Adaptive Algorithms and Collusion via Coupling*

Martino Banchio Giacomo Mantegazza†

Stanford Graduate School of Business

November 7, 2022

Click here for the latest version

Abstract

We develop a theoretical model to study strategic interactions between adaptive learning algorithms. Applying continuous-time techniques, we uncover the mechanism responsible for collusion between Artificial Intelligence algorithms documented by recent experimental evidence. We show that spontaneous coupling between the algorithms' estimates leads to periodic coordination on actions that are more profitable than static Nash equilibria. We provide a sufficient condition under which this coupling is guaranteed to disappear, and algorithms learn to play undominated strategies. We apply our results to interpret and complement experimental findings in the literature, and to the design of *learning-robust strategy-proof mechanisms*. We show that ex-post feedback provision guarantees robustness to the presence of learning agents. We fully characterize the optimal learning-robust mechanisms: they are menu mechanisms.

Keywords: Artificial Intelligence, Learning, Mechanism Design, Continuous Time
JEL Classification Codes: C62, D47, D83, L51

*We are indebted to our advisors Michael Ostrovsky, David Kreps, Kostas Bimpikis, and in particular Andrzej Skrzypacz for invaluable guidance and support. We also want to thank Susan Athey, Anirudha Balasubramanian, Lanier Benkard, Emilio Calvano, Daniel Chen, Peter DeMarzo, Andreas Haupt, Ravi Jadageesan, Irene Lo, Alexander MacKay, Suraj Malladi, Paul Milgrom, Ilan Morgenstern, Evan Munro, Ilya Segal, Takuo Sugaya, Stefan Wager, Larry Wein, Gabriel Weintraub, Kuang Xu, Frank Yang, and seminar participants at Stanford University, Harvard University, IIOC, and SITE Conference for their insightful comments.

†Email: mbanchio@stanford.edu, giacomom@stanford.edu

1 Introduction

In this paper, we develop analytical tools to study the outcomes of games played by learning algorithms. Because the evolution of play is discrete and possibly stochastic, we first show how to compute fluid approximations of the learning paths by such algorithms. Then, we use these approximations to explain why such algorithms may not converge to playing Nash equilibrium and even fail to learn to play dominant strategies. In a Prisoner's Dilemma, algorithms such as ε-greedy Q-learning (not designed to learn dynamic reward strategies such as tit-for-tat) can get to stochastic cycles exhibiting high cooperation rates. We isolate the mechanism responsible for such cooperation (unequal learning rates) and characterize a class of learning algorithms that are guaranteed to learn to play only undominated strategies.

We then apply our findings. First, we show how a market designer can ensure that even simple algorithms learn to play the equilibrium strategy over time. The designer can create feedback policies that help algorithms learn. In other words, we show how to extend the design of strategy-proof mechanisms to be robust to players using learning algorithms to determine their optimal strategy. We do so by characterizing the policies which communicate minimal feedback. Second, we use our theoretical results to provide new insights about the findings of two recent papers that show via discrete-time simulations that Q-learning algorithms sometimes learn to play equilibrium and sometimes learn to collude.

Our study of strategic interactions of learning algorithms is motivated by the increasing adoption of algorithms in business contexts, including automated bidding in online ad auctions, pricing in online shopping platforms, and setting rents for short-term and long-term rentals, among others. However, this increase has been accompanied by concerns voiced by scholars and practitioners alike that automated pricing and bidding software could facilitate collusion. Motivated by these concerns, various national agencies have begun studying these phenomena to regulate and tune online and offline markets (OECD (2017), Competition Bureau (2018)).

We contribute to the analysis of algorithmic collusion by developing a theoretical framework for analyzing strategic interactions between algorithms. Using fluid limit techniques, we approximate the evolution of algorithms driven by discontinuous laws of motion. We focus on a class of algorithms that we call separable reinforcers. That class includes Q-learning (the building block of many AI algorithms) and tracks a vector of values representing each action's payoff consequences. All algorithms in this class estimate such vectors by repeatedly interacting with the environment and updating the vector's

entries based on the observed payoff flows. A policy function determines what action the agent takes, given the estimates of the algorithm. For example, an ε-greedy policy selects the action that currently maximizes the expected payoff of the agent with probability 1-ε, and with probability ε explores the action space uniformly at random.

We start by analyzing the play of ε-greedy Q-learning algorithms in Prisoner's Dilemma. The fluid limit defines a four-dimensional system of differential equations that characterizes the evolution of the two algorithms playing this game. The dynamical system reveals the source of collusive behavior — it is a form of spontaneous "coupling" between independent algorithms that undermines the incentives for competition. Note that if opponents were to play a fixed profile of actions, the environment would be stationary from the algorithm's perspective, and the algorithm would learn to play its best response. However, when both players employ such algorithms, the environment they face is not stationary, and their learning becomes correlated. As a result, even though each algorithm experiments independently, the actions of one affect estimates of the other, and they end up playing symmetric profiles of actions most of the time. Correlated estimates limit their ability to evaluate profitable deviations. We show that coupling leads to the continuous counterpart of stochastic cycles, in which the agents cooperate most but not all of the time.[1].

Intuitively, during a period of joint cooperation, each agent's estimates of cooperation are correct as long as the opponent is also cooperating. Experimenting with defection reveals defection's profitability over time, but once one agent begins defecting, the opponent observes that cooperation became less profitable and will switch to defection too. Mutual defection reduces the estimate of the value of defection quickly. Instead, the estimate of cooperation is more persistent, luring the algorithms away from defection. We dub this phenomenon "spontaneous coupling," highlighting how nearly simultaneous deviations sustain dominated outcomes.

The coupling of epsilon-greedy Q-learning originates from a characteristic of the algorithm, its relative learning rates. The agent updates its estimates only for actions it takes; therefore, it is slow to learn about actions taken seldom. While epsilon-greedy guarantees continued experimentation and updating that never stops, updating is faster for those actions that currently have the largest estimate. Uneven learning rates are responsible for introducing additional persistence in the agent's estimates, thereby sustaining long periods of correlated play. By comparing it with simulations, we show that our approximation

[1] This explains why we observe that, while such algorithms often learn to collude in simulations, collusion is imperfect. It is because the agents cannot converge to a constant profile of actions that is not a pure Nash Equilibrium — otherwise, they would learn to best respond.

is a good predictor of the average behavior of the algorithms in discrete-time games: we observe similar coordination of play and similar levels of cooperation as we do in the theoretical limit.

In contrast, we show that algorithms with uniform learning rates, that is, algorithms that update estimates of the value of each action at the same speed, do not engage in collusive practices. Uniform learning guarantees that every action enjoys the same persistence. Algorithms with uniform learning avoid the stochastic cycles and learn to play only undominated strategies.

We apply this result to construct a theory of learning-robust mechanism design. The designer can help algorithms equalize learning rates by providing feedback and facilitating counterfactual inference. We extend our model to allow for games with private information (where agents privately know their type, such as a value in an auction). Optimal learning-robust mechanisms ask players to submit their reports (for example, bids in an auction) and select an outcome based on the profile of reports. After the outcome is selected, the designer provides information that allows every participant to compute counterfactuals. With this information, algorithms can update their estimates for all actions in every period. We show that the least amount of information the designer needs to provide is described by a menu, which communicates what outcome the agent would have obtained for each alternative report.

Finally, we show that experimental results found in the literature can be better understood with the help of our analytical framework. We apply our model to a recent paper by Asker et al. (2022). In a simulated Bertrand oligopoly, the authors find that feedback about the demand curve can affect long-term pricing by the algorithms. We similarly provide new insight into why algorithms learn to collude in first-price auctions and not in second-price auctions in the simulations studied in Banchio and Skrzypacz (2022). We also explain why a policy of revealing the lowest-bid-to-win, as introduced by Google in online display auctions at the time of their transition to first-price auctions,[2] increases competition by algorithms in Banchio and Skrzypacz (2022).

1.1 Literature Review

Q-learning and Artificial Intelligence have recently sparked interest in Economics, often through experimental work (see e.g. Klein (2021), Hansen et al. (2021), Banchio and Skrzypacz (2022)) or empirical work (see e.g. Musolff (2021), Assad et al. (2021)). The

[2] See https://blog.google/products/admanager/rolling-out-first-price-auctions-google-ad-manager-partners/.

work of Calvano et al. (2020) draws attention to strategies as a proxy for collusion: they argue that simply looking at outcomes of learning might be insufficient, as collusion might arise as a "mistake" by poorly designed algorithms. Our work allows to delve deeper into the dynamics of learning, and through comparative statics and convergence analysis determine whether collusion is systemic. Particularly relevant is the work by Asker et al. (2022), which analyzes the impact of algorithm design on collusion in a Bertrand pricing game, and by Banchio and Skrzypacz (2022), who find that additional feedback in first-price auctions restores competition. We contribute to these questions with analytical tools that allow us to generalize their intuition and prove that counterfactual information guarantees competition in the games they consider. Some papers analyze models of collusion between algorithms, for example Brown and MacKay (2021), Leisten (2022), and Lamba and Zhuk (2022). In these papers, algorithms choose prices based on the opponent's last quoted price: the strategies are Markov, which rules out many plausible learning strategies, including most AI algorithms. We focus on *learning* algorithms, which adapt their decisions along the whole history. Contributions from the bandit literature include Aouad and Van den Boer (2021), who prove tacit collusion schemes arise with classical multi-armed bandits algorithms, and Hansen et al. (2021), which finds supra-competitive prices are sustained by coordinated experiments when bidders use the Upper Confidence Bound algorithm. We prove our results for general classes of algorithms, allowing for heterogeneity in parameters as well as algorithms themselves.

Some work has examined more generally Reinforcement Learning in games, for example Erev and Roth (1998) or Mertikopoulos and Sandholm (2016), but with some notable differences. Firstly, many have analyzed systems experimentally (Erev et al. (1999), Lerer and Peysakhovich (2017)). Our approach is complementary: with the aid of our framework, one can tell apart experimental findings from agent design considerations. On the other hand, there is some theoretical work on convergence of learning procedures. For example, learning through reinforcement has been associated with evolutionary game theory by Börgers and Sarin (1997). Others have formally analyzed some of the simpler models, as Hopkins and Posch (2005). These results have then been extended to more complex systems, but with a focus on the connection with replicator dynamics as their core. Our approach is particularly relevant because instead it examines the workhorse model in applied work, ε-greedy Q-learning. We obtain a tool valuable for regulation and design, but we trade it off against the complete understanding possible in a simpler system. Conveniently, the approach described in Section 2 includes some earlier results under a general algorithmic structure.

Some earlier work analyzes continuous-time approximation of AI algorithms, mostly

in the single-agent setting. Related to ours is the work of Tuyls et al. (2005): the authors examine a continuous-time approximation of multi-agent Q-learning with Boltzmann exploration, and show a link with the Replicator Dynamics from the Evolutionary Game Theory (EGT) literature. Building on their work, Leonardos and Piliouras (2022) characterize the tradeoff between exploration and exploitation in the same setting. Our approximations and results hold in more general settings: we analyze a general class of adaptive algorithms, and our results leverage their discontinuities. Both Gomes and Kowalczyk (2009) and Wunder et al. (2010) propose a continuous-time approximation of Q-learning in a multi-agent setting with ε-greedy algorithms. Their approximations are mutually inconsistent and require additional conditions on the parameter. Most importantly, those approximations remain model-dependent. Our method applies to general adaptive algorithms. The result is a recipe to analyze equilibria through the lens of dynamical systems, abstaining from heuristic modeling choices. The work of Benaim (1996) often serves as a foundation for stochastic approximations in learning. With respect to our approach, those tools have a harder time handling general learning procedures, including AI techniques that fall under the umbrella of Temporal Difference Learning. Additionally, we adopt the formalism of differential inclusions to analyze points of non-differentiability, generally new to the theory of stochastic approximations.[3]

Fluid models and other forms of approximation have enjoyed great popularity in the fields of applied probability and operations research. Starting from the seminal contributions of Iglehart (1965), Kurtz (1970), Halfin and Whitt (1981) and Harrison and Reiman (1981), approximations enabled formal analysis of systems otherwise deemed intractable: see, e.g., Wein (1992) for a classic application to inventory management. More recently, Mitzenmacher (2001) applied fluid models to the analysis of parallel computing systems, while Wager and Xu (2021) employ diffusion approximations to study sequential experiments.

We contribute to the theory of implementation in the presence of boundedly-rational agents. Abreu and Matsushima (1992) provides implementability in dominance-solvable games, but the iterative deletion of dominated strategies requires mixing, which our setting does not allow. A few papers consider implementation in supermodular games, which could be adopted in our model (see Chen (2002) and Mathevet (2010)). Sandholm (2005) develops implementation in potential games, as this class guarantees valuable evolutionary properties. We suspect similar results would hold under our framework. All of these works provide robustness to certain types of bounded rationality and learning pro-

[3]One exception is the paper by Wunder et al. (2010), which however abandons this route in favor of simulations.

cedures. As we show, the robustness guaranteed by the game form may be insufficient — even the stronger strategy-proof incentives fail to guarantee implementation. We instead add a new lever, feedback provision, to the toolbox of the designer.

2 Model

We begin by describing a general class of algorithms we call *reinforcers*: they learn by reinforcing successful actions and penalizing unsuccessful ones. Because their dynamics tend to be complex, we approximate these algorithms in continuous time with ordinary differential equations (ODEs).

2.1 Games with Adaptive Agents

Consider a finite normal-form game $G = (N, (A_i)_{i\in N}, (r^i)_{i\in N})$ with N players. We are interested in what happens when agents play repeatedly in this game by choosing actions using a learning algorithm. For simplicity, let us look at one agent, Alice, choosing her actions from the finite set A_i with cardinality d_i. Instead of selecting an action herself, Alice delegates the decision-making to an algorithm that tries to learn how to maximize her utility. The function $r^i(a^i, a^{-i})$ describes her utility, which depends on her opponents' actions and her own.

For example, Alice might take actions as recommended by a well-known AI algorithm, ε-greedy Q-learning.

Example 1. *A ε-greedy Q-learning algorithm consists of a vector $Q(k)$ for every period k and a decision rule π_ε.*

- *Each entry $Q_a(k)$ is an estimate of the long-run value of action $a \in A$. The update for entry $Q_a(k+1)$ in period $k+1$ is given by*

$$Q_a(k+1) = \begin{cases} Q_a(k) + \alpha\left[r(k) + \gamma \max_{a'} Q_{a'}(k) - Q_a(k)\right] & \text{if } a = a(k) \\ Q_a(k) & \text{else.} \end{cases} \tag{1}$$

  where $r(k)$ is the payoff and $a(k)$ is the action the algorithm took in period k. $\gamma \in [0,1)$ is a discount factor, and $\alpha \in (0,1)$ is called learning rate.

- *Given the vector $Q(k)$, the algorithm takes actions according to a ε-greedy policy. The*

decision rule $\pi_\varepsilon : \mathbb{R}^{|A|} \to \Delta(A)$ *selects the following probability distribution over actions:*

$$\pi_\varepsilon(Q(k)) = \begin{cases} \dfrac{1}{|\operatorname{argmax}_{a \in A_i} \theta^a|} & \forall a \in \operatorname{argmax}_{a \in A_i} Q_a(k) \quad \textit{with probability } 1-\varepsilon \\ \frac{1}{d_i} & \forall a \in A_i \quad \textit{with probability } \varepsilon \end{cases}$$

The Q vector is a more general version of the Erev and Roth (1998) and Börgers and Sarin (1997) reinforcement learning models. We analyze in this work a different, more straightforward decision rule. The ε-greedy policy offers an intuitive recipe for resolving the tradeoff between exploration and exploitation. With probability $1-\varepsilon$ the algorithm is *greedy*: it selects the action corresponding to the largest entry of the Q vector. The algorithm exploits what he considers to be the best action at the time. With probability ε it explores the entire action space by taking an action at random.

Intuitively, the Q-vector estimates the long-run value of actions by iterating over a Bellman equation. In period k, the value of an action at a certain state is a convex combination of its previous estimate (with weight $1-\alpha$) and a new Bellman estimate (with weight α). The weight α serves as a learning rate, and we can interpret it as a measure of persistence: higher αs make the algorithm faster in forgetting the past.

Notice that ε-greedy Q-learning, when adopted as a decision rule in a game, is inherently misspecified. This is because $Q_a(k)$ attemps to estimate the payoff from taking action a today and then playing optimally from tomorrow onwards, denoted by $r^i(a, a^{-i}) + \gamma \max_{a'} r^i(a', a^{-i})$. However, both terms depend on the unobserved actions of the opponents, a^{-i}. If the opponents' profile of strategies was fixed, Q-learning would converge on the best response to that profile; instead, when all agents learn there are no results that guarantee convergence.

Most Q-learning-based methods belong to a larger class of adaptive algorithms that we name reinforcers.

Definition 1. A *reinforcer* for agent i is a pair (θ^i, π^i) consisting of

- A d_i-dimensional stochastic process θ^i that evolves according to

$$\theta^i(k+1) = \theta^i(k) + \alpha^i T^i(a^i(k), r^i(k), \theta^i(k)),$$

  where $a^i(k) \in A_i$ is the action taken by agent i in period k, $\theta^i \in K \subset \mathbb{R}^{d_i}$, and $\alpha^i \in \mathbb{R}_+^{d_i}$ are learning rates for all actions $a^i \in A_i$.

- A *policy* π^i, that is a map $\pi^i : K \to \Delta(A_i)$, which selects a distribution of actions for each value of the process $\theta^i \in K$.

A reinforcer carries a statistic for each available action and selects an action in period k according to its policy. Both agent i's and her opponents' policies introduce randomness in the process θ^i. The update function T^i depends Alice's realized actions directly and on her opponents' actions through her utility. To completely define a reinforcer, we need to specify also an initial value of $\theta^i(0)$, which we call the *initialization* of θ^i. We maintain the following assumption on the update function T^i:

Assumption A1. The functions $T^i(a^i, r, \theta)$ and $\pi^i(\theta)$ are Lipschitz-continuous almost-everywhere in θ.

Let us return to Example 1. The policy π_ε is constant on the subspace of $\mathbb{R}^{d_i}$ where $\operatorname{argmax}_a \theta^i_a$ is fixed and unique. The argmax changes only along the lines of the form $\left\{\theta^i | \theta^i_a = \theta^i_{a'} = \max \theta^i\right\}$, which have zero Lebesgue measure. The discontinuities of the update function of Q lie on the same lines, and thus the update is also a.e. Lipschitz. The ε-greedy Q-learning satisfies Assumption A1.

When multiple agents employ reinforcers, we represent the system as a single vector of dimension $d_1 + \cdots + d_N$ by stacking the individual algorithms, denoted by $\theta(k) = (\theta^1(k), \ldots, \theta^N(k))$. Similarly, T and π indicate the collection of update functions and policies when missing a superscript.

2.2 Approximation in Continuous Time

We are interested in describing the dynamics of learning for a given reinforcer. A common feature of reinforcers is that while relatively simple to analyze in a stationary environment, they often become unpredictable when learning to play against each other. For example, Q-learning is known to converge in stationary, single-decision-maker environments. In strategic settings, if all opponent's actions were constant, ε-greedy Q-learning would learn how to best respond. However, when opponents are themselves non-stationary, its properties are all but well understood. Analyzing the learning path of any number of Q-learning agents requires describing discrete, possibly stochastic updates and how those interact over time.

This section proposes a continuous-time approach to deal with the two former difficulties. We adopt the formalism of fluid approximations first introduced by Kurtz (1970). The idea behind fluid approximations is to analyze the limit of the systems as the jumps get small and their frequency increases. The limiting fluid ODE system provides tractability and (often) closed-form solutions.

The first step in formalizing the approximation consists of *Poissonizing* the stochastic vector θ associated with a reinforcer. That is, we consider a Poisson clock with rate $\lambda^1 = 1$.

On average, the clock ticks once in a unit of time. At every tick of the Poisson clock, the now continuous process $\theta^1(t)$ gets updated according to the vector function T. The Poissonized process $\theta^1(t)$ is a compound Poisson process. We introduced a new layer of randomness, but the path traced by $\theta^1(t)$ remains identical to the path of the discrete $\theta(t)$.

We can now generate a sequence of processes $(\theta^n)_{n\in\mathbb{N}}$ by increasing the rate of the Poisson clock to $\lambda^n = n$. The goal is to regularize the learning dynamics; therefore, we need to compensate for frequent updates by reducing the contribution of each jump to the total estimate. We do this by dividing each jump by $\frac{1}{n}$: in the parlance of Definition 1, at a given arrival time τ of the Poisson process,

$$\theta(\tau) = \theta(\tau^-) + \frac{\alpha}{n} T(a(\tau^-), r_{\tau^-}, \theta(\tau^-).$$

Each process θ^n has the same infinitesimal generator: the sequence $(\theta^n)_{n\in\mathbb{N}}$ preserves the instantaneous rate of change uniformly. We can prove the following result:

Theorem 1. *Let $H \subset K$ be such that T and π are Lipschitz over H, and let $y_0 \in H$ be the initialization point of θ. Then, the sequence of continuous-time stochastic processes $(\theta^n)_{n\in\mathbb{N}}$ converges in probability to the solution of the following Cauchy problem:*

$$\begin{cases} \frac{d\Theta^i(t)}{dt} = \alpha \mathbb{E}_{\pi^i, \pi^{-i}} \left[T^i(a^i, r(a^i, a^{-i}), \Theta^i(t)) \right] \\ \Theta^i(0) = y_0^i \end{cases}$$

for all i. That is, $\lim_{n\to\infty} P\left\{ \sup_{t\leq T} \left\| \theta^n(t) - \Theta(t) \right\| > \eta \right\} = 0$ for all $T \geq 0$ and $\eta > 0$ such that $\{\Theta(t)\}_{t\leq T} \subset H$.

We provide a formal construction of the sequence θ^n and a proof of this result in Appendix A. The process $\Theta(t)$ for $t \geq 0$ is the fluid approximation to θ: it is a deterministic dynamical system whose time-derivative is the expected update that the discrete process θ^k would incur over one unit of time. Theorem 1 guarantees that the sequence of processes $(\theta^n)_{n\in\mathbb{N}}$ we constructed draws closer and closer (in probability) to the continuous process. The theorem relies on a law-of-large-numbers argument: if the updates occur with high frequency and each update's contribution is relatively small, the process behaves similarly to its expectation. We leverage the fundamental theorem of fluid approximations by Kurtz (1970), a stochastic-processes version of the law of large numbers, to conclude convergence in probability.

2.3 Continuous-Time Dynamical Systems

Instead of analyzing the discrete reinforcers, we focus on their continuous-time fluid limits. Their dynamical system structure provides us with a wide range of tools to characterize their path over time.

Definition 2. Given a dynamical system $\frac{d\theta}{dt}$, the *flow* of θ starting in x is the map

$$\begin{array}{rccc} \theta(-,x)\colon & [0,+\infty) & \to & K \\ & t & \mapsto & \theta(t,x) \end{array}$$

that satisfies Equation (2) with initial condition $\theta(0,x) = x$. A *trajectory* of the dynamical system is the graph of the flow, $\left\{(t,\theta(t,x))\colon\ t \in [0,+\infty)\right\}$. The set $\Gamma_x = \left\{\theta(t,x)\colon\ t \in [0,+\infty)\right\}$ is the *orbit* of θ starting from x. If a sequence $(t_n)_{n\in\mathbb{N}}$ is such that

$$\begin{cases} \lim_{n\to\infty} t_n = +\infty \\ \lim_{n\to\infty} \theta(t_n,x) = y \end{cases}$$

we say that y belongs to the *forward limit set* for the orbit Γ_x. A *steady state* of the dynamical system is a fixed point of its law of motion, i.e. $\frac{d\theta}{dt}(t) = 0$.

The actions taken by a reinforcer depend directly on its estimates. Thus, analyzing the underlying dynamical system is a good proxy for the path of play. Studying the forward limit set of a reinforcer allows us to focus on estimated values instead of realized actions. A policy such as ε-greedy trembles organically: even if the reinforcer settles its estimates and identifies the action with the largest expected reward, it will keep playing sub-optimal actions (albeit with a small probability). The reinforcer can nonetheless become stationary if its estimates reach a steady state. We adopt this view for simplicity and clarity of exposition.

Definition 3. We say a reinforcer (θ^i,π^i) initialized at x *converges* if $\theta^i(t)$'s flow started at x converges on a steady-state $\overline{\theta}^i$. We say the agent *converges on action* a_{ss} if the steady-state $\overline{\theta}^i$ is such that $a_{\text{ss}} \in \operatorname{argmax}_{a\in A_i} \overline{\theta}^i_a$. If the argmax is not unique, we say $\overline{\theta}^i$ is a *pseudo-steady-state*.

The requirement of existence of a steady state can, at times, be stringent, which is why we allow agents to learn an action a_l if the estimate of that action is always the largest in the limit.

Definition 4. We say the reinforcer *learns action* a_l if there exists a $T > 0$ such that $a_l \in \operatorname{argmax}_{a \in A} \theta_a^i(t)$ for all $t \geq T$. Equivalently, for all $\overline{\theta}^i$ in the forward limit set of a given orbit, $\overline{\theta}_{a_l}^i = \max_{a \in A} \overline{\theta}_a^i(t)$.

It is a simple exercise to show that if the forward limit set of θ is a singleton, an agent who learns action a_l also converges on action a_l. Importantly, an agent can learn action a_l even if she doesn't play said action in each period. The definition of reinforcers is overly permissive, allowing for many procedures which do not behave well in their limit. For this reason, in the rest of the paper we focus on what we call *separable* reinforcers.

Definition 5. A reinforcer (θ^i, π^i) is said to be *separable* if the fluid approximation of θ^i is of the form

$$\frac{d\theta_a^i(t)}{dt} = \alpha_a^i\Big(\theta^i(t)\Big)\Big[U\Big(\theta_a^i(t), \mathbb{E}_{\pi_{-i}}[r^i(a, \pi_{-i})]\Big) + V\Big(\theta^i(t)\Big)\Big]. \tag{2}$$

where $\alpha_a^i(\theta^i) \in [0,1]$, α, U, and V are a.e. Lipschitz in all components, U is Lipschitz everywhere and increasing in $\mathbb{E}[r(a, \pi_{-i})]$ and decreasing in θ_a^i, and $\frac{\partial V}{\partial \theta^{a^i}} < -\frac{\partial U}{\partial \theta^{a^i}}$ almost everywhere.

Two parts make up a separable reinforcer: a component that is equal for all actions, $V(\theta^i)$, and a component that is action specific but Lipschitz over the entire domain K, $U(\theta_a^i, \mathbb{E}[r^i])$. The function U, uniform across all actions, operates on a given action's estimates. The first of the monotonicity assumptions amounts to requesting that a reinforcer's updates increase with good news. The second states that a reinforcer likes surprises: the update shrinks if the agent already holds an action in high regard.

Notice that we allow for heterogeneity in the learning rates α_a^i across actions. The heterogeneity accounts for differences in learning rates originating from the policy. The ε-greedy Q-learning described in Example 1 exemplifies this assumption. For Q-learning,

$$U(Q_a^i(t), \mathbb{E}[r^i]) = \mathbb{E}_{\pi^{-i}}[r^i(a^i, a^{-i})] - Q_a^i(t)$$

and $V(Q^i)$ is simply $\gamma \max_a Q_a^i(t)$. Thus, U is linear in its arguments and Lipschitz everywhere, increasing in rewards and decreasing in $Q_a^i(t)$. The common component V is constant over its Lipschitz domains and since $\gamma < 1$ its derivative is dominated by U's. The learning rate $\alpha_a^i(\theta^i(t))$ in the case of ε-greedy Q-learning is simply $\alpha_a^i \cdot (\pi_\varepsilon(\theta^i(t)))_a$: the Q-learning rate α^i multiplied by the probability of selecting action a given $\theta^i(t)$.

The restriction to separable reinforcers does not rule out standard algorithms. Separability only requires that an algorithm "treats each action the same": the only term that

can differ across actions is the learning rate α. All entries of the vector θ^i adopt the same action-specific component, and any interaction term appears uniformly in all actions' updates.

3 Q-learning in the Prisoner's Dilemma

We analyze the behavior of a simple Q-learning algorithm in a family of Prisoner's Dilemmas. Simulations indicate that independent Q-learners coordinate on cooperation for a wide range of parameters. Coordination is however imperfect, with players periodically alternating between cooperation and defection. We apply continuous-time techniques to characterize the learning outcomes and identify the critical mechanism that sustains cooperation.

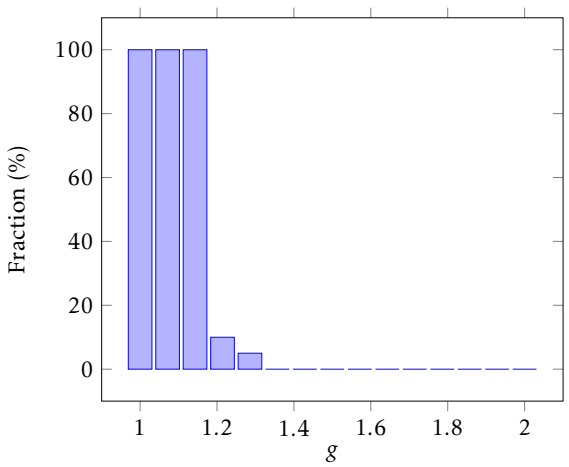
Consider the following family of contribution games, with payoffs given by Figure 1. Alice and Bob have two American dollars each, and they simultaneously decide whether or not to contribute to a shared pool that grows its value by a factor of $1 < g < 2$. The riches in the pool are then shared equally between Alice and Bob, but each agent gets to keep what they didn't contribute on top of that. This game is a canonical model of the free-rider problem. The parameter g models the attractiveness of joint cooperation: the larger g, the more attractive cooperation becomes. However, for all $g \in (1,2)$, the dominant strategy, and the only Nash equilibrium, is to play "defect" and keep one's change.

		Bob: C	Bob: D
Alice	C	$2g, 2g$	$g, 2+g$
	D	$2+g, g$	$2, 2$

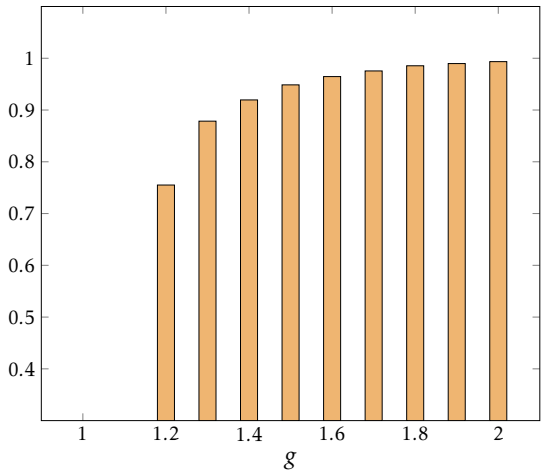
Figure 1: Payoffs of the stage game, $1 < g < 2$.

Simulations. We simulate the path of play of Alice and Bob when they adopt ε-greedy Q-learning in the above free-rider problem 100 times for various values of g. In each simulation, we initialize both algorithms optimistically, i.e. with values larger than the maximum available continuation value, $\frac{2g}{1-\gamma}$. We observe their path of play over $100,000$ time steps, and we say the agents learn profile (a_A, a_B) if in the last 1000 iterations they play such profile over 50% of the time.

Figure 2a shows the results of these numerical experiments. The algorithmic agents learn to play the dominant strategy equilibrium $\{D, D\}$ only for low values of the parameter g. Instead, when g is large, the agents cooperate, albeit imperfectly. Both coopera-



(a) Fraction of runs where the agents learn the Nash Equilibrium $\{D, D\}$.



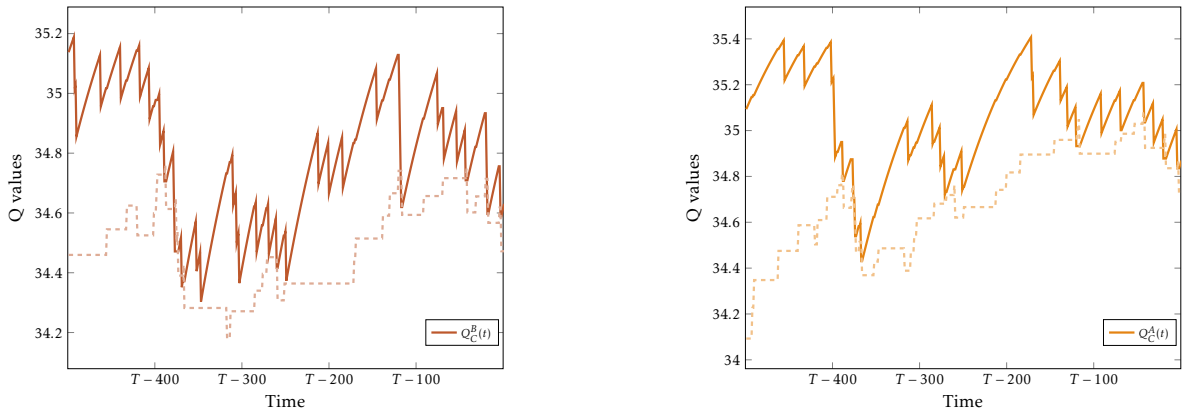
(b) Fraction of time where cooperation is a player's preferred action in a single run for various values of g.

Figure 2: For these simulations, we chose parameters $\varepsilon = 0.1$, $\alpha = 0.05$ and $\gamma = 0.9$.

tion and defection appear in unpredictable but recurrent cycles, as shown in Figures 3a and 3b: the value of collaboration is generally above that of defection, but it drops below $Q_D(k)$ at somewhat regular intervals so that agents switch to playing D. The value of defection then decreases almost immediately, and players revert to cooperation. These simulations highlights a few puzzles. First, cooperation arises only for large values of g, even though defection is always a dominant-strategy. Second, cooperation seems to consist of cycles, but it is hard to impute these to "retaliatory" strategies played by simple Q-learning algorithms. Figure 2b shows that even in the long run Alice and Bob play cooperation for a large fraction of the time.

3.1 Theoretical results

We begin the theoretical analysis by writing down the continuous-time approximation of ε-greedy Q-learning explicitly. When Alice and Bob adopt this algorithm, they learn only about the actions they take, which depend on the values of $Q(k)$. In the parlance of Definition 1, both Alice and Bob evolve according to a function T^i which is discontinuous along the surface $Q_D^i(k) = Q_C^i(k)$. To sidestep this discontinuity in the right-hand side of the discrete-time system, we first apply Theorem 1 to $Q(k)$ over the largest open sets such



(a) Evolution of Alice's Q-values, obtained with $g = 1.8$.

(b) Evolution of Bob's Q-values, obtained with $g = 1.8$.

Figure 3

that $T = (T^A, T^B)$ is everywhere Lipschitz. We call these sets *maximal continuity domains*: in the case of ε-greedy Q-learning these are sets $\omega_{a,b}$ of vectors Q such that Alice's greedy action is a and Bob's greedy action is b.

Over $\omega_{C,C}$ the greedy action for both players is C, so that in every period Alice cooperates with probability[4] $1-\frac{\varepsilon}{2}$ and defects with probability $\frac{\varepsilon}{2}$. Hence, with probability $(1-\frac{\varepsilon}{2})^2$ she collects reward $2g$ — similarly for other profiles. Therefore, the fluid limit in $\omega_{C,C}$ solves

$$\begin{cases} \dfrac{d\mathbf{Q}_C^A(t)}{dt} = \alpha\left(1-\dfrac{\varepsilon}{2}\right)\left[\left(1-\dfrac{\varepsilon}{2}\right)2g + \dfrac{\varepsilon}{2}g + (\gamma-1)\mathbf{Q}_C^A(t)\right] \\ \dfrac{d\mathbf{Q}_D^A(t)}{dt} = \alpha\dfrac{\varepsilon}{2}\left[\left(1-\dfrac{\varepsilon}{2}\right)(2+g) + 2\dfrac{\varepsilon}{2} + \gamma\mathbf{Q}_C^A(t) - \mathbf{Q}_D^i(t)\right] \\ \dfrac{d\mathbf{Q}_C^B(t)}{dt} = \alpha\left(1-\dfrac{\varepsilon}{2}\right)\left[\left(1-\dfrac{\varepsilon}{2}\right)2g + \dfrac{\varepsilon}{2}g + (\gamma-1)\mathbf{Q}_C^B(t)\right] \\ \dfrac{d\mathbf{Q}_D^B(t)}{dt} = \alpha\dfrac{\varepsilon}{2}\left[\left(1-\dfrac{\varepsilon}{2}\right)(2+g) + 2\dfrac{\varepsilon}{2} + \gamma\mathbf{Q}_C^B(t) - \mathbf{Q}_D^i(t)\right] \end{cases} \tag{3}$$

Similar systems appear in all continuity domains, and can be written in matrix form as

$$\dot{\mathbf{Q}}(t) = \begin{cases} A_{C,C}\mathbf{Q}(t) + b_{C,C} \text{ for } \mathbf{Q}(t) \in \omega_{C,C} \\ A_{C,D}\mathbf{Q}(t) + b_{C,D} \text{ for } \mathbf{Q}(t) \in \omega_{C,D} \\ A_{D,C}\mathbf{Q}(t) + b_{D,C} \text{ for } \mathbf{Q}(t) \in \omega_{D,C} \\ A_{D,D}\mathbf{Q}(t) + b_{D,D} \text{ for } \mathbf{Q}(t) \in \omega_{D,D} \end{cases} \tag{4}$$

[4] I.e., C is selected with probability $1-\varepsilon$ if the randomization device instructed the agent to be greedy and with probability $\frac{\varepsilon}{2}$ if the agent plays a random action.

The behavior of this 4-dimensional piecewise-linear system with 4 right-hand sides turns out to be complex. In particular, the system exhibits chaotic behavior over various parametrizations and initial conditions (see Appendix B for more details on chaos theory and its analysis in the contribution game).

To make progress in the analysis of the chaotic system, we begin by restricting attention to a subspace of $\mathbb{R}^4$. In particular, we note that if the initial condition is symmetric, the system is bound to remain symmetric: the space $\{Q \in \mathbb{R}^4 | Q_a^A = Q_a^B \text{ for } a = C, D\} \cong \mathbb{R}^2$ is a subspace for the dynamical system in Equation (4). In what follows, we will then make the following assumption:

Assumption S. Let $\mathbf{Q}_a^A(0) = \mathbf{Q}_a^B(0)$ for $a = C, D$.

Under Assumption S, we can prove the following proposition, which characterizes the limiting behavior of the continuous-time approximation.

Proposition 1. *Let* $\underline{\varepsilon}(g) = 1 - \sqrt{\frac{2-g}{g}}$. *The forward limit set of* $\mathbf{Q}$ *is a singleton for any initial condition. If* $\varepsilon \geq \underline{\varepsilon}(g)$, *all initial conditions lead to the unique steady state*

$$q_D^{eq} = \left(\frac{2\varepsilon + 2g - \varepsilon g}{2} + \frac{\gamma(4 + \varepsilon g)}{2(1-\gamma)}, \frac{4 + \varepsilon g}{2(1-\gamma)} \right)$$

which lies in $\omega_{D,D}$.

If $\varepsilon < \underline{\varepsilon}(g)$, *initial conditions may lie in one of two regions of attractions,* R_D *and* R_C. *All initial conditions in* R_D *lead to the steady state* q_D^{eq}. *All initial conditions in* R_C *instead lead to the pseudo-steady state*

$$q_C^{eq} = (y, y) \text{ where } y = \frac{1 + g + \sqrt{(g-1)(g-1-\varepsilon g + \frac{\varepsilon^2 g}{2})}}{(1-\gamma)},$$

which lies in $\overline{\omega}_{D,D} \cap \overline{\omega}_{C,C}$.

In the symmetric subspace, the limiting behavior of the system can be twofold. When ε is larger than the critical level $\underline{\varepsilon}(g) = 1 - \sqrt{\frac{2-g}{g}}$, the algorithms converge on a steady state q_D^{eq} where both players find defection to be the preferred action. However, when ε is below the critical level, there exists additional steady state. In this steady state, Alice's and Bob's estimates of cooperation and defection coincide — we therefore refer to this as a pseudo-steady-state, q_C^{eq}. In this pseudo-steady state the Q estimates fall between the long-run value of mutual defection, $\frac{\mathbb{E}[r(D,a^{-i})]}{1-\gamma}$, and the long-run value of mutual cooperation, $\frac{\mathbb{E}[r(C,a^{-i})]}{1-\gamma}$. Alice and Bob are indifferent between cooperation and defection in q_C^{eq},

and they play cooperation for a fraction τ of the time and defect for a fraction $1-\tau$ of the time.

Corollary 1. *In the pseudo-steady-state q_C^{eq} agents spend τ_c fraction of their time cooperating, where*

$$\tau = \frac{\frac{\varepsilon^2 g}{2} + \varepsilon - 2 - q_C^{eq}(\gamma-1)(1-\varepsilon)}{2(\varepsilon-1)(1+g+(\gamma-1)q_C^{eq})} \in \left[\frac{1}{2}, 1\right].$$

The pseudo-steady-state q_C^{eq} corresponds to the imperfect cooperation we observed in the experiments. In particular, the analytic expression for the time spent cooperating approximates its discrete-time experimental counterpart closely, as shown in Figure 7b.[5]

3.2 Sketch of the proofs

Under Assumption S the system evolves according to the following piecewise-linear dynamical system:

$$\dot{\mathbf{Q}}(t) = \begin{cases} A_{C,C}\mathbf{Q}(t) + b_{C,C} \text{ for } \mathbf{Q}(t) \in \omega_{C,C} \\ A_{D,D}\mathbf{Q}(t) + b_{D,D} \text{ for } \mathbf{Q}(t) \in \omega_{D,D} \end{cases}$$

We denote the two vector fields over each domain $\omega_{a,a}$ as F_a for $a \in \{C, D\}$. The flows within each $\omega_{a,a}$ characterize the evolution of the Q-functions. The system at large however is discontinuous: we would like to preserve continuity of any given flow along the boundary $\overline{\omega}_{C,C} \cap \overline{\omega}_{D,D}$. Proposition 2 below guarantees that we can suitably extend the flows on the boundary, similarly to continuous pasting techniques, such that the flow defined by the fluid limit is globally defined forward in time.

Proposition 2. *Let F_a be the field defined as above over $\omega_{a,a}$ for all $a \in \{C, D\}$. There exists a global solution in the sense of Filippov (1988) to the differential inclusion*

$$\begin{aligned} &\frac{d\mathbf{Q}_t}{dt} = F_a(\mathbf{Q}_t) && \text{over } \omega_{a,a} \text{ for } a = C, D \\ &\frac{d\mathbf{Q}_t}{dt} \in co\left\{F_a(\mathbf{Q}_t) \,\middle|\, \forall a = C, D\right\} && \text{when } \mathbf{Q}_t \in \overline{\omega}_{C,C} \cap \overline{\omega}_{D,D} \end{aligned}$$

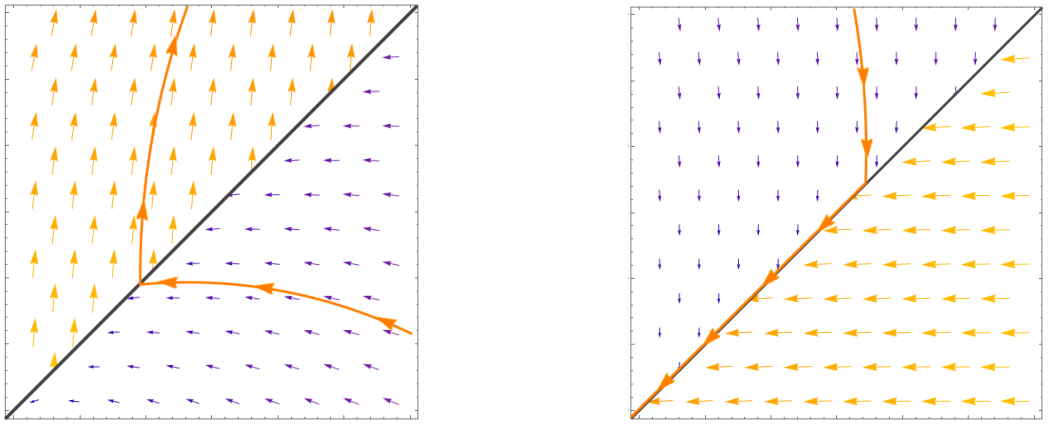
where $co\{\cdot\}$ denotes the convex hull of a set, and $\overline{\omega}$ is the closure of ω.

Both vector fields F_C and F_D are well-defined also on the boundary between $\omega_{C,C}$ and $\omega_{D,D}$. This boundary is called a *switching surface*, because the laws of motion must switch from one field to the other. Adopting the Filippov convention, we can define a vector

[5] In Appendix B we observe that the general asymmetric 4-D system gravitates around a similar equilibrium, where $Q_C = Q_D$. We interpret it as additional support for the restriction to a symmetric subspace.

field on the switching surface such that all flows extend to a global solution and such that certain continuous-pasting conditions are satisfied.[6]

Let us call $\hat{\mathbf{N}}$ the unit normal vector to the switching surface $\overline{\omega}_{C,C} \cap \overline{\omega}_{D,D}$. We divide the switching surface in three regions, according to the signs of the normal components of the two vector fields F_C and F_D. A *crossing region* occurs when both normal fields to the boundary $\hat{\mathbf{N}} \cdot F_a$ are of the same sign. Either F_C or F_D can be used to define the law of motion on the surface: any orbit leaves the switching boundary immediately. A *repulsive region* occurs where both normals $\hat{\mathbf{N}} \cdot F_a$ face away from the boundary, which will then never be reached. Unless initialized here, an orbit will never intersect the repulsive region, thus we do not need to define a field here. Finally, a *sliding region* occurs when both normal components $\hat{\mathbf{N}} \cdot F_a$ of the two vector fields point towards the boundary. Every flow hitting the switching surface in the sliding region must continue on the switching surface, *sliding along the boundary*. The sliding vector field is defined as a convex combination of the two vector fields F_C and F_D with parameter τ such that the normal component to the switching surface vanishes, i.e. $\tau(\hat{\mathbf{N}} \cdot F_C) + (1-\tau)(\hat{\mathbf{N}} \cdot F_D) = \vec{0}$. The vector field $\tau F_C + (1-\tau) F_D$ is the unique vector field in the convex hull of F_C and F_D whose flow is confined to the switching surface and which satisfies the differential inclusion requirements. Figure 4 plots the vector fields around the boundary and shows an example of sliding and crossing boundaries, together with a sample orbit. For more details on how the field is calculated we refer the reader to the proof of Proposition 1 in the Appendix.

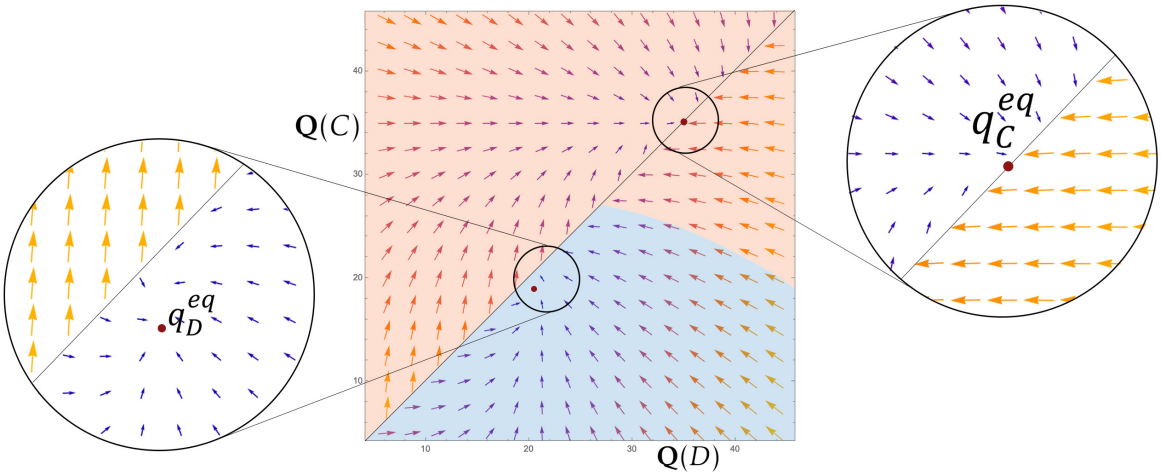


(a) Trajectory crossing the boundary.

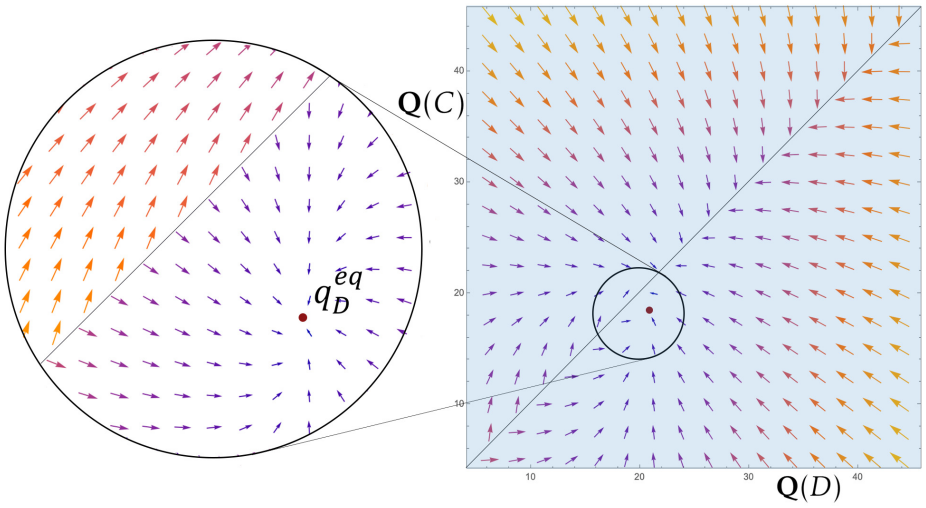
(b) Trajectory sliding along the boundary.

Figure 4: Depiction of two discontinuous flows around a switching surface, in the crossing and sliding case.

[6] Uniqueness in general is not guaranteed: behavior on the switching surface can lead to multiplicities in the behavior of the system. The law of motion on the switching surface needs only to belong to the convex hull of the laws of motion on either side; by constraning the motion to satisfy certain continuous-pasting conditions we obtain well-defined orbits.



(a) Phase space with $g = 1.8$.



(b) Phase space with $g = 1.1$.

Figure 5: Stationary points are marked with a red dot. The domain of attraction of the cooperative outcome is green-shaded, and the one for the non-cooperative outcome is blue-shaded.

Stability analysis In Figure 5 we plot the vector fields that define $\mathbf{Q}$ for two different values of g. The orbits of $\mathbf{Q}$ evolve according to these vector fields. We highlight the presence of two stationary points when g is large — the pseudo-steady-state on the boundary disappears for low values of g. The analytical characterization of the two points mentioned in Proposition 1 follows the derivation in Appendix A.

We will refer to the point q_C^{eq} as the *cooperative steady-state*. Its existence hinges on the relationship between the value of cooperation and the exploration rate. Figure 6 shows the minimum exploration $\underline{\varepsilon}$ that guarantees a cooperative steady-state and how it varies

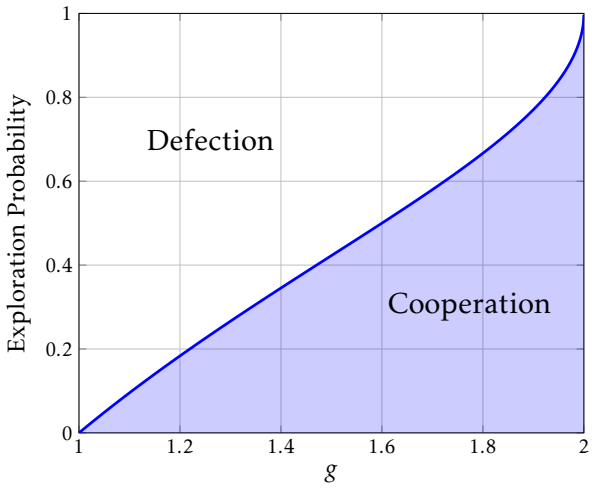


Figure 6: Maximum exploration rate $\underline{\varepsilon}$ to support the cooperative equilibrium, as a function of the growth rate. For all $\varepsilon > \underline{\varepsilon}(g)$ there does not exist a steady-state where both algorithms learn to cooperate.

with the value of cooperation. Intuitively, as g increases, the relative benefit of defecting decreases (and vanishes completely when $g = 2$), so more and more exploration is needed to realize that D is a dominant action. For example, if $g = 1.8$ the exploration rate required to guarantee convergence on $\{D, D\}$ is about 70%, which is considerably larger than the standard employed in practice.[7]

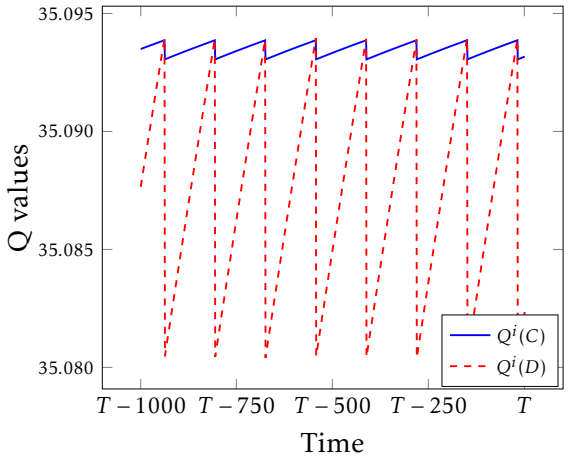
Sustaining cooperation Note that, when simulated with small but discrete time steps, the dynamical system closely mimics the path of play of the discrete Q-learning. We can see this by comparing Figure 7a with Figures 3a and 3b, shedding some light on how Q-learning could sustain cooperation.

Suppose C is the preferred action of both players: Alice and Bob cooperate but with probability $\frac{\varepsilon}{2}$ one defects and realizes its benefit. Over time, Q_D^i must rise above Q_C^i. Suppose this happens first to Alice. Once Alice defects, Bob will defect immediately after — cooperating makes Bob considerably worse off when Alice defects. Mutual defection decreases the value of Q_D^i for Alice and Bob, but it is slow to change Q_C^i. Effectively, when the exploration rate is low, the adaptive agents play a symmetric profile of actions too often. We call this effect the *spontaneous coupling*, because the estimates of independent Q-learners tend to evolve symmetrically.

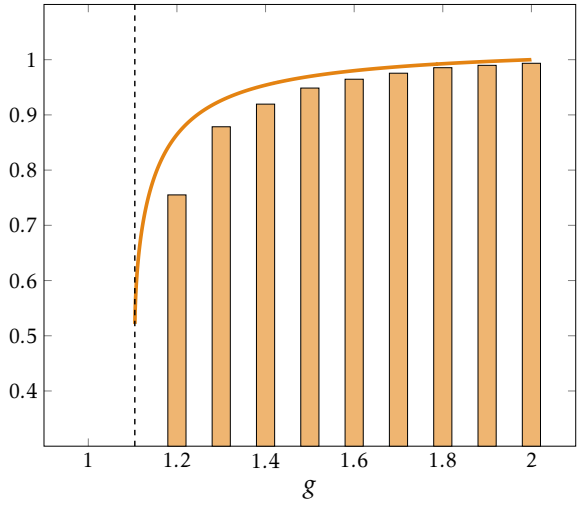
When algorithms start defecting, they experiment with cooperation infrequently, which

[7] The literature on Q-learning in games usually employs $\varepsilon = 0.1$ or smaller, either fixed or decreasing over time. E.g., see Gomes and Kowalczyk (2009).

grants considerable persistence to the estimate of cooperation's value Q_C^i. Alice's estimate Q_C^A remains close to the long-run value of mutual cooperation. Instead, her estimate Q_D^A drops dramatically once both agents begin defecting. Effectively, Alice never realizes the downside of cooperating when Bob defects because both revert to cooperation simultaneously.



(a) Cycles of play around the cooperative equilibrium in the discretized ODE system.



(b) Proportion of time spent playing the cooperative outcome in q_C^{eq}, analytically and in the simulations.

Figure 7

The pseudo-steady-state on the boundary is the continuous-time counterpart of these cooperative cycles in the discrete system. The discrete cycles tend to a single point in the continuous-time limit. In the pseudo-steady-state, agents are 'ìndifferent" between cooperation and defection. Both spend some "local time" playing either action: we interpret the weights $\tau, 1-\tau$ assigned to the fields on the boundary as the fraction of time dedicated to cooperation and defection, respectively.[8] The local time is such that the infinitesimal incentives around the stationary point are balanced.

Figure 7b plots τ and shows that the proportion of time spent playing the cooperative profile at equilibrium varies with g. Compare this with Figure 2b: the analytical expression τ approximates well the estimates of the same quantity from the simulations. Additionally, recall that the discrete simulated system always exhibits chaotic behavior: nonetheless, the expression τ appears to fit the data.

[8]This intuition can formalized using the idea of hysteresis loop around the boundary; see di Bernardo et al. (2008)

4 General Reinforcers

The mechanics of spontaneous coupling, identified in Section 3, appear tied to the policy of the reinforcer. We show that the driving force of inadvertent coordination lies in the algorithm's uneven learning rates across different actions. Learning at different rates about different actions facilitates coordination for general reinforcers. We establish a sufficient condition on the parameters of the algorithm which avoids coupling: learning rates must be uniform across actions.

4.1 Reinforcers in the Prisoner's Dilemma

The ε-greedy Q-learning analysis in the Prisoner's Dilemma of Section 3 brings out a mechanism which sustains collusive behavior between algorithms. Suppose Alice's preferred action is cooperation: at rate $1-\frac{\varepsilon}{2}$ she learns about the payoffs of cooperating, while she learns about the payoffs of defection more slowly, at rate $\frac{\varepsilon}{2}$. Exploration regulates how quickly Alice learns about the payoff of a deviation. When algorithms are coupled, low rates of exploration impair the ability of the algorithm to correctly estimate the value of deviations. The learning rates dampen the law of motion of the estimates unevenly, allowing faster updates only for the greedy actioin.

This phenomenon can be formalized without referring to a specific policy. In fact, a given policy only affects the learning rates α_a of different actions. For example, in the case of ε-greedy Q-learning, recall the differential equations within the maximal continuity domain $\omega_{D,D}$:

$$\begin{cases} \dfrac{d\mathbf{Q}_C^A}{dt}(t) = \boldsymbol{\alpha}\dfrac{\boldsymbol{\varepsilon}}{\mathbf{2}} \left[\left(1-\dfrac{\varepsilon}{2}\right)g + \dfrac{\varepsilon}{2}2g + (\gamma-1)\mathbf{Q}_C^A(t)\right] \\ \dfrac{d\mathbf{Q}_D^A}{dt}(t) = \boldsymbol{\alpha}\left(\mathbf{1}-\dfrac{\boldsymbol{\varepsilon}}{\mathbf{2}}\right)\left[\left(1-\dfrac{\varepsilon}{2}\right)2 + \dfrac{\varepsilon}{2}(2+g) + \gamma\mathbf{Q}_C^A(t) - \mathbf{Q}_D^A(t)\right] \end{cases}$$

In the language of separable reinforcers, the decision rule only affects $\alpha_a^i(\theta^i)$. When time is continuous, each estimate is updated according to its payoff given the opponents' actions — the only role of the policy is to determine the relative learning rates. It is clear from this discussion that the absolute magnitude of the learning rates does not matter, which is why we focus on the relative rates.

Definition 6. The relative learning rate of action a^i is the ratio

$$RLR(a^i) = \frac{\alpha_{a^i}}{\sum_{a\in A_i}\alpha_a}.$$

Theorem 2 shows how differences in RLR across actions generally give rise to spontaneous coupling. The coupling appears in a Prisoner's Dilemma for any pair of agents using any (separable) reinforcer. For simplicity, we restrict attention to reinforcers with maximal continuity domains equal to $\omega_{C,C}$ and $\omega_{D,D}$, and learning rates α_C, α_D constant over both.

Theorem 2. *Let each agent learn through the same greedy reinforcer in any Prisoner's Dilemma, and let Assumption S be satisfied. There exist an open set $A \subset \mathbb{R}^4_+$ such that for all parameters $\{\alpha_j(\omega_{k,k})\}_{j,k=C,D} \in A$ there exists a pseudo-steady-state on the boundary $\overline{\omega}_{C,C} \cap \overline{\omega}_{D,D}$.*

We prove Theorem 2 formally in the Appendix. The proof is constructive: we show that there exist a symmetric tuple of αs such that the sliding vector field on the indifference boundary is null. In particular, we show this when the vector fields on either side of the indifference boundary are opposite, i.e. the local time is $\frac{1}{2}$. We apply homotopical arguments to show that there exists such αs, and then we perturb the problem and we obtain the result by continuity.

The Theorem highlights an important fact: even in dominant-strategy-solvable games, reinforcers will not play the dominant strategy for various ranges of parameters.

4.2 Reinforcers with Uniform Learning Rates

We provide a simple condition that guarantees reinforcers converge on dominant strategies: reinforcers' relative learning rates must be uniform across all actions. Intuitively, even if algorithms are coupled uniform learning rates allow agents to evaluate deviations correctly, thus leading them to their dominant strategy.

We first need the following technical assumption:

Assumption A3 (Thickness)**.** Let $G^t_{-i}(a)$ be the distribution over actions of all players but i at time t. Then, there exists a $\chi > 0$ and a T such that for all $t > T$, $G^t_{-i}(a) \geq \chi$ for all $a \in A_{-i}$.

Thickness ensures that sufficient exploration is carried out by all players in the limit. Any game where all agents adopt a ε-greedy policy satisfies this assumption, for example. More generally, thickness states that each action profile is played with positive (albeit small) probability in the limit.

Theorem 3. *Suppose Assumption A3 is satisfied for all players.*[9] *In any game with a dominant*

[9] We require this assumption because we formulate the Theorem for games solvable by weak dominance. We can instead drop Assumption A3 when the best-response to any of the opponent's strategies is unique.

strategy equilibrium, a reinforcer with $RLR(a^i) = RLR(\tilde{a}^i)$ *for all* $a^i, \tilde{a}^i \in A_i$ *learns the dominant strategy. If the forward limit set of* θ *is a singleton, then* θ *converges on the dominant strategy.*

This result is surprisingly powerful. In particular, we make almost no assumption about the opponent's play: as long as all actions are played with some positive probability even in the limit, the reinforcer will learn to play its dominant strategy. If the game is solvable by strict domination, we do not even require Assumption A3. Compare this with the previous result, Theorem 2, which required symmetric reinforcers in a Prisoner's Dilemma. Uniform learning rates ensure any reinforcer will learn the dominant strategy for any number of opponents, as well as opponents who adopt different learning algorithms, or the same learning algorithm with different parameters. All these asymmetries can be accommodated by Theorem 3.

The assumption that relative learning rates be uniform across actions may appear stringent: it might for example require restricting the exploration of the algorithm to try each action uniformly at random. Instead, we propose a different strategy to achieve identical RLR across actions.

Consider again Q-learning. The algorithm updates the statistic of action a^i when action a^i is taken and its reward is observed, but it leaves other statistics unattended when the corresponding action is not selected. Suppose however that the agents were able to compute counterfactuals. That is, suppose that, after choosing an action a^i in period t, the algorithmic agent was able to work out $r_t(\tilde{a}^i, a^{-i})$ for all $\tilde{a}^i \neq a^i$. Then, the statistics of all actions could be updated simultaneously, using the reward that each action would have procured had it been played in that period. Simultaneous updates are sometimes referred to as *synchronous* learning:[10] in this case learning happens at the same rate for all actions. The ability to compute counterfactuals affects the learning rates: the second term in $\alpha^i_a \cdot (\pi_\varepsilon(\theta^i(t)))_a$ disappears when agents update synchronously. When asymmetric learning rates arise from missing or asymmetric experiments, counterfactual information (that is, a correct model of the environment) is sufficient to eliminate the asymmetry. The following corollary formalizes this intuition.

Corollary 2. *Under the same assumptions of Theorem 3, a reinforcer who can compute counterfactuals always learns the dominant strategy. If its forward limit set is a singleton, it converges to the dominant strategy.*

It is perhaps not surprising that counterfactuals help to learn to play equilibria. In fact, the theory of Nash equilibrium is based on the assumption that agents can compute

[10] The term synchronous appears in Asker et al. (2022), but the idea of agents learning from counterfactuals is present already in Tumer and Khani (2009).

the payoff that would have obtained if they had played a different action, treating the opponents' strategies as fixed. This in turn allows them to evaluate incentives to deviate. Corollary 2 establishes that reinforcer algorithms successfully rule out dominated strategies, provided they have access to a method to compute counterfactuals. Reinforcers with counterfactuals will learn to play the (unique) equilibrium.

More generally, let us consider the procedure of iterated elimination of strictly of dominated strategies (IESDS). However, we rescrict deletion to strategies strictly dominated by another *pure* strategy, because reinforcers do not deal well with mixed strategies.[11]

Definition 7. We say that an action $a^i \in A_i$ is *pure-rationalizable* if there is an order of IESDS such that a^i survives the IESDS procedure.

In general, for a certain order of deletion of dominated strategies action a^i might get eliminated. However, as long as there is an order such that a^i survives the IESDS process, we consider a^i pure-rationalizable. Our next theorem shows that reinforcers with access to counterfactuals only play pure-rationalizable strategies in the limit.

Theorem 4. *Let all players in game G learn through a reinforcer using a ε-greedy policy with $RLR(a^i) = RLR(\tilde{a}^i)$ for all $a^i, \tilde{a}^i \in A_i$ for all $i \in N$. Assume $\varepsilon > 0$ is small enough so that the reward's order is preserved, i.e. if a^{-i} is the preferred profile of actions of agent i's opponent, $\mathbb{E}_{\pi^{-i}}[r(a^i, a^{-i})] > \mathbb{E}_{\pi^{-i}}[r(\tilde{a}^i, a^{-i})]$ when $r(a^i, a^{-i}) > r(\tilde{a}^i, a^{-i})$. Then, all actions learned by the players are pure-rationalizable in the game G under the same IESDS order.*

One implication of Theorem 4 is that, in a supermodular game with a unique equilibrium, ε-greedy reinforcers with uniform learning rates always learn to play Nash equilibrium strategies. More generally, in any pure-dominance-solvable game, reinforcers will learn the pure-strategy Nash equilibrium.

We can interpret relative learning rates in an intuitive sense as the relative ability to work out counterfactuals for a given action. Because the utility of a given action depends on the opponent's path of play, uneven learning rates generate biased estimates. Small relative learning rates fail to account for asymmetric play, impairing the ability of the algorithm to best-respond. Uniform learning rates instead guarantee correct counterfactual estimates. With unbiased counterfactuals, abandoning dominated strategies is immediate.

In the next section, we leverage these results for implementation in hybrid markets: games played by rational as well as learning agents. Furthermore, Sections 6.1 and 6.2

[11] Definition 1 makes clear that it is impossible for adaptive algorithms to learn the value of randomizing across actions.

show how Theorems 3 and 4 can be applied to study analytically environments that received considerable attention in the experimental literature.

5 Application: Learning-Robust Mechanism Design

In this section we apply our findings to the problem of mechanism design. We are interested in designing a dominant strategy mechanism with a feedback rule that guarantees that if players use separable reinforcers and update their estimates according to their feedback, they will learn the dominant strategy. Moreover, we are interested in finding the minimal feedback necessary to accomplish this goal, because we are concerned with unintended consequences of providing the algorithms more information than necessary about the play of the other players.

Consider a canonical model of implementation with private information. Let $\mathcal{X}$ be the set of possible outcomes, and let there be n agents with types $(\lambda^i)_{i=1,\dots,n} \in \bigtimes_{i=1\dots n} \Lambda_i = \Lambda$ fixed over time.[12] Type λ^i determines agent i's preferences $u^i \colon \mathcal{X} \times \Lambda_i \to \mathbb{R}$ over outcomes. A direct revelation mechanism requires each agent to report a type $\hat{\lambda}^i$. The mechanism then maps the reported type profile $\hat{\lambda}$ to an outcome, $f(\hat{\lambda})$. We say a mechanism is *strategy-proof* if it is a direct revelation mechanism and reporting truthfully is a dominant strategy:

$$u^i(f(\lambda^i, \lambda^{-i}), \lambda^i) \geq u^i(f(\hat{\lambda}^i, \lambda^{-i}), \lambda^i) \qquad \forall \hat{\lambda}^i \neq \lambda^i.$$

Assume further that a subset of agents $L \subseteq N$ act according to the choices of their own reinforcer. Agents in L, learners, assess the value of each individual report over multiple iterations of the mechanism. Agents in $N \setminus L$ are instead rational, but we assume they are short-lived (or myopic). That is, they rationally play their static dominant strategy instead of trying to manipulate the learning agents. In each period a fresh group of rational agents arrives, and both rational and learning agents choose a report. Once the mechanism has selected the outcome, rational agents leave, and learners update their estimates. We call this setting a *hybrid market*, because rational and learning agents coexist.

We assume that rational agents play their dominant strategy and report truthfully (by definition). However, as we have seen in Sections 3 and 4 coupling between independent algorithms may lead to behavior consistent with collusive agreements. We thus seek learning-robust strategy-proof mechanisms (LRSM).

[12] It is simple to extend this result to allow for types drawn i.i.d. in every period, but for simplicity we stick to a constant type in this section.

Definition 8. Suppose agents in L adopt a (separable) reinforcer. A strategy-proof mechanism is *learning-robust* if each agent $l \in L$ learns the truthful report $\lambda^l_{\text{truthful}}$.

In a LRSM, algorithmic agents will learn that reporting truthfully delivers the largest value. The actions taken by the reinforcers might be tainted by exploratory policies, but the estimates they identify will agree with the strategy-proof nature of the game.

Let us assume that the information provided by the designer is used by the algorithms to make inference about payoffs from reports they dismissed. Then, our results show that we can ensure robustness of a mechanism by supplying enough information to the reinforcers so that they can evaluate counterfactuals.[13] In this private information setting, the designer can assist algorithms in their counterfactual calculations. Counterfactual utilities hinge on the opponents' reports, so the designer can help by revealing some private information after she receives all reports. We refer to this ex-post revelation as feedback provision. Given a strategy-proof mechanism f, we look for a LRSM for f, that is, an ex-post feedback policy which allows agent to compute counterfactuals.

A feedback policy for agent i is essentially a partition of the space of opponents' types, Λ_{-i}. After receiving the reports, the designer communicates to each agent in which element of their partition they find themselves. Using these partitions, agents can compute what outcome they could have enforced by unilaterally deviating to a different report.

Of course, a designer can always opt for a full revelation feedback policy. Revealing everyone's report after the mechanim allows algorithms to compute payoffs from every report, but it can induce additional costs and concerns. First, insisting on revealing all private information may facilitate tacit or explicit collusion. Second, revealing all private information may result in large communication costs. Finally, it is not necessarily true that, when provided with all reports, computing the allocation is a simple task. In certain complex auctions for example, discovering the price and allocation requires solving complex optimization problems.

We define a privacy order on the space of feedback policies. We can show that this order is a lattice, and thus there exist both a minimally- and a maximally-private feedback policy. The former is the full revelation policy, consistent with our intuition. We characterize the latter, by showing that the designer can communicate just enough information to ensure that each agent can compute its counterfactuals, and nothing more. It turns out that such feedback is informationally-equivalent to the well-known *menu* formulation of the mechanism (Hammond (1979)).

[13] Recall from Corollary 2 that updating the estimates θ_a according to counterfactual information enforces uniform learning rates, but this is only a sufficient condition.

Theorem 5. *Let f be a strategy-proof mechanism. Then,*

1. *There always exists a LRSM for f,*

2. *The maximally-private LRSM for f is a menu description.*

Menu descriptions are the ex-post feedback counterpart of menu mechanisms. Hammond (1979) defines menu mechanism as providing each agent with a menu, which depends on the profile of reports of the opponents, and let the agent choose his preferred outcome. Instead, we provide feedback in the form of menus — algorithms can compute what would have happened had they reported a different type.

The feedback of menu descriptions is an aggregator of market information, which helps agents evaluate the true value of truthful reporting. Parkes (2004) argues that mechanism design can play an important role in shaping algorithmic systems. The author describes *learnable* mechanism design — the idea of explicitly designing mechanisms to maximize and improve performance considering the agent's adaptive behavior. As he suggests, *"a useful learnable mechanism would provide information, for example via price signals, to maximize the effectiveness with which individual agents can learn equilibrium strategies"*. We view this section as a formalization of this intuition, as we have shown that feedback design can make traditional strategyproof settings robust to adaptive algorithmic players by providing price signals, or menu descriptions.

Finally, note that the world of online auctions has partly begun to provide feedback to its participants. Google's auctions for display advertising provide feedback, in the form of a "minimum bid to win", after each auction has concluded. The minimum bid to win is indeed the menu mechanism for an single-item allocation problem. In the next subsection we will show what a menu description would look like in a simple VCG auction for online advertising, such as one used by Yandex.ru for their search advertising business.

All formal statements and proofs in this section are presented in Appendix C.

5.1 VCG for online search advertising

Consider the following simple model of search advertising. There are n bidders for a given query, each with their own value $v_i \in \{0, 0.1, \ldots, 10\}$ for each click. Without loss we can order bidders by their valuations: $v_1 \leq v_2 \leq \ldots v_n$. The search site offers two ad slots. The first ad slot will bring a predicted traffic volume of 100 units, while the second ad slot will only bring in 80 units of traffic. The search site is running VCG: the winners of the two slots will be the agents with the two largest bids. Let us assume for simplicity that ties are broken in favor of the agent with larger index i. Both winners will pay the

largest losing bid for 80 units of traffic. Additionally, the winner of the first ad slot will pay the bid of the winner of the second ad slot for the extra 20 units he receives. This because the pivotal bidder for the last 20 units is the winner of the second ad slot, not the loser with the largest bid.

Suppose first that all agents report truthfully. Then, agent n wins the first ad slot, and agent $n-1$ wins the second. Agent $n-1$ pays an estimated $80v_{n-2}$, while agent n pays $80v_{n-2} + 20v_{n-1}$. Now, imagine agent k was attempting to learn how to play by bidding according to a separable reinforcer. The designer would want to provide feedback to the agent, to ensure he'd be able to compute what would have happened, had he bid an amount $\hat{v}_k \neq v_k$, keeping everyone else's reports fixed. The feedback required is in fact simple: agent k needs a price for the second ad slot, and a price for the first. In this example, the designer would communicate prices v_{n-1} and v_n.

To see why these prices are sufficient, consider agent k's calculations. There are only three possibilities. If he bid $\hat{v}_k \leq v_{n-1}$, then he would receive zero payoff, the same as if he was to bid truthfully. Suppose he bid $v_{n-1} < \hat{v}_k \leq v_n$ instead: then agent k would win the second ad slot, and pay $80v_{n-1}$. Finally, if agent k bid $v_n < \hat{v}_k$, he would win the first ad slot. His payment would then be $80v_{n-1} + 20v_n$. All three counterfactual payoffs only require two prices: the bids of the two winners.

Similarly, the winner of the second slot requires two prices: v_n and v_{n-2}. The winner of the first slot instead requires v_{n-1} and v_{n-2}. In a VCG setting communication reduces to revealing the values of the bidders that are pivotal for the specific agent. This is indeed a menu description, and it is much more private than the full-feedback policy, which would require communicating all reports $\{v_1, \ldots, v_n\}$ to every agent in the auction.

6 Application: Interpreting Results of Simulations

In this section we apply our results to provide new explanations for experimental results in two recent papers by Asker et al. (2022) and by Banchio and Skrzypacz (2022).

6.1 Bertrand Competition

We now apply our results to provide new insights about a recent paper by Asker et al. (2022). The authors simulate algorithmic competition in a Bertrand oligopoly, and find that collusion depends critically on what they call "synchronicity" of the algorithm. To provide insight about their game, we study a simplified version of it with a smaller action space.

There are two firms, Alice Inc. and Bob Ltd., which face a common demand for their product. Assume that the market demand is $D(p_A, p_B) = 3 - \min\{p_A, p_B\}$, and if the two firms charge the same price they split demand equally. Suppose that each firm has 0 marginal cost, and for simplicity let the firms choose only between two prices: $p \in \{0.5, 2\}$. Profits equal price times individual demand. This Bertrand game has only one static Nash equilibrium: the profile $\{0.5, 0.5\}$. Asker et al. (2022) consider two variations of the Q-learning algorithm, both of which are *greedy*, i.e., the action taken is always the one with the highest estimated value.

(i) Asynchronous Greedy Q-learning: the algorithm updates only the Q-value of the action taken in each period;

(ii) Synchronous Greedy Q-learning: the algorithm updates all Q-values in each period, with the return that it could have obtained had he played the other action instead, but holding the opponent's action fixed.

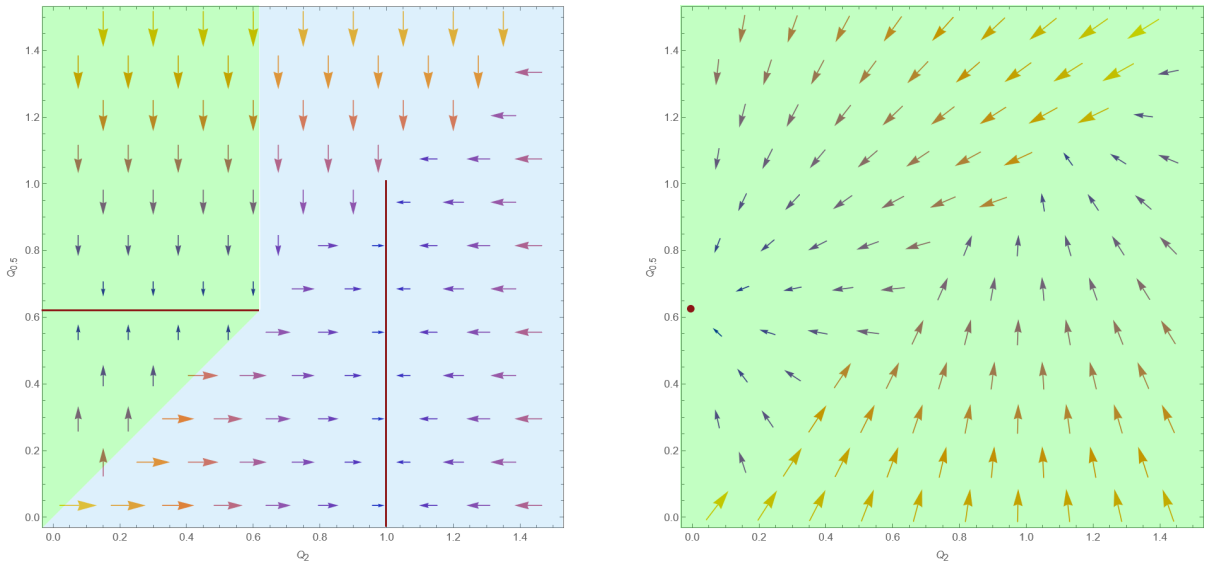
Using our Theorem 1, we analyze the fluid limit of this game. Figure 8 plots the dynamic systems for both versions of the algorithms for $\gamma = 0$.[14] The figures depict the vector fields governing the dynamics of the continuous-time analogous of each Q-learning procedure, with the value of the low price on the vertical axis and that of the high price on the horizontal axis.

Asynchronous Learning. The fields when both firms play asynchronous Q-learning and both choose 0.5 and 2, respectively, are:

$$\begin{cases} \frac{d\mathbf{Q}_{0.5}}{dt} = \alpha\left(\frac{5}{8} + (\gamma - 1)\mathbf{Q}_{0.5}\right) \\ \frac{d\mathbf{Q}_2}{dt} = 0 \end{cases} \quad \text{on } \omega_{0.5,0.5} \qquad \begin{cases} \frac{d\mathbf{Q}_{0.5}}{dt} = 0 \\ \frac{d\mathbf{Q}_2}{dt} = \alpha\left(1 + (\gamma - 1)\mathbf{Q}_2\right) \end{cases} \quad \text{on } \omega_{2,2}$$

Figure 8a plots the fields within their domains. As shown in the picture, there are two equilibrium regions. For values of $\mathbf{Q}_2 \leq \mathbf{Q}_{0.5} = \frac{5}{8}$ the algorithms converge on the competitive outcome. However, for $\mathbf{Q}_{0.5} \leq \mathbf{Q}_2 = 1$ the algorithms collude. Notice also that the domains of attraction of these equilibria are unbalanced: in order to converge on competition, Q-learning needs to be initialized with a strong bias towards competition. Otherwise, the collusive outcome is an attractor. In particular, the optimistic initialization used in Asker et al. (2022) and common in the literature leads to collusive outcomes.

[14] The choice of $\gamma = 0$ reflects the specification of Asker et al. (2022). We have computed the vector fields for other values of γ and the results do not change.



(a) Asynchronous updating.

(b) Synchronous updating.

Figure 8: The green-shaded area denotes the domain of attraction of the competitive outcome, while the blue-shaded area is the domain of attraction of the collusive outcome. The red dot and red lines are the equilibria of the systems.

These results are robust to an $\varepsilon-$greedy specification: the *spontaneous coupling* introduced in Section 3 again sustains collusion. Because most observations of the returns from a supra-competitive price are obtained when colluding, the estimates of returns from charging a price of 2 are consistent with mutual collusion during a competitive phase. The high persistence of these estimates amounts to low exploration of alternative strategies.

Synchronous Learning. Instead, if both firms adopt Synchronous Q-learning, the system is described by the following fields:

$$\begin{cases} \frac{d\mathbf{Q}_{0.5}}{dt} = \alpha\left(\frac{5}{8} + (\gamma-1)\mathbf{Q}_{0.5}\right) \\ \frac{d\mathbf{Q}_2}{dt} = \alpha\left(\gamma\mathbf{Q}_{0.5} - \mathbf{Q}_2\right) \end{cases} \text{ on } \omega_{0.5,0.5} \qquad \begin{cases} \frac{d\mathbf{Q}_{0.5}}{dt} = \alpha\left(\frac{5}{4} + \gamma\mathbf{Q}_2 - \mathbf{Q}_{0.5}\right) \\ \frac{d\mathbf{Q}_2}{dt} = \alpha\left(1 + (\gamma-1)\mathbf{Q}_2\right) \end{cases} \text{ on } \omega_{2,2}$$

There is only one equilibrium, at $\mathbf{Q}_{0.5} = \frac{5}{8}, \mathbf{Q}_2 = 0$. The plot of Figure 8b shows a clear pattern: the two firms can only converge on competition. There is no sliding along the boundary between the competitive and collusive pricing: everywhere the trajectories move from colluding to competing. When the two firms are colluding, all arrows point upward: the algorithms correctly estimate the value of a one-shot deviation, with-

out internalizing the effect that defecting from a collusive outcome will have on returns in the future. The counterfactual rewards do not account for the losses that will stem from a change in the opponents' play after one's deviation. Once the firms begin competing it is then impossible to revert back to collusion: the counterfactual return of a deviation is zero. Observe the contrast with the asynchronous case, where once agents begin to compete, they almost immediately revert back to collusion. The bias present in the asynchronous algorithm disappears in the synchronous version: actions are updated according to counterfactual returns, therefore the value of joint collusion is short-lived after competition begins. The result that in this game with synchronous learning the Q-learning algorithms converge to the Nash equilibrium is predicted by our Theorem 3, because with two price points the game has a dominant strategy.

General Bertrand. The simple model above reduces the Bertrand game to a dominant-strategy game. It is a convenient simplification for the purposes of inspecting and plotting the dynamical systems, but the theory developed in Section 4 allows us to deal with much more general models. For more price points as in Asker et al. (2022) there is no dominant strategy, but we can then apply Theorem 4.

Alice Inc. and Bob Ltd. have now constant marginal costs $c_A = c_B = 2$. They sell homogeneous goods and compete by setting prices. The set of feasible prices is composed of 100 equally spaced numbers between 0.01 and 10, inclusive. The set of prices is denoted by $P = \{p^1, \ldots, p^{100}\}$. Consumers buy from the firm with the lowest price, and demand is parametrized as

$$d_i(p_i, p_{-i}) = \begin{cases} 1 \text{ if } p_i < p_{-i} \text{ and } p_i \leq 10 \\ 1 \text{ if } p_i = p_{-i} \text{ and } p_i \leq 10 \\ 0 \text{ otherwise} \end{cases}$$

As the authors note, there are two Nash equilibria of this game, one (E_1) with $p_A = p_B = 2.0282$ and one (E_2) with $p_A = p_B = 2.1291$. The multiplicity is a consequence of the discretization of the space in equally spaced prices.

Proposition 3. *In a Bertrand oligopoly, if Alice Inc. and Bob Ltd. adopt any ε-greedy separable reinforcers with a small $\varepsilon > 0$ such that the relative speed of learning is the same across all prices, they will learn to play either $p_1 = 2.0282$ or $p_2 = 2.1291$.*

Proof. This proposition follows almost immediately from Theorem 4. The Theorem guarantees that two ε-greedy reinforcers will learn a pure-rationalizable strategy. Discretized homogeneous Bertrand games have only two pure-rationalizable strategies, the two lowest prices above marginal cost, which are also the game's Nash Strategies. □

In particular, this result shows that the parameters of the algorithms are irrelevant for the convergence result.

6.2 First-Price Auction

Consider now the setting of Banchio and Skrzypacz (2022). The authors simulate algorithmic competition in an auction environment, and they find that revenues in First and Second Price Auction are substantially different when Q-learning algorithms choose bids. While the Second Price Auction appears to be competitive, the first-price auction is prone to collusion. Additionally, they find that providing counterfactuals in the first-price auction restores competition.

First, we propose an interpretation of their findings in light of our results. Let us consider the same model, where Alice.com and Bob.net bid in a First- (Second-) Price Auction for display advertising. Both have value of \$1 for the impression being auctioned, and they have access to a set of 19 equally-spaced bids $b^i \in \{0.05, 0.1, \ldots, 0.95\}$. Banchio and Skrzypacz (2022) finds that, when the exploration parameter ε is held constant, the Second Price Auction (SPA) converges on the dominant-strategy equilibrium. In the First Price Auction (FPA) instead the bidders cycle between collusive pairs, interspersed with short "punitive" phases.

We now use our results to shed light on these findings. To this end, consider the following reduced model, where Alice.com and Bob.net each have access only to two bids. The low bid is 0.1, while the high bid is denoted by x and varies in $[0.1, 0.55]$. Under these assumptions, both games are one-dimensional parametrizations of a Prisoner's Dilemma, and can therefore be mapped to the description of Proposition 6. Figures 9a and 9b show the payoff matrix for both game formats, while Figure 9c plots the parametric region where a collusive equilibrium exists, for both FPA and SPA.

Importantly, we observe that parameters such that a pseudo-steady-state exists for a SPA also generate a pseudo-steady-state for the FPA. The payment rules of FPA and SPA parametrize the Prisoner's Dilemma differently, and in such a way that FPA facilitates collusion through coupling.

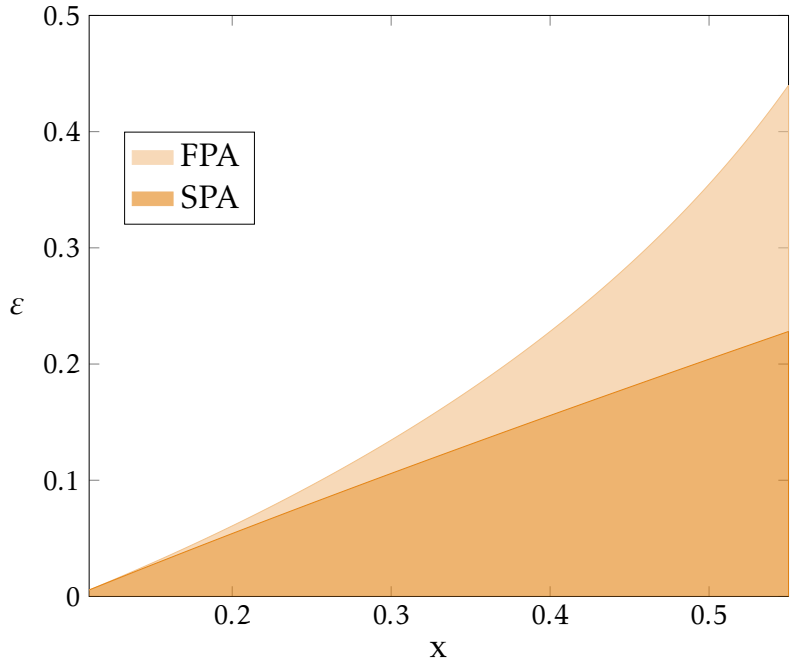
A further result of that paper shows that by providing counterfactual information to Alice.com and Bob.net the designer restores competition in the First Price Auction. Note that the FPA with 19 admissible bids has two equilibria: one (E_1) sees both Alice.com and Bob.net bidding $b_A = b_B = 0.95$, and one (E_2) sees them bidding $b_A = b_B = 0.9$. Again, we can show:

		Bob	
		0.1	x
Alice	0.1	$0.45, 0.45$	$0, \mathbf{1-x}$
	x	$\mathbf{1-x}, 0$	$\frac{1-x}{2}, \frac{1-x}{2}$

(a) Payoffs of the FPA.

		Bob	
		0.1	x
Alice	0.1	$0.45, 0.45$	$0, \mathbf{0.9}$
	x	$\mathbf{0.9}, 0$	$\frac{1-x}{2}, \frac{1-x}{2}$

(b) Payoffs of the SPA.



(c) Region of the parameter space such that a pseudo-steady-state exists for FPA and SPA, respectively.

Figure 9

Proposition 4. *In a first-price auction, if Alice.com and Bob.net adopt any ε-greedy reinforcers with small $\varepsilon > 0$ and same relative speed of learning across bids, both will learn to play either $b = 0.95$ or $b = 0.9$.*

The proof is identical to that of Proposition 3 and is therefore omitted.

7 Conclusion

This paper analyzes collusion in games played by online learning algorithms. We take a theoretical perspective and, complementing burgeoning empirical and numerical evidence, we identify the drivers of collusive behaviour, first in a Prisoner's Dilemma and then in more general settings. We first address the issue of the analytical intractability of strategic interaction among algorithms by showing they can be approximated with a dynamical system. Then we apply this framework to dominant-strategy games, and we show that (ε-)greedy algorithms can learn to collude. We identify the mechanism sustaining collusion, a form of *spontaneous coupling*: when algorithms are slow to realize the value of the competitive action, joint collusion appears more attractive. Involuntary coupling yields self-fulfilling biases in the estimates. We demonstrate this intuition in a Prisoner's Dilemma with Q-learning agents. We expect the techniques developed to analyze the simple Prisoner's Dilemma to yield insight in games with more complex strategic structure. In particular, we believe similar techniques can help understanding how AIs reach tit-for-tat strategies when given monitoring technology. A testament to the power of our method is the application of our techniques to simple auctions in the spirit of Banchio and Skrzypacz (2022), where the same mechanism generates different results across payment rules. Our derivation yields intuition that complements the empirical results.

We show that convergence on the dominant action is instead guaranteed for greedy algorithms if learning occurs at uniform rates for all actions. The simultaneous coupling at the heart of collusion appears when learning rates are uneven, but disappears when they become uniform. Uniform learning rates can be a consequence of access to additional information, as we argued in Corollary 2. Algorithms that evaluate counterfactual rewards from actions not taken learn simultaneously for all actions, and thus converge on undominated strategies.

Following these results, we design learning-robust strategy-proof mechanisms, where even algorithmic agents learn to play their dominant strategy. Learning-robustness is guaranteed by a new layer in the mechanism design: ex-post feedback provision. We show that the menu description of a mechanism is learning-robust, and it communicates

information efficiently. We suspect that these design ideas could be successfully applied to settings with regret-minimizing agents, as additional feedback simplifies regret evaluations. Finally, we validate our theoretical results by analyzing the Bertrand oligopoly introduced in Asker et al. (2022), where our framework delivers an analysis of the driving forces behind some of their experimental results.

We view our paper as contributing to the growing literature studying strategic interaction of algorithmic agents. Algorithms shape the dimensions of rationality of these decision makers, and allow us to carry out a disciplined analysis of equilibria and market design for such boundedly-rational agents. There are many other dimensions of interest in the study of strategic algorithmic interactions that we do not touch upon in this paper. For example, we focused on dominant strategy games, which intrinsically make collusion the hardest to sustain: the outcomes of games where the separation between competition and collusion is less stark remains unclear, and worthwhile to pursue. Our algorithms interact with the environment and adapt according to the feedback they receive, but many deployed market algorithms are instead trained offline. Our analysis points to correlation as a key driver of collusion, thus suggesting that offline algorithms may be less prone to collusive behavior. Another interesting aspect of algorithmic collusion is whether coordination on collusive outcomes would be even easier if algorithms were able to retain memory of recent payoff-relevant quantities. We suspect that when algorithms are enabled to learn dynamic reward-punishment strategies their collusive behavior will increase substantially, as highlighted in the literature.

An interesting question is what could a sophisticated player achieve when competing against an automated decision-maker. The manipulability of these algorithms deserves further analysis, and we believe a setting similar to the one offered in this work could prove helpful in understanding these questions. Finally, algorithmic decision-making can be seen as a form of bounded rationality. This implies that the set of implementable outcomes is, in general, wider than that of rational agents. Theorem 4 suggests that arguments similar to Abreu and Matsushima (1992) could prove useful in enlarging the set of implementable outcomes; characterizing the set of implementable outcomes for purely adaptive decision makers is beyond the scope of this paper, but of independent interest.

References

Abreu, D. and H. Matsushima (1992): "Virtual implementation in iteratively undominated strategies: complete information," *Econometrica*, 993–1008.

Aouad, A. and A. Van den Boer (2021): "Algorithmic Collusion in Assortment Games," Working paper.

Asker, J., C. Fershtman, and A. Pakes (2022): "Artificial Intelligence, Algorithm Design and Pricing," *American Economic Review, P&P*, forthcoming.

Assad, S., R. Clark, D. Ershov, and L. Xu (2021): "Algorithmic Pricing and Competition: Empirical Evidence from the German Retail Gasoline Market," Working paper.

Banchio, M. and A. Skrzypacz (2022): "Artificial Intelligence and Auction Design," Working paper.

Benaim, M. (1996): "A Dynamical System Approach to Stochastic Approximations," *SIAM Journal of Control and Optimization*, 34, 437–472.

Brown, Z. Y. and A. MacKay (2021): "Competition in Pricing Algorithms," *American Economic Journal: Microeconomics*, forthcoming.

Börgers, T. and R. Sarin (1997): "Learning Through Reinforcement and Replicator Dynamics," *Journal of Economic Theory*, 77, 1–14.

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

Chen, Y. (2002): "A family of supermodular Nash mechanisms implementing Lindahl allocations," *Economic Theory*, 19, 773–790.

Competition Bureau (2018): "Big Data and Innovation: Implications for Competition Policy in Canada," Discussion paper.

di Bernardo, M., C. Budd, A. R. Champneys, and P. Kowalczyk (2008): *Piecewise-smooth dynamical systems: theory and applications*, vol. 163, Springer Science & Business Media.

Erev, I., Y. Bereby-Meyer, and A. E. Roth (1999): "The effect of adding a constant to all payoffs: experimental investigation, and implications for reinforcement learning models," *Journal of Economic Behavior & Organization*, 39, 111–128.

Erev, I. and A. E. Roth (1998): "Predicting How People Play Games: Reinforcement Learning in Experimental Games with Unique, Mixed Strategy Equilibria," *The American Economic Review*, 88, 848–881.

Filippov, A. F. (1988): *Differential Equations with Discontinuous Right-Hand Sides*, Springer Science & Business Media.

Gomes, E. R. and R. Kowalczyk (2009): "Dynamic Analysis of Multiagent Q-Learning with ϵ-Greedy Exploration," in *Proceedings of the 26th Annual International Conference on Machine Learning*, 369–376.

Halfin, S. and W. Whitt (1981): "Heavy-traffic limits for queues with many exponential servers," *Operations research*, 29, 567–588.

Hammond, P. J. (1979): "Straightforward Individual Incentive Compatibility in Large Economies," *The Review of Economic Studies*, 46, 263–282.

Hansen, K., K. Misra, and M. Pai (2021): "Algorithmic Collusion: Supra-Competitive Prices via Independent Algorithms," *Marketing Science*, 40, 1–12.

Harrison, J. M. and M. I. Reiman (1981): "Reflected Brownian motion on an orthant," *The Annals of Probability*, 9, 302–308.

Hopkins, E. and M. Posch (2005): "Attainability of boundary points under reinforcement learning," *Games and Economic Behavior*, 53, 110–125.

Iglehart, D. L. (1965): "Limiting diffusion approximations for the many server queue and the repairman problem," *Journal of Applied Probability*, 2, 429–441.

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

Kurtz, T. G. (1970): "Solutions of ordinary differential equations as limits of pure jump Markov processes," *Journal of Applied Probability*, 7, 49–58.

Lamba, R. and S. Zhuk (2022): "Pricing with Algorithms," working paper.

Leisten, M. (2022): "Algorithmic Competition, with Humans," Working paper.

Leonardos, S. and G. Piliouras (2022): "Exploration-exploitation in multi-agent learning: Catastrophe theory meets game theory," *Artificial Intelligence*, 304.

Lerer, A. and A. Peysakhovich (2017): "Maintaining cooperation in complex social dilemmas using deep reinforcement learning," *arXiv preprint arXiv:1707.01068*.

Mathevet, L. (2010): "Supermodular mechanism design," *Theoretical Economics*, 5, 403–443.

Mertikopoulos, P. and W. H. Sandholm (2016): "Learning in games via reinforcement and regularization," *Mathematics of Operations Research*, 41, 1297–1324.

Mitzenmacher, M. (2001): "The power of two choices in randomized load balancing," *IEEE Transactions on Parallel and Distributed Systems*, 12, 1094–1104.

Musolff, L. A. (2021): "Algorithmic Pricing Facilitates Tacit Collusion: Evidence from E-Commerce," Working paper.

OECD (2017): "Algorithms and Collusion: Competition Policy in the Digital Age," Technical report.

Parkes, D. C. (2004): "On learnable mechanism design," in *Collectives and the Design of Complex Systems*, Springer, 107–131.

Sandholm, W. H. (2005): "Negative externalities and evolutionary implementation," *The Review of Economic Studies*, 72, 885–915.

Tumer, K. and N. Khani (2009): "Learning from actions not taken in multiagent systems," *Advances in Complex Systems*, 12, 455–473.

Tuyls, K., P. J. Hoen, and B. Vanschoenwinkel (2005): "An Evolutionary Dynamical Analysis of Multi-Agent Learning in Iterated Games," *Autonomous Agents and Multi-Agent Systems*, 12, 115–153.

Wager, S. and K. Xu (2021): "Diffusion asymptotics for sequential experiments," *arXiv preprint arXiv:2101.09855*.

Wein, L. M. (1992): "Dynamic scheduling of a multiclass make-to-stock queue," *Operations Research*, 40, 724–735.

Wunder, M., M. L. Littman, and M. Babes (2010): "Classes of Multiagent Q-learning Dynamics with epsilon-greedy Exploration," in *Proceedings of the 27th Annual International Conference on Machine Learning*, 1167–1174.

Appendix A

We restate Theorem 1 in its more formal version.

Theorem (1). *Let (θ, π) be a collection of adaptive agents that satisfy Assumption A1 individually, and let the domain $K \subset \mathbb{R}^{\sum_i d_i}$ of θ be a compact set. Let $(H_j)_{j\in J}$ be the collection of θ's maximal Lipschitz-continuity domains, that is, let H_j be the largest open set such that θ is Lipschitz over H_j and there is a discontinuity on $\overline{H}_j$. For all $j \in J$ the collection of Cauchy problems*

$$\begin{cases} \frac{d\Theta^i(t)}{dt} = \alpha \mathbb{E}_{\pi^i, \pi^{-i}}\left[T^i(\pi^i, r(\pi^i, \pi^{-i}), \Theta^i(t))\right] \\ \Theta^i(0) = y_0^i \end{cases}$$

has a solution Θ^i for all i over H_j for all $y_0 \in H_j$. There exists a sequence of processes $\{\theta^n\}_{n\in\mathbb{N}}$ such that:

- $\mathbb{E}\left[\theta^1(\tau(k))\right] = \mathbb{E}\left[\theta(k)\right]$ *for all k,* $\tau(k) = \inf\left\{t \mid \theta^1(t) \text{ jumped } k \text{ times}\right\}$,
- *the infinitesimal generators $\mathcal{A}T_n(\theta)$ are all identical to $\mathcal{A}T_1(\theta)$ for all $\theta \in H_j$ and n,*
- $\lim_{n\to\infty} P\left\{\sup_{t\le T}\left\|\theta^n(t) - \Theta(t)\right\| > \eta\right\} = 0$ *for all $T \ge 0$ and $\eta > 0$ such that $\{\Theta(t)\}_{t\le T} \subset H_j$.*

Proof of Theorem 1. The existence of a solution for the Cauchy problem is guaranteed by Assumption A1 and the restriction to the maximal continuity domains H_j. In particular, notice that one can write $T^i(\pi^i(\theta), r(\pi^i(\theta^i), \pi^{-i}(\theta^{-i}), \theta^i)$ as a map $\hat{T}(\theta, Y)$ where Y is a random variable representing the uncertainty introduced by the policies π.

We can divide the proof of Theorem 1 in two main steps:

1. Finding the correct continuous-time embedding of the adaptive algorithm θ;
2. Identifying a scaling that guarantees limits are well-defined.

First, let us fix a compact ball of radius r in $\mathbb{R}^{\sum_i d_i}$. We will consider the set $E = H \cap B(r)$ with the Borel intersection sigma algebra. Since we can choose r to be as large as we want, the approximation will hold for any finite values of θ. Let us add one component to the vector θ, in position $\sum_i d_i + 1$, which will keep track of the iteration k.

As far as the first step is concerned, let us define a Poisson process N_1 of rate $\lambda_1 = 1$. Consider the sequence of (stochastic) arrival times $0 < \tau_1 < \tau_2 < \tau_3 < \ldots$. We define the process $\theta^1(t)$ as

$$\theta^1(t) = \theta(k) \qquad \text{if } \tau_{k+1} > t \ge \tau_k$$

for all times $t \geq 0$. The process $\theta^1(t)$ is a compound Poisson process, cadlag and Markov. Naturally, its $\sum_i d_i + 1$ component always coincides with the iteration k. At arrival times the adaptive algorithm is equal to its continuous time equivalent θ^1, which proves item 1 of the Theorem.

We now want to increase the pace of the updates while retaining the same uncertainty in expectation. Intuitively, we can "speed up" the Poisson arrivals, but we also need to "dampen" the jumps accordingly, otherwise the process will diverge to infinity. Formally, we consider a sequence of Continuous-Time Markov Chains indexed by $n \in \mathbb{N}$ as follows:

- The jump rate λ_n is defined as $\lambda_n = n$.
- At each jump, the update in the first $\sum_i d_i$ components is[15]

$$\theta^n(t) - \theta^n(t^-) = \frac{1}{n}\hat{T}(\theta^n(t^-), Y)$$

- At each jump, the update in the coordinate $\sum_i d_i + 1$ is

$$\theta^n(t) - \theta^n(t^-) = \frac{1}{n}$$

Intuitively, the updates of the original process θ are scaled down by a factor n and the last coordinate keeps track of how many updates have occurred scaled by n.

Consider then the measure $\mu^n(x, dz)$ of updates at x, with

$$\mu_n(x, dz) = \mathbb{P}\left\{\theta^n(\tau_n) \in dz | \theta^n(0) = x\right\}$$

where τ_n is the first exit time of θ^n from x. We define the component-wise function

$$F_n(x)^m = \lambda_n \int (z^m - x^m)\mu_n(x, dz), \tag{5}$$

which intuitively describes the expected jump of θ^n from x along the m-th component over one unit of time. In fact, F_n can be rewritten as

$$F_n(x)^m = n \int \frac{\alpha}{n}\hat{T}^m(x, Y)\mu = \int \alpha \hat{T}^m(x, Y)\mu.$$

We chose the scaling in such a way that the function F_n is independent of n. The function F_n is exactly the infinitesimal generator of the compound Poisson process $\theta^n(t)$, therefore

[15] Note that we omit the dependence on the iteration since iterations are now part of the process θ^n.

item (2) of the Theorem is proved.

Let $F(x) \coloneqq \lim_{n\to\infty} F_n(x)$. It is clear that $F(x) = F_n(x)$ for all n. Moreover the function $F(x)$ is Lipschitz in every component. We will need a technical lemma:

Lemma 1. *Let E be a compact set in $\mathbb{R}^{|\mathcal{I}|\times|\mathcal{S}|\times|\mathcal{A}|}$. There exists a sequence $\{\varepsilon_n\}_n > 0$ with $\lim_{n\to\infty} \varepsilon_n = 0$ such that*

$$\lim_{n\to\infty} \sup_{x\in E} \lambda_n \int_{|z-x|>\varepsilon_n} |z-x|\mu_n(x,dz) = 0$$

Moreover,

$$\sup_n \sup_{x\in E} \lambda_n \int_E |z-x|\mu_n(x,dz) < \infty$$

Proof. Since E is compact $\hat{T}(\theta, Y)$ must be bounded. Let $M = \sup_{\theta\in E} T(\theta, Y)$ across dimension, and note that $M < +\infty$. Let then $\{\frac{M}{n}\}_n$ be a sequence satisfying the assumptions of the Lemma, and notice that $\int_{|z-x|>\varepsilon_n} |z-x|\mu_n(x,dz) = 0$ for all x and n. We thus proved the first claim. The second claim follows a simple observation: $|z-x|\mu_n(x,dz) \le \frac{M}{n}$ for all x. Since $\lambda_n = n$, we obtain that

$$\sup_n \sup_{x\in E} \lambda_n \int_E |z-x|\mu_n(x,dz) = M < \infty$$

which concludes the proof. □

This lemma verifies one of the conditions of the following Theorem by Kurtz (1970):

Theorem 2.11. *Suppose there exists $E \subset \mathbb{R}^k$, a function $F\colon E \to \mathbb{R}^k$ and a constant M such that $|F(x) - F(y)| \le M|x-y|$ for all $x, y \in E$ and*

$$\lim_{n\to\infty} \sup_{x\in E_n\cup E} |F_n(x) - F(x)| = 0$$

Let $X(t, x_0), 0 \le t \le T, x_0 \in E$ satisfy

$$X(0, x_0) = x_0, \qquad \dot{X}(t, x_0) = F(X(t, x_0))$$

Suppose additionally that the sequence F_n satisfies the conditions of Lemma 1, then for every $\eta > 0$

$$\lim_{n\to\infty} P\Big\{\sup_{t\le T} |X_n(t) - X(t, x_0)| \ge \eta\Big\} = 0$$

If $X(t, x_0) = \Theta$, we can verify that the assumptions of the Theorem hold:

- since $\hat{T}$ is Lipschitz, and $F_n = F$ are integrals of Lipschitz functions, it is clear that $|F(x) - F(y)| \leq M|x - y|$ holds,
- $\lim_n \sup_{x \in H} |F_n(x,t) - F(x,t)| = 0$ is satisfied by definition of $F_n = F$,
- the conditions of *Lemma* 1 are verified.

Then, Theorem 2.11 implies that for every $\eta > 0$

$$\lim_{n \to \infty} P\Big\{ \sup_{t \leq T} |\theta^n(t) - \Theta(t)| \geq \eta \Big\} = 0$$

which proves item 3 of the Theorem and concludes the proof.

As advanced at the beginning, the process Θ is a deterministic process with all uncertainty collapsed into the drift component. □

Proof of Proposition 1 The existence of the equilibrium q_D^{eq} follows directly from setting the field over $\omega_{D,D}$ to zero.

We prove existence of q_C^{eq} and its related property for one agent; by symmetry, the other agent's Q-values enjoy the same properties. The boundary is defined as $\Sigma = \{q \in \mathbb{R}^2 \colon c \cdot q = 0\}$ where $c = (1, -1)$ and $\cdot$ denotes the usual dot product. Using the Filippov convention, we can further divide Σ in three regions:

- a *crossing region*, $\Sigma^c = \{q \colon (c \cdot (A_C q + b_C))(c \cdot (A_D q + b_D)) > 0\}$
- a *repulsive region*, $\Sigma^r = \{q \colon c \cdot (A_C q + b_C) > 0,\ c \cdot (A_D q + b_D) < 0\}$
- a *sliding region*, $\Sigma^s = \{q \colon c \cdot (A_C q + b_C) < 0,\ c \cdot (A_D q + b_D) > 0\}$

where we have

$$A_C = \begin{bmatrix} \alpha\left(1 - \frac{\varepsilon}{2}\right)(\gamma - 1) & 0 \\ \alpha\gamma\frac{\varepsilon}{2} & -\alpha\frac{\varepsilon}{2} \end{bmatrix} \qquad A_D = \begin{bmatrix} -\alpha\frac{\varepsilon}{2} & \alpha\gamma\frac{\varepsilon}{2} \\ 0 & \alpha\left(1 - \frac{\varepsilon}{2}\right)(\gamma - 1) \end{bmatrix},$$

and

$$b_C = \begin{bmatrix} \alpha\left(1 - \frac{\varepsilon}{2}\right)\left(2 - \frac{\varepsilon}{2}\right)g \\ \alpha\frac{\varepsilon}{2}\left(2 + g - g\frac{\varepsilon}{2}\right) \end{bmatrix} \qquad b_D = \begin{bmatrix} \alpha\left(1 + \frac{\varepsilon}{2}\right)\frac{\varepsilon}{2}g \\ \alpha\left(1 - \frac{\varepsilon}{2}\right)\left(2 + \frac{\varepsilon}{2}g\right) \end{bmatrix}.$$

We can define the sliding solution as the field $\frac{d\mathbf{Q}}{dt} = F^s(\mathbf{Q})$ over the sliding region where

$$F^s(\mathbf{Q}) = \frac{c \cdot (A_D\mathbf{Q} + b_D)(A_C\mathbf{Q} + b_C) - c \cdot (A_C\mathbf{Q} + b_C)(A_D\mathbf{Q} + b_D)}{c \cdot (A_D\mathbf{Q} + b_D) - c \cdot (A_C\mathbf{Q} + b_C)}$$

The relative time spent on $\omega_{C,C}$ at point $\mathbf{Q}$ is defined as

$$\tau_C = \frac{c \cdot (A_D\mathbf{Q} + b_D)}{c \cdot (A_D\mathbf{Q} + b_D) - c \cdot (A_C\mathbf{Q} + b_C)}$$

The sliding vector field becomes

$$\frac{d\mathbf{Q}_j}{dt} = \frac{\alpha\left(\frac{1}{2}\varepsilon g(2-\varepsilon)(g-1) + (2g + (\gamma-1)\mathbf{Q}_j)(2 + (\gamma-1)\mathbf{Q}_j)\right)}{2(1 + g + (\gamma-1)\mathbf{Q}_j)}$$

for every direction j. By setting the field equal to zero and solving for $\mathbf{Q}_j$, we find that there is an equilibrium at

$$q_{C,j}^{eq} = \frac{1 + g - \sqrt{(g-1)(g-1-\varepsilon g + \frac{\varepsilon^2 g}{2})}}{(\gamma-1)}$$

for all j. This equation has a feasible solution for all $\varepsilon < 1 - \sqrt{\frac{2-g}{g}}$. □

Proof of Corollary 1. The result follows immediately from the proof of Proposition 1. In particular, it is sufficient to compute

$$\tau_C = \frac{c \cdot (A_D\mathbf{Q} + b_D)}{c \cdot (A_D\mathbf{Q} + b_D) - c \cdot (A_C\mathbf{Q} + b_C)}$$

at $\mathbf{Q} = q_C^{eq}$. □

Proof of Theorem 2. First off, note that there is a given ordering of rewards in a Prisoner's Dilemma:

$$r(D,C) > r(C,C) > r(D,D) > r(C,D)$$

We will prove existence of an equilibrium of the following system:

$$\begin{cases} \dot{\theta}^C = \alpha\left[U(\theta^C, r(C,C)) + V(\theta)\right] \\ \dot{\theta}^D = (1-\alpha)\left[U(\theta^D, r(D,C)) + V(\theta)\right] \end{cases}$$

in the half-plane where C is the preferred action, and

$$\begin{cases} \dot{\theta}^C = (1-\alpha)\Big[U(\theta^C, r(C,D)) + V(\theta)\Big] \\ \dot{\theta}^D = \alpha\Big[U(\theta^D, r(D,D)) + V(\theta)\Big] \end{cases}$$

in the half-plane where D is the preferred action. Note that we are assuming WLOG that the learning speeds sum to 1, since all it matters is the relative speed of learning. To start, let us assume α is identical across the half planes.

We want to show that there exist an α and a θ^* such that:

$$\begin{cases} \alpha U(\theta^*, r(C,C)) + (1-\alpha)U(\theta^*, r(C,D)) + V(\theta^*) = 0 \\ (1-\alpha)U(\theta^*, r(D,C)) + \alpha U(\theta^*, r(D,D)) + V(\theta^*) = 0 \end{cases} \tag{6}$$

Because we used the same α on both sides, we are simply looking for a translation of Equation (6), therefore we can develop the argument disregarding the $V(\theta^*)$ component. For a given θ, we can write this problem as a convex combination of two vectors:

$$\alpha\vec{x} + (1-\alpha)\vec{y} = \vec{0}$$

where

$$\vec{x}(\theta) = \begin{bmatrix} U(\theta, r(C,C)) \\ U(\theta, r(D,D)) \end{bmatrix} \qquad \vec{y}(\theta) = \begin{bmatrix} U(\theta, r(C,D)) \\ U(\theta, r(D,C)) \end{bmatrix}$$

Let θ^1 be the value of θ such that $U(\theta^1, r(C,C)) = 0$. Then, using the monotonicity of U in rewards, the two vectors $\vec{x}, \vec{y}$ computed in θ^1 will be positioned as in Figure 10. Let θ^2 instead be the value of θ such that $U(\theta^2, r(C,D)) = 0$. Again, the two vectors $\vec{x}, \vec{y}$ are positioned as in Figure 10. Notice that by monotonicity of U in its first component, $\theta^1 > \theta^2$. The same assumption guarantees that the lines $\vec{y}(\theta^1 + (\theta^2 - \theta^1)t)$ and $\vec{x}(\theta^1 + (\theta^2 - \theta^1)t)$ lie on the left and right of the vertical axis, respectively.

Define $f : [0,1] \to \mathbb{R}^2$ as

$$f(t) = \begin{cases} (1-4t)\vec{x}(\theta^1) + 4t\vec{y}(\theta^1) & t \in \left[0, \frac{1}{4}\right] \\ \vec{y}\Big(\theta^1 + (\theta^2 - \theta^1)(4t-1)\Big) & t \in \left[\frac{1}{4}, \frac{1}{2}\right] \\ (3-4t)\vec{y}(\theta^2) + (4t-2)\vec{x}(\theta^2) & t \in \left[\frac{1}{2}, \frac{3}{4}\right] \\ \vec{x}\Big(\theta^1 + 4(\theta^2 - \theta^1)(1-t)\Big) & t \in \left[\frac{3}{4}, 1\right] \end{cases}$$

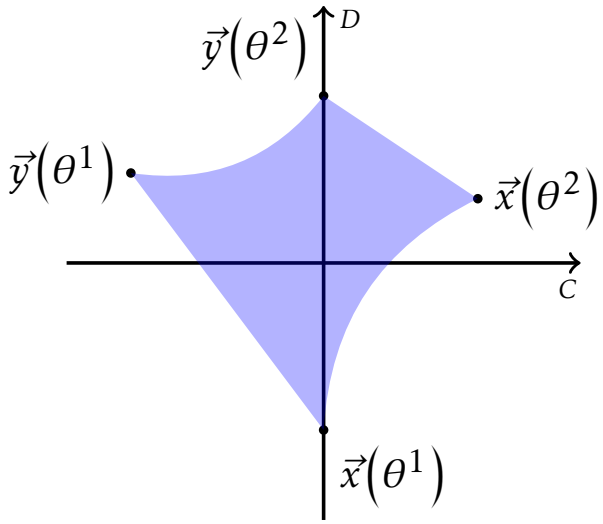


Figure 10: Homotopy

Note that, by continuity of U, $f(t)$ is a loop based in $x(\theta^1)$. We can show that f is null-homotopic by providing the following simple homotopy $H\colon [0,1]\times[0,1]\to\mathbb{R}^2$.

$$H(t,s)=\begin{cases}(1-4ts)\vec{x}(\theta^1)+4ts\vec{y}(\theta^1) & t\in\left[0,\frac{1}{4}\right]\\ (1-s)\vec{x}\left(\theta^1+s(4t-1)(\theta^2-\theta^1)\right)+s\vec{y}\left(\theta^1+s(4t-1)(\theta^2-\theta^1)\right) & t\in\left[\frac{1}{4},\frac{1}{2}\right]\\ (1-s(3-4t))\vec{x}\left((1-s)\theta^1+s\theta^2\right)+s(3-4t)\vec{y}\left((1-s)\theta^1+s\theta^2\right) & t\in\left[\frac{1}{2},\frac{3}{4}\right]\\ \vec{x}\left(\theta^1+4s(\theta^2-\theta^1)(1-t)\right) & t\in\left[\frac{3}{4},1\right]\end{cases}$$

We verify that H is a homotopy between $f(t)$ and the constant path at $\vec{x}(\theta^1)$.

- $H(0,s)=H(1,s)=\vec{x}(\theta^1)$
- $H(t,0)=\vec{x}(\theta^1)$
- $H(t,1)=f(t)$

Suppose now that there is no pair t,s such that $H(t,s)=(0,0)$. Then, by continuity it must be the case that there is an open neighborhood $V\ni(0,0)$ such that for all points $z\in V$, $z\notin Im(H)$. Note that we can restrict the loop f to a the domain $\mathbb{R}^2\setminus V$, and because $V\notin Im(H)$ we can restrict the homotopy to a homotopy $H\colon[0,1]^2\to\mathbb{R}^2\setminus V$. Thus we proved that a loop around the open set V is null-homotopic, which is equivalent to proving that $\mathbb{R}^2\setminus V$ is simply connected, a contradiction.

Therefore, there exists a pair $\bar{t}, \bar{s}$ such that $H(\bar{t}, \bar{s}) = (0, 0)$. There is a bijective transformation between t, s and θ, α over their respective domains, which guarantees that the system of Equation (6) is satisfied.

Notice that we allowed for $\alpha \in (0, 1)$ so far. However, it makes little sense to allow for $\alpha < \frac{1}{2}$: the algorithm would learn about actions it considers suboptimal faster than those he considers best. Fortunately, we observe the following:

$$\frac{1}{2}U(\theta, r(C,C)) + \frac{1}{2}U(\theta, r(C,D) < \frac{1}{2}U(\theta, r(D,C)) + \frac{1}{2}U(\theta, r(D,D).$$

This inequality follows from the ordering of rewards in a Prisoner's Dilemma game. This in particular implies that the line $\frac{1}{2}\vec{x}(\theta^1 + (\theta^2 - \theta^1)t) + \frac{1}{2}\vec{y}(\theta^1 + (\theta^2 - \theta^1)t)$ lies above the 45 degree line. Thus, we can restrict the homotopy to the parameter space $\alpha \in [\frac{1}{2}, 1]$ without affecting the result. This guarantees the existence of a parameter $\alpha > \frac{1}{2}$ which sets the sliding vector field to zero.

Now, notice that in the previous construction in Equation (6) we solved for α and θ^* such that the local time was equal on both sides of the switching boundary. We can relax this assumption so that Equation (6) becomes

$$\begin{cases} \alpha\tau U(\theta^*, r(C,C)) + (1-\alpha)(1-\tau)U(\theta^*, r(C,D)) = 0 \\ (1-\alpha)\tau U(\theta^*, r(D,C)) + \alpha(1-\tau)U(\theta^*, r(D,D)) = 0. \end{cases} \tag{7}$$

Following the previous construction, we get

$$\vec{x}(\theta) = \begin{bmatrix} \tau U(\theta, r(C,C)) \\ (1-\tau)U(\theta, r(D,D)) \end{bmatrix} \qquad \vec{y}(\theta) = \begin{bmatrix} (1-\tau)U(\theta, r(C,D)) \\ \tau U(\theta, r(D,C)) \end{bmatrix}$$

The points $\vec{x}(\theta^i), \vec{y}(\theta^i)$ for $i = 1, 2$ each remain on their respective quadrants, enabling us to construct the same, rescaled, homotopy for this case. Therefore, for each τ we can identify a pair $\alpha^\tau, \theta^*(\tau)$ such that Equation (7) is satisfied.

Not all solutions to Equation (7) are steady-states however. Recall from our construction in Section 3 that we need to verify a sliding condition: the normal components of the two vector fields to the switching surface must have opposite sign and must be attractive. In equations, this translates to the following system which needs to be satisfied:

$$\begin{cases} (1-\alpha^\tau)\tau U(\theta^*(\tau), r(D,C)) - \alpha\tau U(\theta^*(\tau), r(C,C)) \geq 0 \\ \alpha^\tau(1-\tau)U(\theta^*(\tau), r(D,D)) - (1-\alpha^\tau)(1-\tau)U(\theta^*(\tau), r(C,D)) \leq 0. \end{cases} \tag{8}$$

We need a preparatory lemma:

Lemma 2. *The only region of θ where Equation (7) can be verified is such that*

$$U(\theta, r(D,C)) > U(\theta, r(C,C)) \geq 0 > U(\theta, r(D,D)) > U(\theta, r(C,D))$$

Proof. The statement follows immediately from inspection of Figure 10. Since the path $\vec{y}(\theta^1 + (\theta^2 - \theta^1)t)$ for all t falls within the second quadrant, any $\vec{x}(\theta^*)$ which satisfies Equation (7) must fall within the fourth quadrant, which directly implies the result. □

Now, rearranging Equation (7) we derive the following equalities:

$$\begin{cases} -(1-\alpha^\tau)(1-\tau)U(\theta^*(\tau), r(C,D)) = \alpha^\tau \tau U(\theta^*(\tau), r(C,C)) \\ -(1-\alpha^\tau)\tau U(\theta^*(\tau), r(D,C)) = \alpha^\tau (1-\tau)U(\theta^*(\tau), r(D,D)) \end{cases}$$

Substituting these in Equation (8) and rearranging, we obtain

$$\begin{cases} \alpha^\tau \tau U(\theta^*(\tau), r(C,C)) + \alpha^\tau (1-\tau)U(\theta^*(\tau), r(D,D)) \leq 0 \\ \alpha^\tau (1-\tau)U(\theta^*(\tau), r(D,D)) + \alpha^\tau \tau U(\theta^*(\tau), r(C,C)) \leq 0 \end{cases}$$

Thus, only one condition needs to be satisfied to guarantee sliding:

$$\tau U(\theta^*(\tau), r(C,C)) + (1-\tau)U(\theta^*(\tau), r(D,D)) \leq 0$$

Lemma 2 guarantee that a solution to this inequality exists for τ sufficiently close to 0. In particular, there exists a $\overline{\tau}$ such that for all $\tau \leq \overline{\tau}$ we obtain sliding. Therefore there exists an open set of parameters $\{\alpha^\tau \mid \tau \leq \overline{\tau}\}$ such that a sliding steady-state exists. Now, we can perturb the value of α on either side of the sliding boundary. The region we derived depends continuously from α, in particular from α^C and α^D. Therefore, the same result holds for small perturbations of α into different α^C and α^D, concluding the proof of the Theorem.

□

Proof of Theorem 3. We set to prove that, in the limit, the statistic θ^n of the dominant action dominates the statistic θ^i of any other non-dominant action. First of all, since α^{a_i} is identical across actions, the evolution in time of θ is shifted by $V(\theta)$, but V won't affect the relative rankings of the actions' estimates. Therefore, we drop V in the rest of the proof, and we focus on U.

Regardless of the opponent's policy, the reinforcer in every step observes a return in hindsight for every action. We denote by $r_n(t)$ the return from playing action n in period t, whatever the opponents' actions are. We consider the evolution of the weights pairwise: θ^n and θ^i for all i. By assumption, for any sequence of actions taken by the opponent(s), $r_n(t) \geq r_i(t)$. In particular, Assumption A3 guarantees that there exists a $T > 0$ such that $r_n(t) > r_i(t)$ for any $t > T$. Thus, in the limit the derivative $\dot{\theta}^n$ strictly dominates $\dot{\theta}^i$ in each point (x,t): $U(x, r_n(t)) > U(x,, r_i(t))$. In particular, there exists a ε such that $U(x, r_n(t)) > U(x,, r_i(t)) + \varepsilon$.

Suppose now that for some $t \geq 0$, $\theta^n(t) \geq \theta^i(t)$. We prove that it can never be the case that $\theta^i(T) > \theta^n(T)$ for some $T > t$. $\theta^n(t)$ is a solution to the ODE given by $\dot{\theta}^n(t) = U(\theta^n(t), r_n(t))$ and

$$U(\theta^n(t), r_n(t)) \geq U(\theta^n(t), r_i(t)) = U(\theta^i(t), r_i(t))$$

Thus, $\dot{\theta}^i(t) \leq U(\theta^i(t), r_n(t))$. The fundamental theorem of differential inequalities guarantees that the solution $\theta^i(T)$ is bounded above by $\theta^n(T)$ for all $T > t$ on its maximal domain.

Consider instead the case where $\theta^i(T) > \theta^n(T)$. From the definition of Reinforcer, we have that

$$U(\theta^i(T), r_i(T)) \leq U(\theta^i(T), r_n(T)) < U(\theta^n(T), r_n(T))$$

We want to show that there exists a $t > T$ such that $\theta^n(t) = \theta^i(t)$. To this end, suppose by contradiction that $\forall t > T$, $\theta^i(t) > \theta^n(t)$. Observe that the previous inequalities imply that

$$\left.\frac{d}{ds}\right|_{s=t} (\theta^i(s) - \theta^n(s)) < 0 \qquad \forall t > T. \tag{9}$$

Then, $(\theta^i(t) - \theta^n(t))$ is a monotone decreasing function of time. Because the algorithm is bounded, it must be the case that

$$\lim_{t \to +\infty} (\theta^i(t) - \theta^n(t)) = b \geq 0. \tag{10}$$

From the definition of limit and the monotonicity of the derivative, for any ϵ there exists a $t' > T$ such that, for all $t > t'$,

$$\theta^n(t) + b \leq \theta^i(t) < \theta^n(t) + b + \epsilon.$$

Thus, strict monotonicity of the reinforcer's update implies that there exists a $\delta > 0$ such

that

$$U(\theta^i(t), r_i(t)) \leq U(\theta^n(t), r_i(t)) < U(\theta^n(t), r_n(t)) + \delta. \tag{11}$$

Note that, by Equation (10) and the monotonicity given by Equation (9), the limit of the derivative of the difference $\theta - \theta^n$ must converge to 0:

$$\lim_{t\to+\infty} (\theta^i(t) - \theta^n(t))' = \lim_{t\to+\infty} (U(\theta^i(t), r_i(t)) - U(\theta^n(t), r_n(t)))' = 0$$

This is a contradiction of Equation (11). Then, there must exist a T such that $\theta^i(T) = \theta^n(T)$. From the first part of the proof, this implies that $\forall t > T$ we have $\theta^i(T) \leq \theta^n(T)$, and this completes the proof, □

Proof of Theorem 4. The proof is largely based upon Theorem 3. We will show that there is always a $\mathcal{T}$ such that the actions taken after $\mathcal{T}$ survive an IESDS procedure.

Let a^i be a strategy for player i which is dominated by b^i in the game G. With the same argument of the proof of Theorem 3 we can show that it must be the case that, in the limit, $\theta^{b^i} > \theta^{a^i}$. Equivalently, there exists a $\mathcal{T}_1$ such that for all $t \geq \mathcal{T}_1$ we have $\theta^{b^i}(t) > \theta^{a^i}(t)$. Therefore, after time $\mathcal{T}_1$ it is as if the agents were playing in a reduced game $G^1 = (N, (A_i^1)_i, (u_i)_i)$, where $A_i^1 = A_i \setminus \{a^i\}$ and $A_j^1 = A_j$ for all $j \neq i$.[16] We can now apply the same idea to this new reduced game G^1 and eliminate a strictly dominated strategy in this reduced game. While IESDS eliminates strategies "in place", the algorithms abandon dominated strategies over time, reducing the effective game played. Of course, the components of θ that correspond to dominated actions keep getting updated, but note that if an action is dominated given a larger subset of opponent's strategies, it is also (weakly) dominated given a smaller subset. Therefore, following the usual differential inequality argument, the θ corresponding to a dominated action will always remain below that of actions surviving IESDS. We then define $\mathcal{T}$ as the largest among the $\mathcal{T}_k$ that correspond to an action being eliminated, and this concludes the proof. □

Appendix B

In this Appendix we discuss the chaotic theory of the system in Section 3. Chaos theory studies non-linear systems whose trajectories appear stochastic but are the result of deterministic laws of motion. The most commonly accepted definition of chaotic behavior

[16] Of course, Assumption A3 guarantees that even action a^i will be played with some positive probability, but the statement of Theorem 4 guarantees that the probability is small enough to not affect the order of the expected rewards.

is high sensitivity to small perturbations in initial conditions. We plot a sample trajectory of the system in Figure 11. The system behaves erratically and unpredictably along its deterministic trajectory, and it is highly sensitive to its initial condition. The "butterfly" traced by the trajectory appears hard to characterize. However, the heatmap shows that most of the time is spent in a region where the estimates of cooperation and defection coincide.

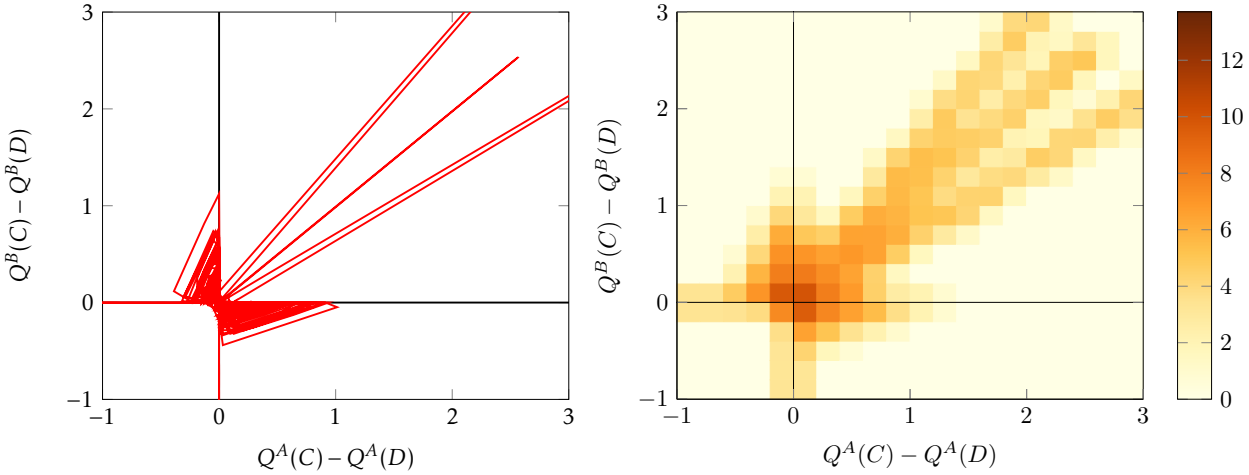


Figure 11: A chaotic trajectory of the system of Equation (4) in a 2-D representation. On the left, we depict the motion in the difference space. Agents appear chaotically attracted to a butterfly-like motion around the origin. The figure on the right represents the logarithmic density of time spent in a given square cell of side 0.2. Effectively, over 90% of time is spent in the square centered on the origin.

Figure 12 shows the effect of a small initial perturbation on two trajectories of the 4-D system along its first component.

The deterministic chaos we observe makes anly stability analysis impossible. However, as shown in Figure 11, the system spends most of its time in a region near the double sliding boundary. That is, most of the time the system will show equal values for cooperation and defection for both Alice and Bob. The sliding analysis we carried out in the symmetric case is not too far off the mark in the 4-D case as well, as one can see in Figure 7b: the gap between the predicted local time and the realized local time is due to the asymmetry that always arises in the discrete algorithmic system.

Moreover, the double sliding plane is locally attractive. When both agents cooperate, both are drawn towards defection, and thus towards sliding. Suppose Alice is the first to hit the switching surface, and thus to slide. Bob's vector fields adapt to Alice's new, sliding behavior, but the switching surface retains its attractivity. Bob will join Alice in

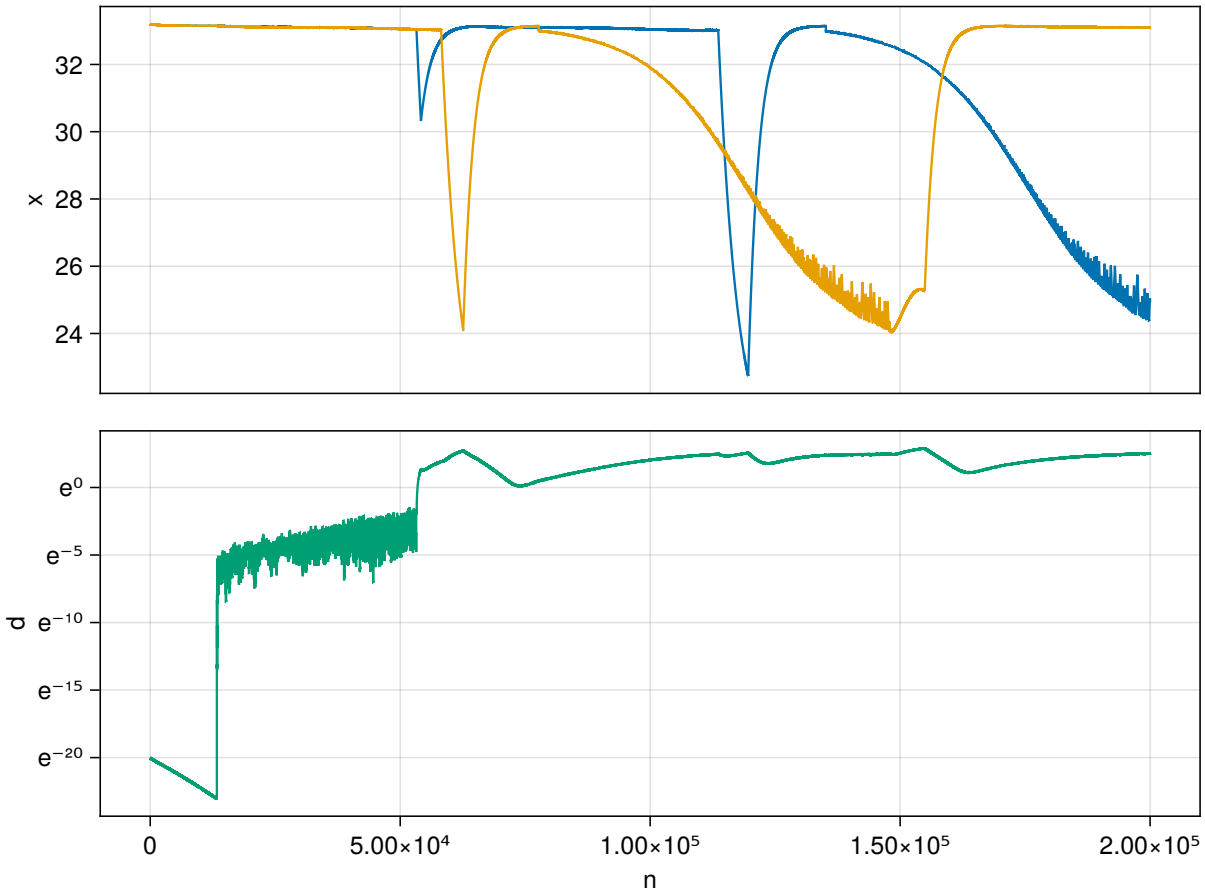


Figure 12: The top figure depicts the evolution of (the first component of) two trajectories with minimal perturbations of their initial conditions. The bottom figure represents the absolute distance between these two trajectories, in logarithmic scale. After an initial convergence, hitting the boundary leads to exponential divergence (the green line has constant slope upwards in the logarithmic scale). Finally the trajectories saturate, i.e. the boundedness of trajectories limits their absolute divergence.

sliding, unless Alice will already have left the switching surface. Either Alice or Bob will however exit from the 2-dimensional sliding plane. The coordination bias's cycles take place in 4 dimensions — they do not collapse on a pseudo-equilibrium.

Appendix C

In this Appendix we formalize the loose definitions given in Section 5 and we prove Theorem 5.

First, note that we can define an equivalence relation on the outcome space, $\sim_i$, such that $x \sim_i y$ if and only if $u^i(x, \lambda^i) = u^i(y, \lambda^i)$ for all $\lambda^i \in \Lambda_i$. Let $\mathcal{X}_i = \mathcal{X}/\sim_i$ be the quotient of the outcome space with respect to this equivalence relation. We will refer to an element of $\mathcal{X}_i$, an equivalence class $[x]_i$ as an i-outcome.

Let us define formally what it means to provide ex-post feedback.

Definition 9. A *feedback policy* for mechanism f and agent i is a factorization of f through a partition. It is composed of the following elements:

- A *signal space* S, which is a partition of Λ_{-i};
- A map $\phi_i \colon \Lambda_{-i} \to S$;
- A map $g \colon \Lambda_i \times S \to \mathcal{X}_i$;

such that

- The diagram commutes, i.e. $[f(\lambda^i, \lambda^{-i})]_i = g(\lambda^i, \phi_i(\lambda^{-i}))$ for all λ^i, λ^{-i};
- The map ϕ_i is a partition map, i.e. $\phi_i(\lambda^i) \ni \lambda^i$.

We denote a feedback policy by its map ϕ_i. A collection $(\phi_i)_i$ of feedback policies for each agent i is a *feedback structure*.

Remember the loose definition of Section 5: a feedback policy for agent i is a partition of the space of opponents' types, Λ_{-i}. This partition is such that agent i can evaluate what outcome he could have enforced had he unilaterally deviated to a different report λ^i. We request that a feedback policy allows agent i to compute all of his i-outcomes for any given report.

Next, we formalize the desire for reduced communication and revelation. We define the following partial order on feedback policies:

Definition 10. Feedback policy ϕ_i is *more private* than feedback policy ψ_i, denoted $\phi_i \trianglerighteq \psi_i$, if $\phi_i(\lambda^{-i}) \supseteq \psi_i(\lambda^{-i})$ for all λ^{-i}.

Any two type profiles that are indistinguishable under a less private rule should be indistinguishable under a more private rule. As mentioned, the privacy order is a weak partial order: not all feedback policies are comparable. However, it turns out that under the privacy order maximum and minimum are well-defined: the feedback policies together with the privacy order form a lattice.

Proposition 5. *Feedback policies together with the privacy order form a complete lattice.*

Proof. We simply need to show that for any two elements ϕ_i, ψ_i there exist a join $\phi_i \vee \psi_i$ and a meet $\phi_i \wedge \psi_i$ which satisfy the feedback policy definition. The argument follows from the lattice structure of the set of partitions with the partial order *coarser-than*. Note that $\phi_i^{-1}(\{x\colon x \in \Lambda_{-i}\})$ defines a partition of the space Λ_{-i}, and the same is true for the ψ_i. We then require the preimage of join $(\phi_i \vee \psi_i)$ to be the finest partition which is coarser than both the preimages of ϕ_i and ψ_i. Formally, let $A \subset \phi_i^{-1}(\{x\colon x \in \Lambda_{-i}\}) \vee_P \psi_i^{-1}(\{x\colon x \in \Lambda_{-i}\})$, then

$$(\phi_i \vee \psi_i)(\lambda^{-i}) = (\phi_i \vee \psi_i)(\hat{\lambda}^{-i}) \text{ for all } \lambda^{-i}, \hat{\lambda}^{-i} \in A$$

Similarly, define the meet as the function $\phi_i \wedge \psi_i$ such that it is constant over the meet of the two partitions. Again, let $B \subset \phi_i^{-1}(\{x\colon x \in \Lambda_{-i}\}) \wedge_P \psi_i^{-1}(\{x\colon x \in \Lambda_{-i}\})$, then

$$(\phi_i \wedge \psi_i)(\lambda^{-i}) = (\phi_i \wedge \psi_i)(\hat{\lambda}^{-i}) \text{ for all } \lambda^{-i}, \hat{\lambda}^{-i} \in B$$

The completeness of the lattice structure descends directly from the completeness of the partition lattice. □

Proposition 5 implies that there exist both a minimally- and a maximally-private feedback policy. The minimally-private policy is the full-revelation feedback policy: it reveals all information, and it is clearly less private than any other feedback policy. The maximally-private rule instead is a menu description.[17]

Definition 11. Let $[x]_i$ be the equivalence class of outcome x in $\mathcal{X}_i$. A *menu* for mechanism f and agent i, given reports λ^{-i} of the opponents, is the set

$$\mathcal{M}_{\lambda^{-i}} = \left\{ [f(\hat{\lambda}^i, \lambda^{-i})]_i \mid \hat{\lambda}^i \in \Lambda_i \right\}.$$

[17] Note that we defined the privacy lattice for feedback policies, not for feedback structures. When we analyze feedback structures, we implicitly consider the product lattice on the product space of feedback policies. This is correct because we assume private communication, but the analysis would likely change if we allowed public communication.

That is, the menu is the set of outcomes such that agent i could have received had he reported any type in his type space.

We can show that from the collection of all possible menus $\{\mathcal{M}_{\lambda^{-i}}\}_{\lambda^{-i}}$ we can construct a feedback policy, and it is the maximally private feedback policy for agent i.

Lemma 3. *There exists a feedback policy μ^i corresponding to the collection of menus $\{\mathcal{M}_{\lambda^{-i}}\}_{\lambda^{-i}}$, and μ^i is the maximally private feedback policy for agent i.*

Proof. Consider the space Λ_{-i} with the following equivalence relation:

$$\lambda^{-i} \sim \hat{\lambda}^{-i} \text{ iff } \mathcal{M}_{\lambda^{-i}} = \mathcal{M}_{\hat{\lambda}^{-i}}$$

and denote its quotient by $\Lambda_{-i}/\sim$. An element of the quotient is an equivalence class of opponents' types, denoted by $\left[\lambda^{-i}\right]$. Since $\sim$ is an equivalence relation, the quotient set $\Lambda_{-i}/\sim$ is a partition of Λ_{-i}. Let

$$\begin{array}{rccc} \mu^i\colon & \Lambda_{-i} & \to & \Lambda_{-i}/\sim \\ & \lambda^{-i} & \mapsto & \left[\lambda^{-i}\right] \end{array}$$

Let us show first that μ^i satisfies the definition of feedback policy. Of course, μ^i is a partition map: an element always belongs to its equivalence class. Define g as the function $g\left(\lambda^i, \left[\lambda^{-i}\right]\right) := [f(\lambda^i, \lambda^{-i})]_i$. Commutativity then follows from the fact that for all $\lambda^{-i} \in \left[\lambda^{-i}\right]$ the menus $\mathcal{M}_{\lambda^{-i}}$ coincide.

Now, we need to show that there is no feedback policy which is more private than μ^i. Equivalently, we can show that there is no feedback policy ϕ^i such that $\phi^i \triangleright \mu^i$. Suppose instead such a ϕ^i exists. Then, it must be that there exists a λ^{-i} such that $\phi^i(\lambda^{-i}) \supset \mu^i(\lambda^{-i})$. Then there exists a $\hat{\lambda}^{-i} \in \phi(\lambda^{-i})$ such that $\hat{\lambda}^{-i}$ is not in the same equivalence class of λ^{-i}. This implies that $\mathcal{M}_{\lambda^{-i}}$ is a different menu than $\mathcal{M}_{\hat{\lambda}^{-i}}$. Then, there exists a λ^i such that $[f(\lambda^i, \lambda^{-i})]_i \neq [f(\lambda^i, \hat{\lambda}^{-i})]_i$. We have then that $g(\lambda^i, \phi(\lambda^{-i})) = [f(\lambda^i, \lambda^{-i})]_i \neq [f(\lambda^i, \hat{\lambda}^{-i})]_i = g(\lambda^i, \phi(\lambda^{-i})$, a contradiction. □

We observe another interesting property of the privacy order that stems from its connection with the set of partitions of the power set of Λ_{-i}:

Corollary 3. *The communication complexity of a feedback policy is monotonically decreasing in the privacy order.*

The maximally private feedback policy has the lowest communication complexity and it maintains the highest level of privacy. Our characterization says that the most private

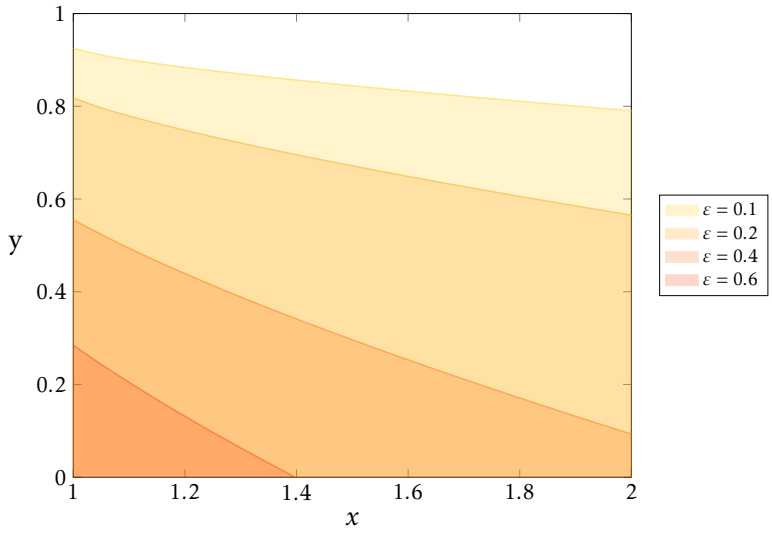
mechanisms are not particularly esoteric: they provide menu descriptions, which are well-known for their simplicity.

Appendix D

The contribution game of Section 3 can be seen as a particular one-dimensional parametrization of the Prisoner's Dilemma. In this Appendix we extend our analysis to general symmetric Prisoner's Dilemma games. We parametrize them as described in Figure 13a. We normalize the cooperation payoff to 1 and the sucker's payoff to 0, while we vary the payoff to deviation x and the payoff to mutual defection y.

		Bob	
		C	D
Alice	C	$1,1$	$0,x$
	D	$x,0$	y,y

(a) Payoffs of the stage game. $1 < x < 2$, $0 < y < 1$.



(b) Existence of cooperative pseudo-steady-state in the Prisoner's Dilemma parameter space.

Figure 13

We can replicate the analysis carried out for the contribution game in this more general setting, and we reach similar conclusions.

Proposition 6. *Consider a Prisoner's Dilemma with payoffs as in Figure 13a played by ε-greedy Q-learning algorithms. The forward limit set of* $\mathbf{Q}$ *is a singleton for any initial condition. Suppose the following inequalities are satisfied:*

$$\begin{cases} 1 < x < \frac{4+2\varepsilon-\varepsilon^2}{2\varepsilon-\varepsilon^2} - 4\sqrt{\frac{1}{2\varepsilon-\varepsilon^2}}, \\ 0 \leq y \leq -4\sqrt{\frac{(\varepsilon-2)^2\varepsilon^2[\varepsilon^2(x-2)-2\varepsilon(x-2)+4(x-1)]}{(4-2\varepsilon+\varepsilon^2)^4}} + \frac{16-4\varepsilon^3(x-1)+\varepsilon^4(x-1)-8\varepsilon(x+2)+4\varepsilon^2(1+2x)}{(4-2\varepsilon+\varepsilon^2)^2} \end{cases} \tag{12}$$

Then, there are two regions of attractions, R_C and R_D. Initial conditions in either region are attracted to two different steady states, q_C and q_D respectively. The steady-state q_D lies in $\omega_{D,D}$, while q_C is a pseudo-steady-state — it lies in $\overline{\omega}_{C,C} \cap \overline{\omega}_{D,D}$.

If Equation (12) is not satisfied, all initial conditions are attracted to the steady-state q_D.

The conditions for existence of a pseudo-steady-state appear complex, but the visualization in Figure 13b helps disentangling the various forces at play.

The higher the exploration rate, the more extreme the parameters x, y need to be to sustain the cooperative equilibrium. When both x and y are large the payoff from mutual defection is close to mutual cooperation, and a one-period defection provides large unilateral benefits. These are the cases providing the strongest incentives for defection, while when both x and y are low the opposite is true. The Figure reflects these intuitions for various levels of exploration.