do not, but rather because they learn systematically to collude partially. This is confirmed by examining the prices that they eventually charge. Figure 2 shows the relative frequency of the different price combinations for an experiment that is representative of the entire batch. Prices are rarely as high as under monopoly but are almost always higher than in the Bertrand-Nash equilibrium. Price dispersion is low, and firms tend to price symmetrically.

In sum, our algorithms consistently and symmetrically charge supra-competitive prices, obtaining a sizable profit gain.

5.6. *Strategies*

Let us now turn to the issue of what strategies underpin these documented non-competitive outcomes. The key question is whether the high prices are the result of the algorithms' failure to learn the static Bertrand-Nash equilibrium or of genuine collusion. The policy implications would be radically different: the former means that the algorithms are not smart enough, the latter that they are already, in a sense, "too smart." As AI technology advances, in the former case the problem is likely to fade away; in the latter, to worsen.

At this point, it may be useful to spell out exactly what we mean by collusion. Following Harrington (2017), we define collusion as "a situation in which firms use a reward-punishment scheme to coordinate their behavior for the purpose of producing a supra-competitive outcome." That is, what is crucial is not the level of profits as such but the way the supra-competitive result is achieved. Even extremely high profit levels may be regarded as collusive only insofar as deviations that are profitable in the short run would trigger a punishment.

We have already noted that the average profit gain seems to increase with the amount of equilibrium play, which itself suggests that Q-learning algorithms actually do learn to collude. Here we set forth two additional, perhaps even more compelling arguments. First, in economic environments where collusion cannot arise in equilibrium, we find that the algorithms learn instead to set competitive prices. Second, going back to settings where collusion is possible, we perform the following test: we exogenously force one player to defect by manually lowering its price for some time and observe how the algorithms react. In fact, they punish such defections, with a characteristic pattern: the punishment is proportional to the extent of the deviation and of finite duration, with a gradual return to the pre-deviation prices. This is perhaps the most direct possible evidence of collusive behavior where collusion is tacit.

5.6.1. *Competitive environments*

In certain environments, collusion is impossible by default; in others, it can never arise in equilibrium. If the supra-competitive prices that we find were the result of erratic choices, or of something other than collusion, then they should also be ex-

pected in settings where collusion is impossible. In particular, collusion is impossible when $k = 0$ (the algorithms have no memory), and it cannot arise in equilibrium when $\delta = 0$ (the immediate gain from defection cannot be outweighed by the loss due to future punishments). As noted in Section 3, the literature has indeed found supra-competitive prices even in these settings, which poses the question of how to interpret the findings in economic terms.

By way of contrast, in both of the above cases we find profit gains on the order of 10% to 20%. This means that the algorithms do learn to charge the static Bertrand-Nash prices when this is the only equilibrium of the game. If they do not play the competitive equilibrium when other equilibria exist, it must be because they have learned more sophisticated strategies.

5.6.2. *Deviations and punishments*

To look into how these strategies are structured, we have designed a specific exercise. At the end of a session we let the agents play for a number of periods according to the *learnt* strategies. Then we step in and manually override one agent's choice, forcing it to defect. We impose not only defections lasting for a single period but also defections lasting several periods; and defections both to the static best-response and to smaller price cuts. For all of these cases, we then examine the reaction of both agents in the subsequent periods. In a word, we derive "impulse-response" functions.

Figure 3 shows the average of 1,000 impulse-response functions derived from this exercise for all the sessions of one experiment ($\alpha = 0.05$, $\beta = 8 \times 10^{-6}$, $\delta = 0.95$) that is representative of the region in the grid where collusion is most pervasive. It shows the average prices chosen by agents τ periods after the deviation. The bottom plot simply provides greater detail zooming in around the shock. Figure 3 depicts the case of a one-period deviation to the static best-response to the pre-deviation price.

Clearly, the exogenous deviation gets punished, but the punishment is not as harsh as it could be (e.g., a reversion to the static Bertrand-Nash equilibrium), and it is only temporary: in the subsequent periods, the algorithms gradually return to their pre-deviation behavior.[38] We have also considered smaller exogenous price cuts:

[38]In fact, on average prices stabilize on a new level that is slightly lower than the pre-deviation

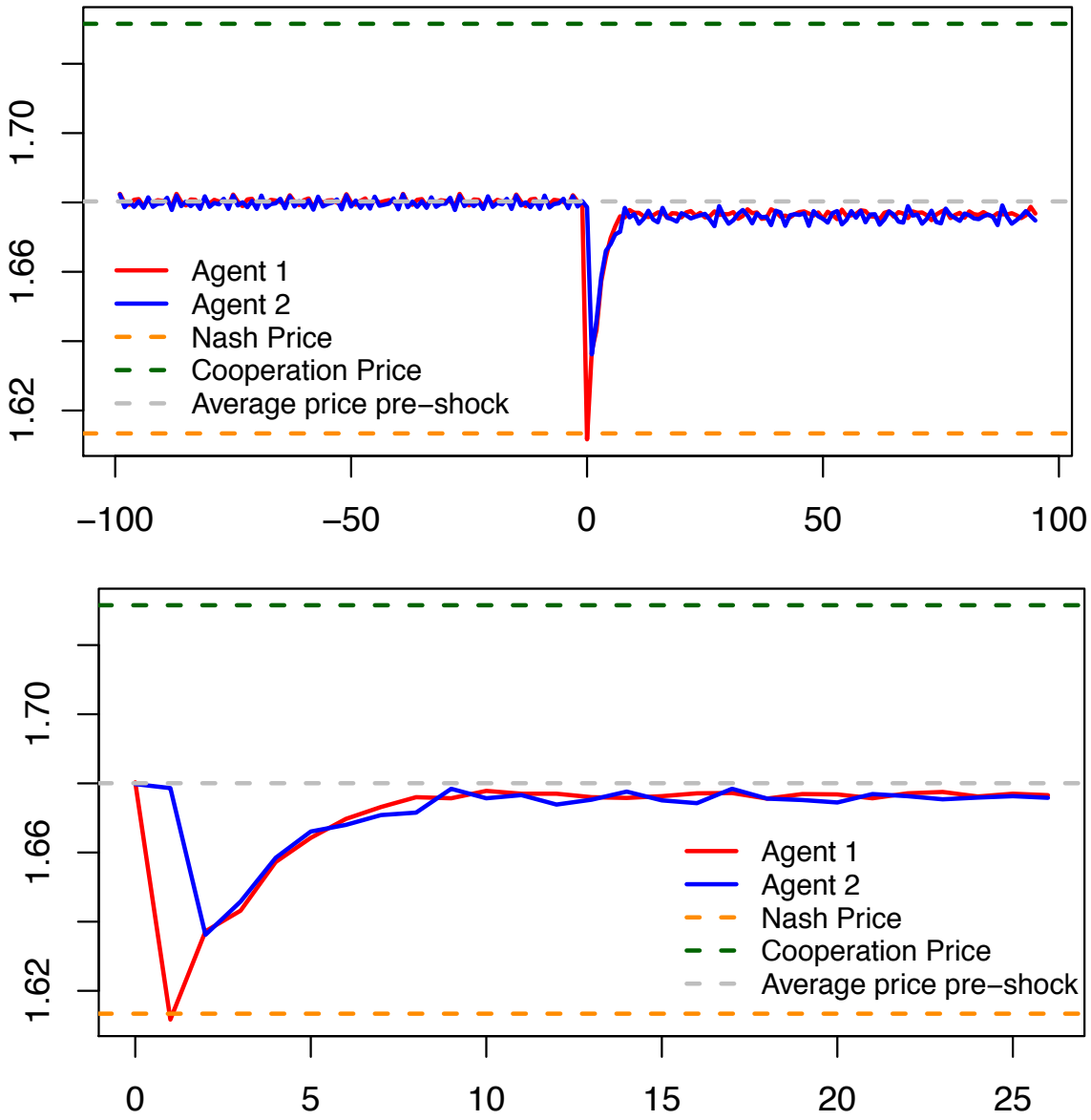

Figure 3: Price Impulse response, $\alpha = 0.05, \beta = 8 \times 10^{-6}, \delta = 0.95$. The top plot zooms out around the shock to show pre- and post-deviations levels.

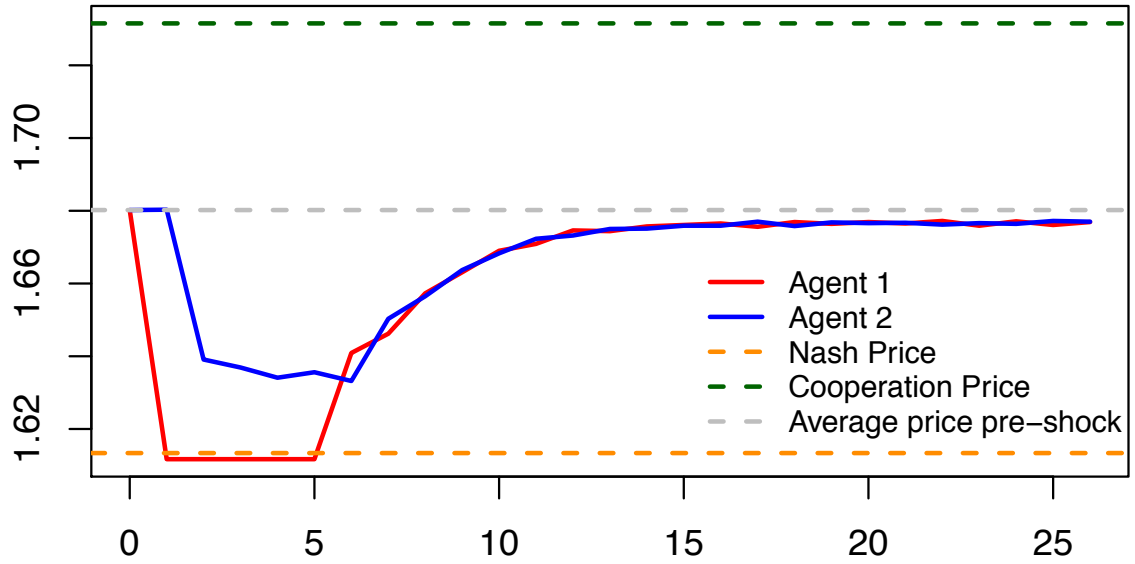

Figure 4: Price Impulse response, 5-period deviation, $\alpha = 0.05, \beta = 8 \times 10^{-6}, \delta = 0.95$.

when the deviation is milder, the punishment tends to be less harsh.

On the face of it, one may wonder whether the algorithms are effectively punishing the deviation, or whether instead the price cuts simply serve to regain market share. Looking only at the non-deviating firm's behavior, in fact, it is hard to tell these alternative explanations apart. But if you focus on the behavior of the deviating firm, the difference becomes perfectly clear. Given that in the deviation period (i.e., period $\tau = 1$) the rival has stuck to its old price, in period $\tau = 2$ the deviating algorithm, which meanwhile has regained control of the pricing, has no reason to cut its price endogenously unless it is taking part in the punishment itself. If its only concern were to maintain its market share, the deviating algorithm would cut its price only in period $\tau = 3$, i.e. after observing the rival's price reduction in period $\tau = 2$. Actually, however, in period $\tau = 2$ the deviating algorithm prices almost exactly the same as the other. This clearly shows that the deviating algorithm is responding not only to its rival's but also to its own action. Such self-reactive behavior is often crucial to achieve genuine collusion,[39] and it would be difficult to rationalize otherwise.

---

one.

[39] See for instance Maskin and Tirole (1982).

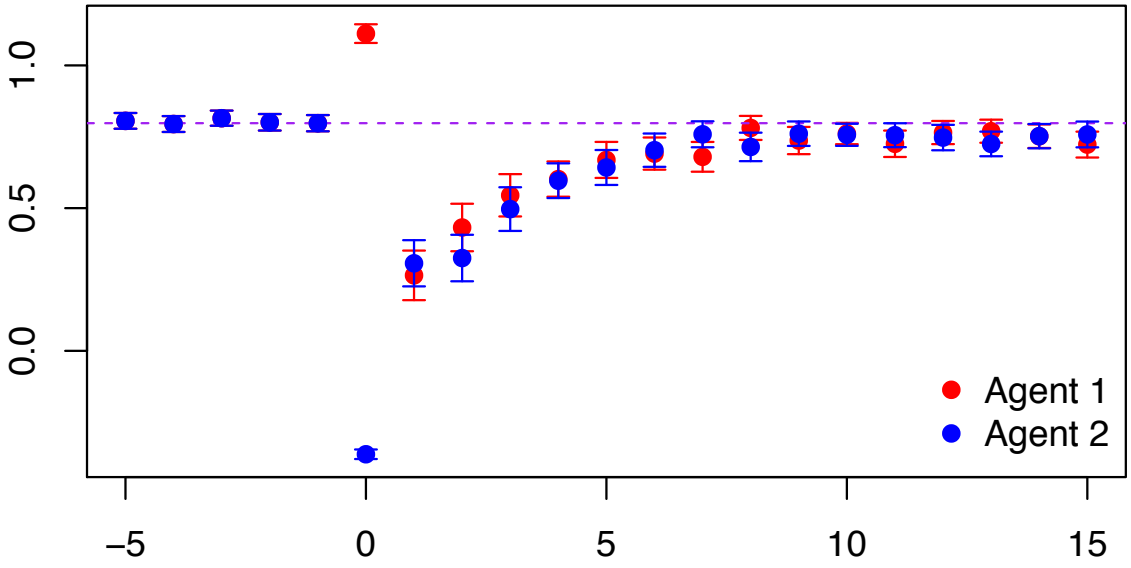

Figure 5: Impulse response of Profit gain Δ for $\alpha = 0.05, \beta = 8 \times 10^{-6}, \delta = 0.95$

Figure 4 illustrates the case of a 5-period deviation. The punishment gets a bit harsher after the second deviation but then stabilizes. In any case, the non-deviating firm stands ready to forgive: as soon as the defection ends, there is a gradual return to the original prices, on a pattern similar to the one-period deviation.

Going back to that case, Figure 5 shows the evolution of profits for a one-period deviation. Clearly, the punishment makes the deviation unprofitable. Agent 2's retaliation wipes out agent 1's profit gain already in the very next period.

Summarizing, we find clear evidence of collusive behavior: the supra-competitive profits are supported by off-path punishments. These punishment have a finite duration, with a slow return to the pre-deviation prices.

6. ROBUSTNESS

In this section, we hold the learning and experimentation parameters constant and consider a number of economic factors that may affect firms' ability to sustain a tacit collusive agreement. Specifically, we assume $\alpha = 0.05$ and $\beta = 8 \times 10^{-6}$. As a benchmark, for these parameter values the average profit gain in the baseline experiment is $\Delta = 80\%$.

We first consider some factors that might disrupt coordination, showing that algorithmic collusion is in fact quite robust. We then deal with other, more neutral factors whose impact is not clear *a priori*.

6.1. *Number of players*

Theory predicts that collusion is harder to sustain when the market is more fragmented. This is indeed one reason why antitrust authorities regulate mergers: apart from the unilateral effects, the concern is that more concentrated markets may be conducive to tacit collusion.

The experimental literature provides some support for this thesis, showing that without explicit communication it is very difficult for more than two agents to collude. With three agents, prices are typically pretty close to the static Bertrand-Nash equilibrium, and with four agents or more they may even be lower.[40]

In this respect, Q-learning algorithms would appear to be different. In simulations with three firms, the average profit gain Δ decreases only marginally, from 80% to 74%. With four agents, the profit gain is still a substantial 70%. The fact that Δ decreases accords with the theoretical predictions; the fact that it decreases so slowly seems to be a peculiar and somewhat worrying property of pricing algorithms.

What changes when the number of firms increases is the structure of the punishment. Figure 6 shows the impulse-response function for the case of three firms. The algorithms no longer return to the pre-deviation prices but settle quickly to a price level intermediate between the old and the deviation prices.

One possible explanation is that the enlargement of the state space impedes learning. Indeed, moving from $n = 2$ to $n = 3$ or $n = 4$ drastically enlarges the Q-matrix, from 3,375 to around 50,000 or over 750,000 entries. Since the amount of exploration is held constant, the increase in the size of the matrix makes learning off the equilibrium path much more incomplete.[41] However, we still have punishments, and in fact the punishment is harsher than in the two-firms case.

[40] See for instance Huck et al. (2004). Cooper and Khun (2014) stress the importance of communication to sustain collusion in the lab.

[41] It seems that for Q-matrices of this size some kind of Q-function approximation may yield more reliable results. We come back to this issue in the concluding section.

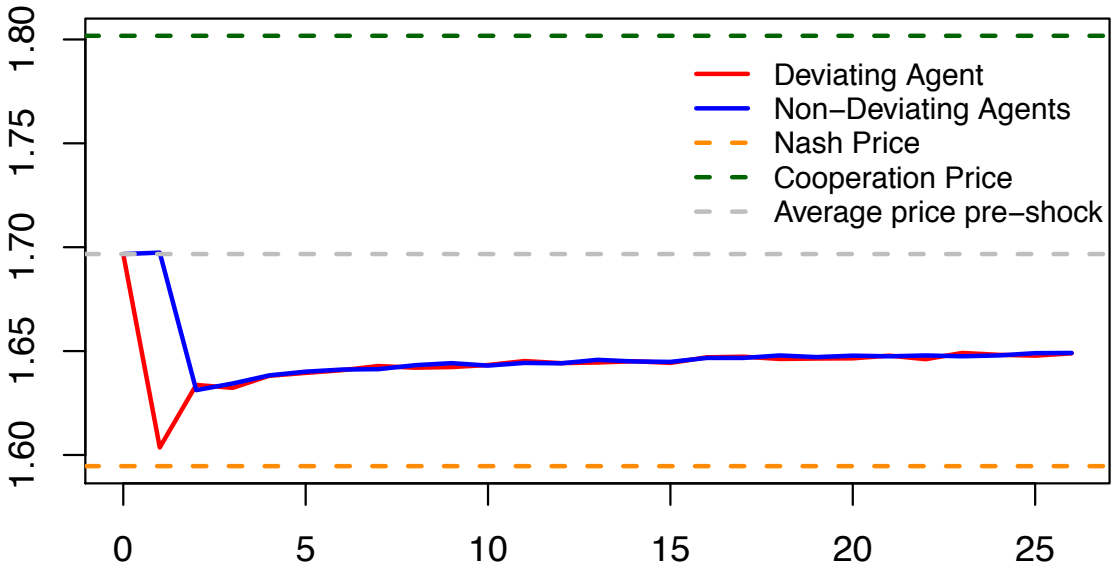

Figure 6: Price Impulse response with 3 players, $\alpha = 0.05, \beta = 8 \times 10^{-6}, \delta = 0.95$.

6.2. *Asymmetric firms*

The conventional wisdom has it that asymmetry impedes collusion. For one thing, asymmetric firms have different incentive compatibility constraints. This mechanical effect by itself tends to restrict the set of parameter values where the constraints are satisfied and thus supports the assumption that collusion is less likely when firms are asymmetric. More to the point, firms contemplating a tacit collusive agreement must solve a two-fold problem of coordination: they must choose both the average price level, which determines the aggregate profit, and the relative prices of the two firms, which determine how the total profit is split. Achieving coordination on both issues without explicit communication is generally regarded as a daunting task.

To see how Q-learning algorithms cope with these problems, we considered both cost and demand asymmetries of different degrees. The results are not sensitive to the source of the asymmetry, so Table 1 reports only the case of cost asymmetry.

c_2	1	0.875	0.750	0.625	0.5	0.25
2's market share	50%	55%	60%	64%	67%	73%
Δ	80%	78%	74%	69%	65%	54%
$\frac{\pi_1/\pi_1^N}{\pi_2/\pi_2^N}$	1	1.001	1.003	1.006	1.008	1.016

Table 1: Cost asymmetry ($c_1 = 1$).

As the table shows, asymmetry does reduce the average profit gain, but only to a limited extent. For example, in a variant of the baseline model where the more efficient firm (firm 2) has a 12.5% cost advantage (that is, $c_1 = 1$ and $c_2 = 0.875$), the average profit gain falls from 80% to 78%. With a 25% cost advantage, the average profit gain is 74%, and even with a 50% advantage it is still 65%.

In part the decrease in the average profit gain is simply a consequence of the absence of side payments. To see why this is so, consider how the two algorithms divide the aggregate profit. Remarkably, the ratio $\frac{\overline{\pi_i}}{\pi_i^N}$ is roughly the same for both firms, irrespective of the degree of asymmetry. That is, the gain from collusion is split among the firms roughly proportionally to their respective profit shares in the competitive equilibrium.

This "equitable" division is striking, as the notion of fairness is clearly not incorporated into the algorithms, but just as clearly does have an impact on the joint profit level. The maximization of joint profit indeed requires that the more efficient firm expand and the less efficient one contract relative to the Bertrand-Nash equilibrium.[42] However, this would produce a division strongly biased in favor of the more efficient firm. Conversely, a proportional division of the gain entails a cost in terms of the total profit.

This by itself explains why the average profit gain decreases as the degree of asymmetry increases. In other words, it seems that asymmetry doesn't actually make the coordination problem tougher for the algorithms but simply leads them to coordinate on a solution that does not maximize total profit.

[42]This effect may be so pronounced that the less efficient firm may actually get less profit under joint profit maximization than in the Bertrand-Nash equilibrium. This is why the level of the individual profit gains Δ_i given asymmetry needs to be interpreted with caution.

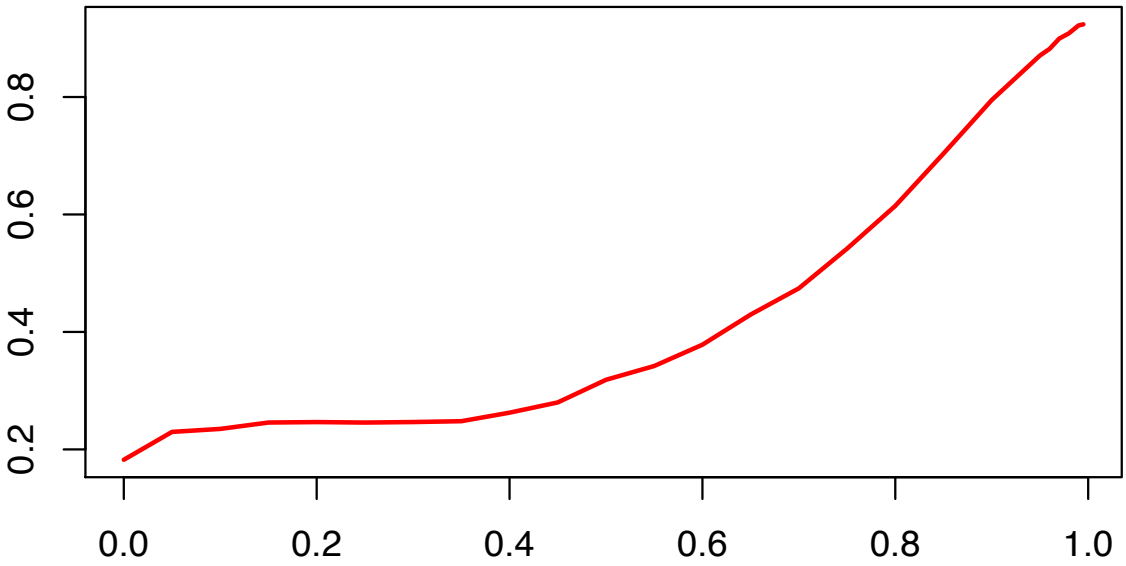

Figure 7: Average Profit Gain Δ for $\alpha = 0.05, \beta = 8 \times 10^{-6}$

6.3. *Patience*

The literature on infinitely repeated games has focused mostly on the discount factor as a critical factor for cooperation. The consensus is that lower discount factors, i.e. less patient players or else less frequent interaction, impede collusion.[43] The initial concern over algorithmic pricing was simply that algorithms may adjust the price much more quickly than humans, thereby increasing the frequency of interaction and hence the implicit discount factor δ.

Figure 7 shows how the average profit gain depends on δ. The theoretical postulate is largely supported by our simulations: collusion increases smoothly as the discount factor rises.

Given the frequency with which algorithms can adjust prices (often a matter of minutes), the baseline value of δ, i.e. $\delta = 0.95$, may actually seem quite low. For any realistic value of the interest rate, with algorithmic pricing δ is bound to be so close to 1 that impatience is no longer a factor that could actually impede collusion.[44]

[43]One exception is Sannikov and Skrzypacz (2007), who however posit a setting with imperfect monitoring. With more frequent interaction, firms respond more quickly to noise signals, and this may disrupt collusion.

[44]When δ gets very close to 1 Q-learning ceases to work well: we observe less convergence, less

6.4. *Stochastic demand*

In the baseline model, the economic environment is deterministic. The only source of uncertainty for a player is rivals' experimenting and learning. It is often argued that tacit collusion is harder to sustain when market conditions vary stochastically.

To assess the algorithms' ability to cope with uncertainty, we let the aggregate demand parameter a_0 vary stochastically. Specifically, a_0 is assumed to be an i.i.d. binary random variable that takes two values, $a_0^H = (1 + x)$ and $a_0^L = (1 - x)$ with equal probability. This corresponds to positive and negative demand shocks of $x\%$ of the average demand. The shocks are purely idiosyncratic and have no persistency – perhaps the most challenging situation for the algorithms.

When $x = 5\%$, the average profit gain under uncertainty falls from 80% to 78%; and even when $x = 10\%$ (so demand is 20% higher in good than in bad states) the average profit gain is still 72%. Apparently, then, demand variability does hinder collusion among firms, as one would have expected, but it does not eliminate it.

6.5. *Variable market structure*

In our baseline experiments, all firms are active at all times. To analyze the impact of a variable market structure, we repeat the exercise with one firm (the "outsider") entering and exiting the market in random fashion. This exercise is performed both for the case of two players (the market thus alternating between monopoly and duopoly) and of three players (duopoly and triopoly).

We take entry and exit to be serially correlated. Formally, let $\mathbb{I}_t$ be an indicator function equal to 1 if the outsider is in the market in period t and to 0 otherwise. We set

$$\text{prob}\{\mathbb{I}_t = 1|\mathbb{I}_{t-1} = 0\} = \text{prob}\{\mathbb{I}_t = 0|\mathbb{I}_{t-1} = 1\} = \rho. \tag{10}$$

This implies that the unconditional probability of the outsider's being in at some random time is 50%. Equivalently, the market is a duopoly half the time on average.

equilibrium play, and also smaller profit gains. This failure of Q-learning algorithms for $\delta \approx 1$ is well documented in the computer science literature. The problem seems to be that in the learning equation (4) the past Q-value is weighted too heavily by comparison compared with the new information π_t. This is so even if α is taken to be close to 1.

The degree of serial correlation ρ is set at 99%, so that when the outsider enters, it stays in the market for an average of 100 periods. Since in marketplaces where algorithmic pricing is commonly adopted periods can be very short, this level of persistency can be considered low.[45] We accordingly also run simulations setting this probability at 99.9%, meaning that when the outsider enters, it stays in for a longer period.

The state s now includes the prices of the previous period if all firms were active, or the prices of the active firms and the fact that the outsider was not active.

We find that the average profit gain can be even greater than with a fixed number of firms. In the duopoly/triopoly model, it is 76% when $\rho = 0.99$, and 86% when $\rho = 0.999$. By way of comparison, the average of the profit gain under duopoly is 80% and under triopoly it is 77%.

Figure 8 shows the impulse-response function for the case when the market alternates between duopoly and triopoly with $\rho = 0.99$. Punishment strategies closely resemble those obtained in the case of two firms. It seems that in periods of duopoly the two active firms learn the sophisticated punishment strategy with a reversion to the pre-deviation prices and that the third firm, when it enters, somehow manages to imitate its rivals' behavior. This learning pattern also emerges in a related extension (see subsection 6.10 below).

6.6. *Memory*

As we have seen, the assumption of one-period memory is less restrictive than it might seem, but at all events we have also run experiments with two-period memory ($k = 2$). One might presume that longer memory allows more flexible strategies and hence facilitates collusion, but in this experiment, perhaps surprisingly, average profit gain actually decreases, from 80% to 76%. The effect is small, to be sure, but its sign is the opposite of what one would perhaps expect.

One possible explanation is that longer memory makes learning harder by increasing the size of the Q-matrix, as is confirmed by the fact that the punishment strategies now become similar to those with three or four firms.[46]

[45] If a period were a quarter of an hour, say, then 100 periods would be barely one day.

[46] The matrix is exactly the same size with two firms and a two-period memory as with four

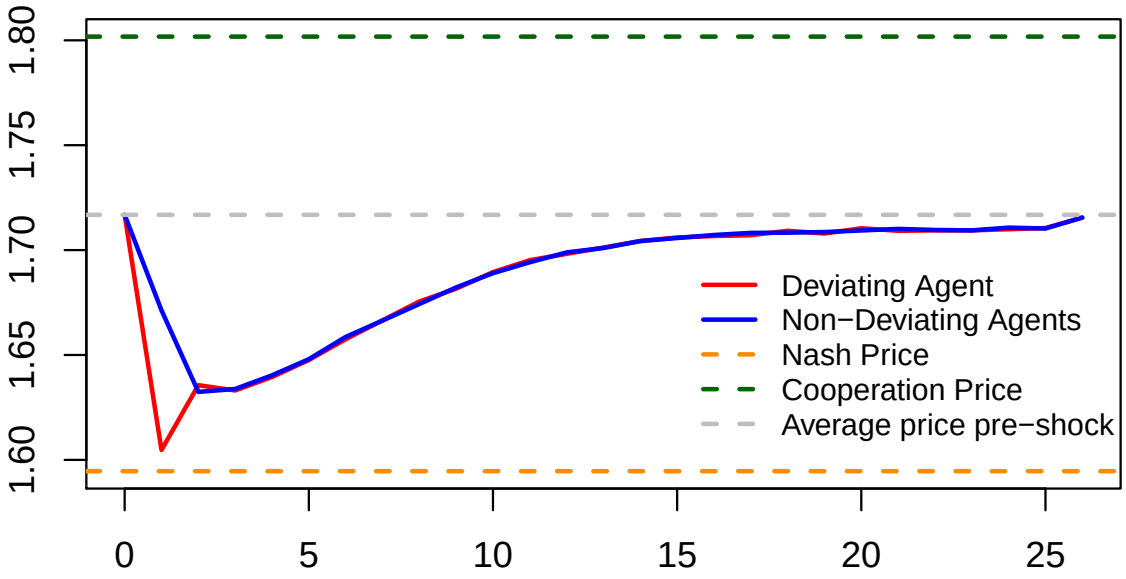

Figure 8: Price Impulse response with alternating entry and exit of a third player, $\alpha = 0.05, \beta = 8 \times 10^{-6}, \delta = 0.95$.

6.7. *Product substitutability*

In the logit model, a decrease in μ means that individual demand becomes more price-sensitive. That is, the reduction in μ captures an increase in product substitutability. In principle, the impact of changes in substitutability on the likelihood of collusion is ambiguous: on the one hand, when products are more substitutable the gain from deviation increases, but at the same time punishment can be harsher. This ambiguity is confirmed by the theoretical literature.[47]

In our setting, we test the consequences of lowering parameter μ from 0.5 (baseline) to 0.25 and to 0.1. The effect on average profit gain is not statistically significant: Δ remains more or less constant around 80%. This outcome is consistent with the theoretical predictions.

firms and a one-period memory.

[47]For example, Tyagi (1999) shows that greater substitutability hinders collusion when demand is linear or concave but may either hinder or facilitate it when demand is convex.

6.8. *Linear demand*

We repeated the analysis for the case of duopoly with linear demand

$$q_{it} = 1 - p_{it} - \gamma p_{jt}, \tag{11}$$

for various values of the horizontal differentiation parameter γ. We find average profit gains between 75% and 80%, and impulse-response functions almost identical to those observed with logit demand. Other alternative demands could also be tested, of course, but it would seem that the results are robust to the demand specification.

6.9. *Finer discretization*

As Section 3 notes, one reason why the repeated prisoner dilemma may be a misleading approach to the analysis of collusion is that coordination is much easier when there are only few possible actions. With two actions only, for instance, there is basically only one way to cooperate. As the number of actions and hence strategies increases, the strategy on which the players should coordinate is no longer self-evident. Failing explicit communication, this could engender misunderstandings and prevent cooperation.

To determine whether a richer strategy space facilitates or impedes cooperation, we run experiments with 50 or 100 feasible prices, not just 15. The average profit gain decreases slightly, but with $m = 100$ it is still a substantial 76%. We conjecture that Δ decreases because a larger number of actions and states necessitates more exploration to achieve the same amount of learning as in baseline model.[48]

We have also considered the case of parameter ξ higher than 10%, but greater flexibility in price setting - below Bertrand-Nash or above monopoly - turns out to be immaterial. This is not surprising, given that the players never converge on these very low or very high prices.

[48]This is consistent with the fact that the punishments (i.e. impulse responses) become similar to the other cases with larger Q matrices, that is when there are more firms and when the memory is longer.

6.10. *Asymmetric learning*

Going back to the baseline environment, we consider the case in which the two algorithms have different learning rates α, or different intensity of experimentation. Collusion appears to be robust to these changes. For example, when α_2 is set at 0.025 or 0.1 with α_1 constant at 0.05, the average profit gain is 75% or 80%, respectively. In both cases, the firm with lower α gains more, suggesting that setting $\alpha = 0.05$ still gives too much weight to new information.

We also halved and doubled the value of β for one algorithm only, keeping the other fixed at $\beta = 8 \times 10^{-6}$. In the former case, the value of Δ stays practically constant, in the latter it comes marginally down to 78%. In both cases, the algorithm that explores less performs slightly better than the rival.

Finally, we consider a case in which when two algorithms have completed their learning a third firm enters the market. Naturally, the new entrant has to experiment and learn, while the incumbents cease to experiment but may keep adjusting their Q-matrices (because they continue to learn). Now the average profit gain is 74%, exactly the same as in the model with three firms all learning simultaneously. What is more, the entrant's gain is similar to the incumbents'. It seems the entrant cannot exploit the incumbents even if they have stopped experimenting.

These highly preliminary, incomplete results for situations in which different algorithms follow different learning and exploration strategies suggest that diversity among the algorithms does not significantly affect the degree of collusion.

7. CONCLUSION

We have shown that in stationary environments Q-learning pricing algorithms systematically learn to collude. Collusion tends to be partial and is sustained by punishment in case of defection. The punishment is proportional to the extent of the deviation and is of finite duration, with a gradual return to pre-deviation prices. The algorithms learn to play these strategies by trial and error, requiring no prior knowledge of the environment in which they operate. They leave no trace whatever of concerted action: they do not communicate with one another, nor have they been designed or instructed to collude.

From the standpoint of competition policy, these findings should clearly be wor-

rying. They suggest that with the advent of Artificial Intelligence and algorithmic pricing tacit collusion may become more prevalent, heightening the risk that tolerant antitrust policy may produce too many false negatives and so possibly calling for policy adjustments. Before altering policy, however, more research is needed.

We conclude with a brief discussion of three major unsettled issues. The first is the complexity of the economic environment. We have considered a good many extensions of the baseline model, but all separately, the model thus remaining quite highly stylized. To produce a more realistic setting for analysis, one might well posit a model with several firms, longer memory, stochastic demand and possibly also structural breaks. Analyzing such a model would not appear to be unfeasible: even in the more complex settings set out in Section 6, for instance, our algorithms achieve convergence in a matter of minutes, which suggests that we are still far from having exhausted their learning capacity.[49] In order to learn to price in more complex environments, however, tabular Q-learning algorithms need to experiment much more extensively than we have allowed them to do. This may result in a failure to converge, as one algorithm's experimentation may disrupt others' learning. It seems therefore that more complex environments should be analyzed for algorithms whose learning is more efficient than tabular Q-learning - say, with deep learning algorithms.

This leads to the second, related issue. Not only can deep learning manage more complex environments, it can also speed the learning process. This is important, because the training of the algorithms cannot always be conducted entirely off-line, and in the short run experimentation is costly. On-line learning seems necessary, in particular, when the economic environment is changeable, or when the training environment does not exactly reflect the reality of the markets where the algorithms are eventually deployed. As noted, these first two issues can be addressed by using algorithms that are smarter than tabular Q-learners. The consensus choice for this purpose, at the present, seems to be deep learning algorithms. Thus extending our analysis to this type of algorithm is an important task for future work.

A third, more general issue is the external validity of the experimental analysis. While Q-learning is natural, common and well understood, there are also other

[49]It bears repeating that training our algorithms takes much less time than training algorithms designed to solve complex tasks.

forms of Reinforcement Learning. Moreover, Q-learning algorithms themselves come in different varieties, with alternative exploration strategies and estimation methods. We do not know exactly what algorithms are used in practice, and even if we knew what firms do now, things are likely to change rapidly.

One possible approach would be systematic replication of our analysis on other existing pricing algorithms, as well as on those that will be introduced in the future. Harrington (2017) advocates precisely this kind of regulatory approach to algorithmic pricing. He envisions a "sandbox" regulatory model similar to that currently used for authorizing new drugs, in which new algorithms would be tested for collusion-proneness and rejected if they display excessive propensity to collude. From this perspective, our work may be seen as part of a preliminary effort to determine whether such pervasive regulation is really necessary, and to develop the methods most suitable for such tests.

More broadly, however, we feel that there may be space for theoretical analysis that abstracts from the details of particular algorithms and essentially considers their common architecture. Perhaps the most striking feature of our Q-learning algorithms is that they do not behave erratically but appear to follow very systematic patterns. Experimental analysis may help reveal these patterns, but ultimately theory will be needed in order to provide deeper explanations.

REFERENCES

[1] Barlo, M., Carmona, G. and Sabourian, H. (2016). Bounded memory Folk Theorem. *Journal of economic theory*, **163**, 728–774.

[2] Bergemann, D. and Valimaki, J. (2006). Bandit Problems.

[3] Calvano, E., Calzolari, G., Denicolò, V. and Pastorello, S. (2018). Q-Learning to Cooperate.

[4] 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*, WWW '16, Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, pp. 1339–1349.

[5] Cooper, D. J. and Kühn, K.-U. (2014). Communication, Renegotiation, and the Scope for Collusion. *American Economic Journal: Microeconomics*, **6** (2), 247–278.

[6] Dal Bó, P. and Fréchette, G. R. (2018). On the Determinants of Cooperation in Infinitely Repeated Games: A Survey. *Journal of economic literature*, **56** (1), 60–114.

[7] Erev, I. and Roth, A. E. (1998). Predicting How People Play Games: Reinforcement Learning in Experimental Games with Unique, Mixed Strategy Equilibria. *The American economic review*, **88** (4), 848–881.

[8] — and — (2014). Maximization, learning, and economic behavior. *Proceedings of the National Academy of Sciences of the United States of America*, **111 Suppl 3**, 10818–10825.

[9] Ezrachi, A. and Stucke, M. E. (2016). Virtual Competition. *Journal of European Competition Law & Practice*, **7** (9), 585–586.

[10] — and — (2017). Artificial intelligence & collusion: When computers inhibit competition. *University of Illinois law review*, p. 1775.

[11] Fudenberg, D. and Levine, D. K. (2016). Whither Game Theory? Towards a Theory of Learning in Games. *The journal of economic perspectives: a journal of the American Economic Association*, **30** (4), 151–170.

[12] Greenwald, A., Hall, K. and Serrano, R. (2003). Correlated Q-learning. In *ICML*, aaai.org, vol. 3, pp. 242–249.

[13] Greenwald, A. R., Kephart, J. O. and Tesauro, G. J. (1999). Strategic Pricebot Dynamics. In *Proceedings of the 1st ACM Conference on Electronic Commerce*, EC '99, New York, NY, USA: ACM, pp. 58–67.

[14] Harrington, J. E., Jr (2017). Developing Competition Law for Collusion by Autonomous Price-Setting Agents.

[15] Ho, T. H., Camerer, C. F. and Chong, J.-K. (2007). Self-tuning experience weighted attraction learning in games. *Journal of economic theory*, **133** (1), 177–198.

[16] Hu, J., Wellman, M. P. and Others (1998). Multiagent reinforcement learning: theoretical framework and an algorithm. In *ICML*, vol. 98, pp. 242–250.

[17] Huck, S., Normann, H.-T. and Oechssler, J. (2004). Two are few and four are many: number effects in experimental oligopolies. *Journal of economic behavior & organization*, **53** (4), 435–446.

[18] Klein, T. (2018). Assessing Autonomous Algorithmic Collusion: Q-Learning Under Short-Run Price Commitments.

[19] Könönen, V. (2006). Dynamic pricing based on asymmetric multiagent reinforcement learning. *International Journal of Intelligent Systems*, **21** (1), 73–98.

[20] Kühn, K.-U. and Tadelis, S. (2018). The Economics of Algorithmic Pricing: Is collusion really inevitable?

[21] Leufkens, K. and Peeters, R. (2011). Price dynamics and collusion under short-run price commitments. *International Journal of Industrial Organization*, **29** (1), 134–153.

[22] 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*, **56** (3), 571–599.

[23] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S. and Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, **518** (7540), 529–533.

[24] Roth, A. E. and Erev, I. (1995). Learning in extensive-form games: Experimental data and simple dynamic models in the intermediate term. *Games and economic behavior*, **8** (1), 164–212.

[25] Salcedo, B. (2015). Pricing Algorithms and Tacit Collusion. *Manuscript, Pennsylvania State University*.

[26] Sannikov, Y. and Skrzypacz, A. (2007). Impossibility of Collusion under Imperfect Monitoring with Flexible Production. *The American economic review*, **97** (5), 1794–1823.

[27] Shoham, Y., Powers, R., Grenager, T. and Others (2007). If multi-agent learning is the answer, what is the question? *Artificial intelligence*, **171** (7), 365–377.

[28] Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., Chen, Y., Lillicrap, T., Hui, F., Sifre, L., van den Driessche, G., Graepel, T. and Hassabis, D. (2017). Mastering the game of Go without human knowledge. *Nature*, **550** (7676), 354–359.

[29] Singh, N. and Vives, X. (1984). Price and quantity competition in a differentiated duopoly. *The Rand journal of economics*, pp. 546–554.

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

[31] Tesauro, G. and Kephart, J. O. (2002). Pricing in Agent Economies Using Multi-Agent Q-Learning. *Autonomous agents and multi-agent systems*, **5** (3), 289–304.

[32] Tuyls, K., Hoen, P. J. T. and Vanschoenwinkel, B. (2006). An Evolutionary Dynamical Analysis of Multi-Agent Learning in Iterated Games. *Autonomous agents and multi-agent systems*, **12** (1), 115–153.

[33] Tyagi, R. K. (1999). On the relationship between product substitutability and tacit collusion. *Managerial and Decision Economics*, **20** (6), 293–298.

[34] Waltman, L. and Kaymak, U. (2008). Q-learning agents in a Cournot oligopoly model. *Journal of economic dynamics & control*, **32** (10), 3275–3293.

[35] Watkins, C. J. C. H. (1989). *Learning from delayed rewards*. Ph.D. thesis, King's College, Cambridge.

[36] — and Dayan, P. (1992). Q-learning. *Machine learning*, **8** (3), 279–292.