JOB MARKET PAPER
Adaptive Algorithms and Collusion via Coupling*

Martino Banchio Giacomo Mantegazza†

Stanford Graduate School of Business

October 7, 2022

PRELIMINARY: 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 inadvertent coupling between the algorithms' estimates leads to periodic coordination on actions that are more profitable than static Nash equilibria. The coupling is generated by a set of parameters, the relative learning rates: collusion disappears when the learning rates are uniform. We apply our results to the design of *learning-robust truthful mechanisms*, which guarantee implementation robust to the presence of learning agents. Robustness relies on ex-post feedback provision, which sidesteps inadvertent coordination between learning agents by allowing counterfactual evaluations. We identify optimal learning-robust mechanisms, which communicate personalized menu prices, in a class of canonical design problems.

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

*We are thankful to Anirudha Balasubramanian, Kostas Bimpikis, Emilio Calvano, Peter DeMarzo, Andreas Haupt, David Kreps, Irene Lo, Alexander MacKay, Paul Milgrom, Ilan Morgenstern, Evan Munro, Michael Ostrovsky, Andrzej Skrzypacz, Stefan Wager, Larry Wein, Gabriel Weintraub, Kuang Xu and seminar participants at Stanford University, Harvard University, IIOC, and SITE Conference.

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

1 Introduction

Scholars and practitioners alike have expressed concerns that automated pricing and bidding software facilitates collusion. Simulations show how such software can learn sophisticated stick-and-carrot strategies (Calvano et al. (2020)), and empirical evidence shows prices in retail markets suffer a sharp increase after the introduction of automated pricing (Assad et al. (2021)). 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)).

Regulators and market design practitioners generally rely on theoretical models to guide policy interventions. The accepted wisdom in collusion analysis is that improvements in monitoring technology as well as in response times simplify collusive agreements, making the case for a high risk of algorithmic collusion. Yet, as noted by Kühn and Tadelis (2018), tacit collusion requires independent algorithms to solve a coordination problem. Spontaneous coordination does arise between algorithms enjoying substantial success in single-agent environments, but whose interactions in strategic environments remain puzzling. Although flourishing in the last few years, the theory surrounding markets with algorithmic agents suffers from several challenges.[1] In this paper, we argue that the existing theory developed for rational agents is not suitable for learning algorithms, which adapt and evolve over time. Regulators and mechanism designers face a new challenge: the takeaways from industrial organization and implementation theory are not robust to the presence of learning agents.

We contribute to the analysis of algorithmic collusion by developing a theoretical framework for the analysis of strategic interactions between algorithms. With fluid limit techniques, we approximate the evolution of algorithms driven by discontinuous laws of motion. In turn, this allows us to discover a vulnerability of traditional mechanisms. A form of "coupling" between independent adaptive agents undermines strategy-proof incentives. Non-uniform learning rates are responsible for introducing bias in the agent's estimates, sustaining long periods of correlated play. Symmetry makes profitable deviations hard to find, promoting collusive behavior.

We show that algorithms with uniform learning rates do not engage in collusive practices. We apply this result to construct a theory of learning-robust mechanism design. A designer may equalize learning rates by ensuring that agents can infer counterfactuals. In imperfect information settings the designer must reduce information asymmetry by providing feedback. Optimal learning-robust mechanisms select an outcome and release just

[1] See Tuyls and Weiss (2012) for a review of outstanding issues in the field of multi-agent learning.

enough information for every participant to compute counterfactuals. We design pVCG, a learning-robust mechanism that implements the social optimum. pVCG provides ex-post personalized menu pricing as feedback, and thus guarantees simple counterfactual evaluations with minimal information disclosure.

We focus on learning algorithms, which adapt rapidly to market conditions and have found ubiquitous market applications. In both retail pricing and online auctions, pricing and bidding algorithms need to quickly react to changes in demand and competition, often without direct human supervision. Because of their adaptive nature, learning algorithms are an ideal tool for decision-making in these contexts. We characterize a class of such adaptive algorithms, which we call *reinforcers*: they learn by inflating estimates of successful actions and deflating unsuccessful ones. Various popular automated decision software fall into this category, and their adaptive structure allows us to construct a continuous-time approximation. The piecewise-smooth system approximating the evolution of the algorithms is tractable enough for us to characterize its equilibria and regions of attraction.

We provide an example in which two AI algorithms (Q-learning) learn to cooperate in a Prisoner's Dilemma. The simulation shows that, even in a dominant-strategy-solvable game, algorithms may fail to reach equilibrium. We reconstruct analytically the results and prove that algorithms inadvertently coordinate on dominated actions. Intuitively, during a period of joint cooperation, each agent's estimates are correct if the opponent is also cooperating. Experimenting with defection appears profitable in the short run. However, the coupled agents soon begin to jointly defect, and the estimated value of defection falls. Instead, cooperation's estimate remains high, luring the algorithms away from defection. We dub this phenomenon the coordination bias, highlighting how nearly simultaneous deviations sustain dominated outcomes.

Inadvertent algorithmic coupling proves more general. The coordination bias arises when an agent learns about some actions faster than others. A single parameter for each action, the relative learning rate, determines the presence of coordination bias and inadvertent collusion. Actions with high relative speed adjust quickly to new information. Conversely, actions with slow relative speed take much longer to be re-assessed. Different persistence of estimates impairs the ability of a learning agent to adapt its behavior in response to strategic incentives.

The coordination bias disappears when agents observe the unrealized payoffs of neglected actions. Counterfactuals allow agents to update each action in lockstep, and persistence is uniform across all actions. While computing counterfactuals could itself be challenging, various environments make it impossible. For example, an agent's private

information may determine her opponent's price. In imperfect information settings, the evaluation of counterfactuals relies on information that may not be public. However, markets can develop disclosure policies that guarantee access to all unrealized payoffs. Market designers have control over the release of the private information gathered at any stage of the mechanism.

We formalize the notion of learning-robust truthful implementation: we require both adaptive and rational agents to report their private information truthfully. We show that implementation is achieved by jointly designing the allocation mechanism and a feedback structure. A generic strategy-proof implementation might fail without ex-post feedback to help algorithms learn — learning-robust mechanisms provide such feedback. We can augment any mechanism with a feedback structure, making it learning-robust. These mechanisms are ordered according to the degree of information they reveal to their participants. We look for the optimal learning-robust mechanism, which discloses the least private information to all participants. We characterize the maximally-private mechanism for efficient implementation with quasi-linear utilities. Optimal feedback takes the form of personalized menu pricing. Each choice from the menu corresponds to a price for a given outcome, simplifying counterfactual evaluations.

Finally, we show that experimental results found in the literature can be derived analytically within our framework. We apply our model to a recent paper by Asker et al. (2022). In a simulated Bertrand oligopoly, the authors find that different learning rules can lead to either competition or collusion. By exploiting our results, we can show that their algorithms reach competition only when they have access to counterfactual profits. We then repeat the same analysis in the context of auctions studied in Banchio and Skrzypacz (2022).

The paper is structured as follows: in the next Section we review the related literature; in Section 2 we introduce the theoretical framework and derive the approximation results. In Section 3 we present a complete analysis of a Prisoner's Dilemma and we build intuitions that we generalize in Section 4 to various settings. In Section 5 we employ these results to establish a theory of learning-robust mechanisms, and we finally apply our results in Section 6.2 and 6.3 to the Bertrand pricing of Asker et al. (2022) and the auctions of 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 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 these earlier results under a general structure: Examples 1 and 2 show how our results apply to Fictitous Play

and a well-known implementation of regret minimization.

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.[2]

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 evo-

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

lutionary properties. We suspect similar results would hold under our framework. All of these works provide robustness to certain types of bounded rationality and learning procedures. As we show, the robustness guaranteed by the game form may be insufficient — even the strongest 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 which we term adaptive. Because their dynamics tend to be complex, we approximate these algorithms in continuous-time with ordinary differential equations (ODEs). We then formally define the learning agents we focus on, which we call *reinforcers*: they learn by boosting successful actions and sinking unsuccessful ones.

2.1 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 adaptive agents playing the game G repeatedly, over an infinite horizon. For simplicity, let us look at one adaptive agent, Alice, choosing her actions from the finite set A_i with cardinality d_i. Her payoff is captured by the function $r_i(a_i, a_{-i})$, which depends on her opponent's actions as well as her own. In this environment, Alice learns how to play optimally.[3] We assume that the adaptive agents do not have access to a monitoring technology, and in particular they cannot explicitly condition future play on past play. That is, the agents do not learn optimal strategies of a repeated game, but simply optimal actions.[4] Within this environment, we define a class of learning algorithms that includes the most common automated decision-making procedures.

Definition 1. An *adaptive agent* in game G is a pair (θ, π) consisting of

- An adaptive algorithm θ, that is a d_i-dimensional stochastic process which evolves according to

$$\theta_k = \theta_{k-1} + \alpha T^{\pi}(\theta_{k-1}, Y_k, k),$$

[3] The environment resembles stochastic multi-armed bandit settings, with the difference that the actions by other players may affect the "distribution" of the rewards offered to our agent.

[4] This is in contrast with Calvano et al. (2020), who instead allow algorithms to condition their behavior on past play of the opponents. While a reasonable assumption, we should only expect collusion to be more likely when providing additional monitoring technology — we will find that behavior akin to tacit collusion takes place even without monitoring.

where $\theta \in K \subset \mathbb{R}^{d_i}, \alpha \in \mathbb{R}$, $(Y_k)_k$ are i.i.d. random variables in $\mathbb{R}^e$ for some $e \in \mathbb{N}$, and T^π is a function,

- A *policy* π, that is a map $\pi\colon K \times \mathbb{R}^e \to A_i$, which selects an action for each value $\theta \in K$ of the algorithm and for each realization y of the random variable $Y \in \mathbb{R}^e$.

The adaptive agent carries a statistic for each available action, and in each step k selects an action $a \in A$ according to its policy π. Notice how the definition of policy incorporates an element of randomness: adaptive agents may randomize their actions. This implies that, from the perspective of Alice, rewards in any given stage will appear stochastic: Alice's opponents may randomly select actions independently of her choices. This randomness is captured by the random variables Y_k, with distribution $\nu(Y)$ over $\mathbb{R}^e$. The function T^π, commonly referred to as *update*, depends on the policy. We will drop the superscript when it is obvious which policy we are referring to. We require T^π to be extendable over the positive real line $[0, +\infty)$ in its last component.

We maintain the following assumption on the update function T^π:

Assumption A1. The function $T^\pi(\theta, Y, k)$ is Lipschitz-continuous almost-everywhere in θ and k.

Definition 1 is rather general:[5] most of temporal-difference-based algorithms belong to this class, with popular choices such as Q-learning, SARSA and Actor Critic methods. Additionally, many popular procedures such as Fictitious Play (Brown (1951)) and the reinforcement learning model of Erev and Roth (1998) can be formulated as agents using adaptive algorithms.

Example 1. *Fictitious Play is a learning procedure that requires agents to best respond to the empirical distribution of the opponent's strategies. Suppose there are two players, A and B, and let $\beta(\cdot)$ be the multi-valued best-response function. In our language, the empirical distribution is the relevant stocastic process, and its evolution for player A is guided by the following:*

$$\theta_k^A = \theta_{k-1}^A + \frac{\beta(\theta_{k-1}^B) - \theta_{k-1}^A}{k}.$$

Since the space K is $\Delta(A)$, the policy of Fictitious Play maps an empirical distribution into the uniform distribution over its best-response set. That is, Fictitious Play draws an action at random from the best-response correspondence of the current empirical distribution.

[5] θ_t represents any learned statistic that the decision maker conditions his choices upon.

Fictitious Play has been well studied and thoroughly understood. However, it is seldom implemented in practice, mostly because of its model-dependent policy, which requires knowledge of the strategic structure of the game. The algorithm designer must know in advance the game in order to be able to compute the best response. Similarly, regret minimization algorithms require the agent to know the ex-post optimal payoff.

Example 2. *A simple adaptive implementation of regret minimization is the Polynomial Weights algorithm proposed by Cesa-Bianchi et al. (2005). The algorithm maintains weights w_k^i for each action i and updates the action chosen according to*

$$w_k^i = w_{k-1}^i - \lambda l_k^i w_{k-1}^i,$$

where $l_k^i = r_k^* - r_k^i$ is the regret from obtaining payoff r^i instead of the optimal payoff r^* in iteration k and λ is a normalization parameter. The weights are then aggregated in a probability of choosing action i in each period with $\theta_t^i = \frac{w_t^i}{\sum_j w_t^j}$. Some algebra shows that one can write the update of the probabilities as

$$\theta_t^i = \theta_{t-1}^i + \left(\frac{\theta_{t-1}^i (1 - \lambda l_t^i)}{\sum_j \theta_{t-1}^j (1 - \lambda l_t^j)} - \theta_{t-1}^i \right)$$

The policy corresponding to the polynomial weights algorithm is simply the identity map: the agent draws an action from the distribution given by the current state of the adaptive algorithm.

Klos et al. (2010) show that such procedure shares the structure of the Replicator Dynamics from the evolutionary game theory literature.

A common feature of adaptive algorithms is that while rather simple to analyze in a stationary environment, they often become unpredictable when learning to play against each other. Specific algorithms carry guarantees in some classes of games (e.g. fictitious play and potential games, regret minimization and zero-sum games, etc.) but, when such guarantees are missing, predicting outcomes proves a difficult task. The random component Y as well as the discreteness of the jumps in the process make equilibrium analysis intractable even for the simplest strategic environments. In order to overcome these difficulties, we turn our attention to a continuous-time approximation, which we construct in the next section.

2.2 Approximation in Continuous Time

Our goal is to obtain mechanisms robust to the dynamics of their learning participants. Therefore, we are interested in describing the dynamics of learning of a given adaptive algorithm. This goal presents three main difficulties:

1. Stochastic updates make it hard to predict the path of the stochastic process θ;
2. Even if uncertainty was resolved, the statistics of the learning agents evolve according to difference equations, which in general are more complex analytically than their continuous counterpart;
3. The interactions between independent learning agents further increases the dimensionality of the problem.

In this section we take care of items 1 and 2 through the use of a continuous-time approximation. We adopt the formalism of fluid approximations first introduced by Kurtz (1970). The limiting fluid ODE system provides tractability and (often) solutions in closed form. This technique allows us to bypass the complexity induced by discreteness and randomness present in general adaptive algorithms.[6]

The cornerstone of our continuous-time framework is the following theorem, which guarantees that for any adaptive procedure of the type of Definition 1 there exists a continuous, deterministic process, that approximates the original system.

Theorem 1. *Let (θ, π) be an adaptive agent that satisfies Assumption A1, and let the domain $K \subset \mathbb{R}^d$ 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*

$$\begin{cases} \frac{d\Theta(t)}{dt} = \alpha \int_{\mathbb{R}^e} T(\Theta(t), Y, t) d\nu(Y) \\ \Theta(0) = y_0 \end{cases}$$

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

- $\mathbb{E}\left[\theta^1(t)\right] = \mathbb{E}\left[\theta_t\right]$ *for all t,*
- *the infinitesimal generators $\mathcal{A}T^n(\theta, t)$ are all identical to $\mathcal{A}T(\theta, t)$ for all (θ, t) and n,*

[6] Refer to Section 3 for a fully worked-out example of these difficulties.

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

The remainder of this section is devoted to first interpreting this result, and then providing a short sketch of the proof. A formal proof can be found in Appendix A.

The process $\Theta(t)$ for $t\geq 0$ is the fluid approximation to θ_k: it is a deterministic process whose time-derivative is the expected update that the discrete process θ_k would incur over one unit of time. Theorem 1 states that it is possible to construct a sequence $\{\theta^n\}$ of processes that draw closer and closer (in probability) to the continuous process. The infinitesimal evolution of θ^n is identical across all n. The theorem relies on a law-of-large-numbers argument: if the updates occur with high frequency, but each update's contribution is relatively small, the process behaves similarly to its expectation. The proof is constructive, in the sense that we first define this sequence of processes, and then show that we can apply the fundamental theorem of fluid approximations by Kurtz (1970) to conclude convergence in probability.

Outline of the proof. The proof proceeds in three steps. The first step is to turn the discrete-time process into a continuous-time one. This transformation is carried out by means of a *Poissonization* argument: given a Poisson clock with unit rate, we assume that the updates of θ occur at each tick of the clock. Mathematically it is equivalent to constructing a compound Poisson process with unit rate. Notice in particular that this construction makes sure that, at time $t_k\in\mathbb{R}_+$ such that the clock ticked k times, the continuous-time process thus obtained is equal to θ_k.

The second step consists in finding an appropriate rescaling of time and of the updates; in particular, we accelerate time and downsize the jump of every update. Formally, we consider an increasing sequence of rates λ_n of the Poisson clock while dividing the learning parameter α by n. When accelerating time, the learning statistic θ is updated more often and more observations of the random quantities occur in the same unit of time; thus, downsizing the extent of the jump is necessary to avoid divergence of the process. These two effects combine in such a way that we can apply a law of large numbers, so that one can approximate a system with very frequent updates with one whose derivative is the expected update.

2.3 A Class of Adaptive Agents

We now state formally the class of adaptive agents we consider in the rest of the paper. We define the class of *reinforcers*: agents who learn by reinforcing the successful actions

and penalizing the unsuccessful ones. Since we will work with the fluid approximation in what follows, we state the reinforcer definition in continuous-time.

Definition 2. A *reinforcer* is a pair (θ, π) consisting of

- An adaptive algorithm θ on a compact domain $K \subset \mathbb{R}^{d_i}$ such that, across each maximal continuity domain H_j the update for each action $a_i \in A_i$ when the opponents' actions are $a_{-i} \in A_{-i}$ is of the form

$$\frac{d}{dt}\theta^{a_i}(t) = \alpha^{a_i}\Big(\theta(t)\Big)\Big[U\Big(\theta^{a_i}(t), r(a_i, a_{-i})\Big) + V\Big(\theta(t)\Big)\Big]. \tag{1}$$

  where $\alpha^{a_i}(\theta) \in [0,1]$. Moreover, we assume that α, U and V are Lipschitz in all components, U is Lipschitz everywhere on K and increasing in $r(a_i, a_{-i})$, decreasing in θ^{a_i}, and $\frac{\partial V}{\partial \theta^{a_i}} < -\frac{\partial U}{\partial \theta^{a_i}}$ almost everywhere.

- A policy $\pi\colon K \to \Delta(A_i)$, which specifies a distribution of actions for each instantaneous value of the algorithm θ.

Two parts make up a reinforcer: a component which is equal for all actions ($V(\theta)$), and a component which is action specific but Lipshitz over the entire domain K ($U(\theta^{a_i}, r)$). The function U remains identical for all actions, but it takes a given action's estimates as values. The monotonicity assumptions amount to requesting that a reinforcer's updates are larger after good news, but the larger the value of θ the smaller the update is. As mentioned earlier, the definition of reinforcer is given in continuous-time: an adaptive agent is a reinforcer if its fluid approximation satisfies Definition 2.

To analyze the ODE system of reinforcers, we will need some dynamical systems tools.

Definition 3. Given a dynamical system $\frac{d\theta}{dt}$, the *flow* of θ starting in x is the map

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

that satisfies Equation (1) with initial condition $\theta_x(0) = x$. A *trajectory* of the dynamical system is the graph of the flow, $\Big\{(t, \theta_x(t))\colon t \in [0,+\infty)\Big\}$. The set $\Gamma_x = \Big\{\theta_x(t)\colon t \in [0,+\infty)\Big\}$ is the *orbit* of θ starting from x. A *forward limit set* for the orbit Γ_x is the set of points y such that there exists a a sequence $(t_n)_{n\in\mathbb{N}}$ such that

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

A *steady state* of the dynamical system is a fixed point of its law of motion, i.e. $\frac{d\theta}{dt}(t) = 0$.

We will use these tools in describing the behavior of reinforcers in the limit. 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.

We can now show how Q-learning, the building block of many Artificial Intelligence algorithms, belongs to the class of adaptive algorithms and in particular reinforcers. The popularity of Q-learning-based methods combined with its interpretability lead us to adopt it as a running example in what follows.

Example 3. *Q-learning approximates the optimal action-value function in a repeated game. It maintains estimates $Q_k(a)$ of the value of an action a, and in each iteration the Q-vector $(Q_k(a))_a$ gets updated as*

$$Q_k(a) = \begin{cases} Q_{k-1}(a) + \alpha\left[r_{k-1} + \gamma \max_{a'} Q_{k-1}(a') - Q_{k-1}(a)\right] & \text{if } a = a_{k-1} \\ Q_{k-1}(a) & \text{else.} \end{cases} \tag{2}$$

where r_{k-1} is the payoff obtained in iteration k.

Q-learning is an adaptive algorithm, whose estimates of each action evolve according to a Lipschitz function of the reward and the estimate of the continuation value.Intuitively, the Q-vector estimates continuation values according to a Bellman equation. In iteration k, the value of an action at a certain state is a convex combination of its previous estimate (with weight $1-\alpha$) and of a new Bellman estimate (with weight α). The weight α serves as rate of learning, and can be thought of as a measure of persistence: higher αs make the algorithm faster in forgetting the past.

We will focus our attention in most of this work on a particular policy, known as ε-*greedy*:

$$(\varepsilon\text{-greedy})(\theta)) = \begin{cases} p(a) = \dfrac{1}{|\operatorname{argmax}_{a\in A_i}\theta^a|}, & \forall a \in \operatorname{argmax}_{a\in A_i}\theta^a \quad \text{with probability } 1-\varepsilon \\ p(a) = \frac{1}{d_i}, & \forall a \in A_i \setminus \operatorname{argmax}_{a\in A_i}\theta^a \quad \text{with probability } \varepsilon \end{cases}$$

The inverse image of the ε-greedy policy separates the space K in subsets identified by the action with the largest estimated continuation value. We will exploit this structure in constructing continuous pasting in Section 3. Observe that ε-greedy Q-learning is a reinforcer: it is increasing in the reward obtained in each iteration, and since $\gamma < 1$ it is always decreasing in an action's own estimates.

Note that a policy such as ε-greedy naturally incorporates "mistakes": 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 small probability). Learning agents can nonetheless become stationary, if their estimates reach a steady state. We adopt this view for simplicity and clarity of exposition.

Definition 4. We say a reinforcer (θ, π) *converges* if $\theta(t)$'s flow converges on a steady-state $\overline{\Theta}$. We say the agent *converges on action* a_{ss} if the steady-state $\overline{\theta}$ is such that $a_{\text{ss}} \in \text{argmax}_{a \in A} \overline{\theta}^a$. If the argmax is not unique, we say $\overline{\theta}$ is a *pseudo-steady-state*.

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

Definition 5. We say the agent *learns action* a_l if there exists a $T > 0$ such that $a_l = \text{argmax}_{a \in A} \theta^a(t)$ for all $t \geq T$.

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 then, an agent can learn action a_l even if such action is not played in each period.

Finally, when multiple agents employ adaptive algorithms, the system can be represented as a single algorithm that belongs to the class of Definition 1 by stacking the stochastic processes together, for example as $\theta_k = \left(\theta_k^1, \theta_k^2, \ldots, \theta_k^N\right)$. Throughout the next sections, θ_k will denote this collection of learning statistics for all states and all players unless otherwise specified.

3 An Illustrative Example

We now analyze the behavior of adaptive agents in a simple environment with our continuous-time approach. We focus on a game with an equilibrium in dominant strategy, and we how even dominant-strategy fails to guarantee equilibrium convergence. This seems at odds with our perception of dominant strategies as the strongest incentives we can provide to rational agents. Of course, reward-punishment schemes could sustain collusion when strategies can be conditioned on past play, but the adaptive agents we consider are unable to sustain explicit tit-for-tat strategies. Nonetheless, we show how algorithms sustain cooperation in a contribution game, despite cooperation being strictly dominated. We explain these findings by analytically identifying a bias that arises from coordination:

the algorithms cooperate and defect symmetrically, therefore they overestimate the value of cooperation.

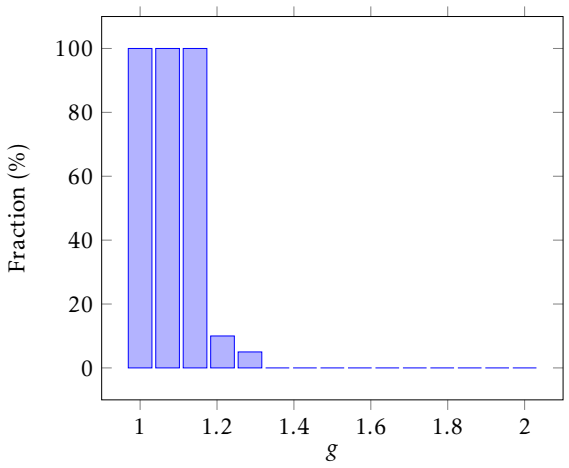
Consider the following family of contribution games, with payoffs given by Figure 1. Alice and Bob have two American dollars each, and they simultaneously decide whether or not to contribute to a common pool which grows its value by a factor of $1 < g < 2$. The riches in the pool are then shared equally between Alice and Bob, but additionally each agent gets to keep what they didn't contribute to the pool. This is a canonical model of the free-rider problem, parameterized by g which 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 always play "defect" and keep the change.

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

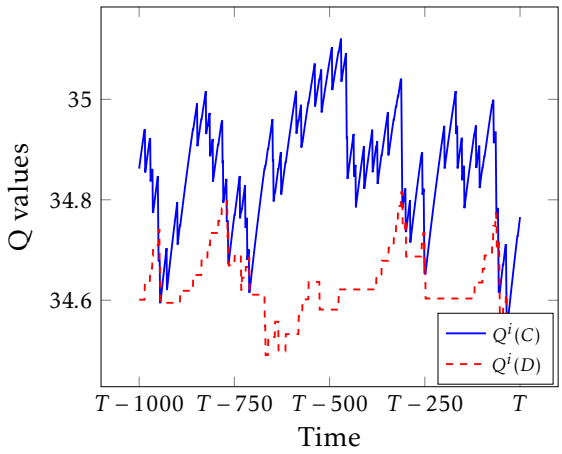
Figure 1: Payoffs of the stage game, $1 < g < 2$.

Simulations. We simulate the learning dynamics of Alice and Bob, when they adopt reinforcers in the above free-rider problem. We choose to analyze Q-learning, which maintains independent estimates of the value of taking each action, $Q_k(C)$ and $Q_k(D)$. In this static game the algorithms compress all past play in their estimates of the values Q_k. We assume that both Alice and Bob choose their actions according to a ε-greedy policy: recall from Section 2.2 that at each iteration k Alice and Bob play the action with the highest estimated value, i.e., $\text{argmax}_{a \in \{C,D\}} Q_k(a)$, with probability $1 - \varepsilon$, and with probability ε they choose an action uniformly at random.

Figure 2a shows the results of these numerical experiments. The algorithmic agents learn the dominant strategy equilibrium $\{D, D\}$ only for low values of the parameter g. Instead, for high values of g the agents cooperate, albeit imperfectly. In fact, both cooperation and defection appear in "quasi-recurrent cycles", as shown in Figure 2b: the value of collaboration is generally above that of defection, but at somewhat regular intervals it drops below $Q_k(D)$, so that agents switch to playing D. The value of defection then decreases almost immediately, and players revert to cooperation. This analysis highlights a few puzzles: (i) the parameter g should have no effect on strategic decisions, and instead



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



(b) Cycles in the discrete system, obtained with $g = 1.8$.

Figure 2: We initialized 100 independent runs of Q-learning with $T = 100,000$ time steps each, and we consider a profile learned if it is played over 50% of the last 1000 iterations.. In all cases, $\varepsilon = 0.1$, $\alpha = 0.05$ and $\gamma = 0.9$. The initialization is optimistic, i.e. all Q-values are larger than the maximum value they could ever achieve. This leads to a phase of intense exploration at the outset.

it leads to stark differences in outcomes; (ii) collusion seems to consist of cycles, but since agents cannot condition on the past action of the opponent, it is hard to impute these to "retaliatory" strategies. The continuous-time approximation will help us to transparently identify the causes of these patterns.

3.1 Theoretical results

In this section we present an in-depth analysis of the contribution game from a theoretical perspective. We will show how to recover the experimental results, and how to interpret them.

A fundamental step in understanding these algorithms is thinking about their long-run behavior. The actions taken according to an ε-greedy policy will never be constant in the long run, and therefore there will always be some variation in the discrete-time algorithms. Similarly, the adaptive algorithm never converges: its statistics oscillate forever due to the constant learning rate α. When analyzing continuous-time approximations, however, these complications disappear: the limiting behavior of simple ODE flows is much easier to classify.

We begin the analysis of the contribution game by writing down explicitly its continuous-

time approximation. Notice that when Alice and Bob adopt ε-greedy Q-learning 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 have two different functions T: one for $Q_k(D) \geq Q_k(C)$, and one for the opposite case. To sidestep this discontinuity in the right-hand side of the discrete-time system, we first apply Theorem 1 to Q_k over T's maximal continuity domains. Recall from Section 2.2 that the maximal continuity domains are the sets where the policy is constant:

$$\omega_{a,b} = (\varepsilon\text{-greedy})^{-1}\left(\left\{Q \mid (Q^A(a) > Q^A(a')\ \forall a') \wedge (Q^B(b) > Q^B(b')\ \forall b')\right\}\right)$$

That is, $\omega_{a,b}$ is the set of values of 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[7] $1 - \frac{\varepsilon}{2}$ 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 solves

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

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

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

This 4-dimensional piecewise-linear system's behavior turns out to be complex. In particular, the system exhibits chaotic behavior over a range of parametrizations and initial conditions (see Appendix B for more details on chaos theory and its analysis in the contribution game).

In order to make progress on the analysis of the chaotic system, we begin by restricting

[7] 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 was instructed to play an action at random.

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 $\{Q \in \mathbb{R}^4 | Q^A(a) = Q^B(a) \text{ for } a = C, D\} \cong \mathbb{R}^2$ is a subspace for the dynamical system in Equation (4). In what follows, we will then make the following assumption:

Assumption S. Let $\mathbf{Q}_0^A(a) = \mathbf{Q}_0^B(a)$ 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. *There always exists a steady-state $q_D^{eq} \in \omega_{D,D}$. Moreover, if $\varepsilon < 1 - \sqrt{\frac{2-g}{g}}$ there exists a pseudo-steady-state $q_C^{eq} \in \overline{\omega}_{D,D} \cap \overline{\omega}_{C,C}$, given by*

$$q_C^{eq}(C) = q_C^{eq}(D) = \frac{1 + g + \sqrt{(g-1)(g-1-\varepsilon g + \frac{\varepsilon^2 g}{2})}}{(1-\gamma)}.$$

In the symmetric subspace, the limiting behavior of the system can be twofold. When $\varepsilon > 1 - \sqrt{\frac{2-g}{g}}$, the algorithms converge on a steady state q_D^{eq} where both players find defection to be the preferred action. However, when the condition is reversed, there exist an additional steady-state. Recall Definition 4: this pseudo-steady-state q_C^{eq} is located at the intersection of the sets where Alice and Bob prefer cooperation and defection. In this steady-state, Alice and Bob play cooperation only a fraction τ_C of the time, and defect for a fraction $1 - \tau_C$ of the time.

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

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

The 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 closely approximates its discrete-time experimental counterpart, as shown in Figure 6b. Moreover, the notice that the general asymmetric 4-D system gravitates around a similar equilibrium, where $Q(C) = Q(D)$.

We will spend the next subsection constructing the necessary tools to formally prove Proposition 1 and its corollary.

3.2 Sketch of the proof

Recall that under Assumption S the system can evolve according to one of the following two equations:

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

The two systems allow us, by describing their flows, to characterize the evolution of the Q-functions within the continuity domains. The system at large however is a vector field with a discontinuous right-hand side: the motion of the flow changes discontinuously 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 field defined by the fluid limit possesses a global solution.

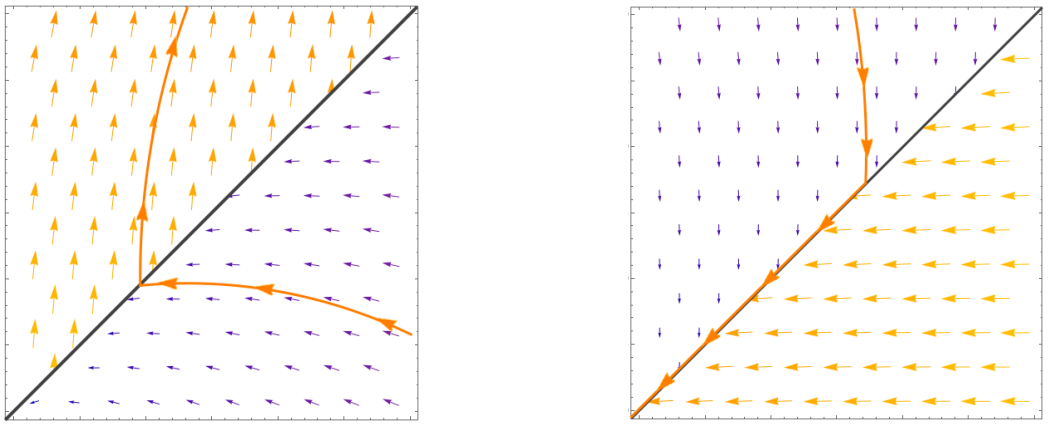
Proposition 2. *Let F_a be the field defined as above over $\omega_{a,a}$ for all $a \in \{C, D\}$. There exists a global solution in the sense of Filippov (1988) to the differential inclusion*

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

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

The two systems of ODEs are well-defined also on the boundary between $\omega_{C,C}$ and $\omega_{D,D}$. This boundary is called switching surface, as the laws of motion switch from one field to the other. Adopting the Filippov convention, we can define a vector field on the switching surface such that the flows extend to a global solution. The switching surface can be divided into three regions. A *crossing region* occurs where the components of the two vector fields F_j normal to the boundary are of the same sign. On the crossing region there is no need to define a field on the boundary: flows continue seamlessly without additional requirements. A *repulsive region* occurs where both normal components face away from the boundary, which will then never be reached. Again, since this boundary will never be reached we do not define a vector field there. Finally a *sliding region* occurs when both normal components point towards the boundary. The only possible path for a 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 $F_{C,C}$ and $F_{D,D}$ such that the normal component to the switching surface vanishes. This is the unique vector field in the convex hull whose flow

is confined to the switching surface and which satisfies the differential inclusion requirements. Figure 3 plots the flow of the system around the boundary and shows an example of sliding and crossing boundaries, together with a sample trajectory. For more details



(a) Trajectory crossing the boundary.

(b) Trajectory sliding along the boundary.

Figure 3: Depiction of two discontinuous flows around a switching surface, in the crossing and sliding case.

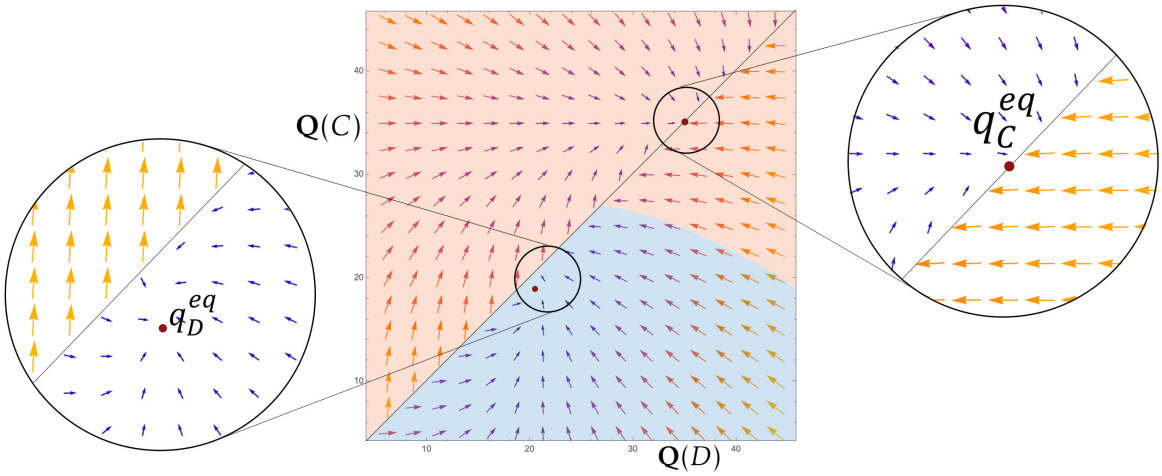
on how the field is calculated we refer the reader to the proof of Proposition 1 in the appendix.

mbanc

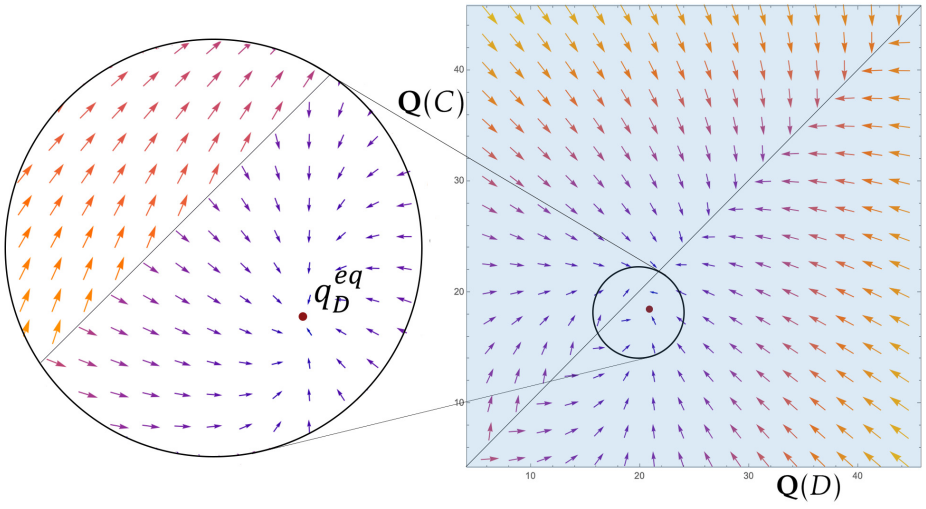
Stability analysis Figure 4 shows the vector fields that characterize the continuous-time approximation: its stationary points correspond to the steady states of the original system. The figures highlight the presence of two stationary points when g is large. The pseudo-steady-state on the boundary disappears for lower 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 equilibrium*. Its existence hinges on the relationship between the value of cooperation and the exploration rate. Figure 5 shows the minimum exploration $\underline{\varepsilon}$ that guarantees a cooperative equilibrium and how it varies 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.[8]

[8]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).



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



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

Figure 4: Stationary points are marked with a red dot. The domain of attraction of the cooperative outcome is green-shaded, the one for the non-cooperative outcome is blue-shaded.

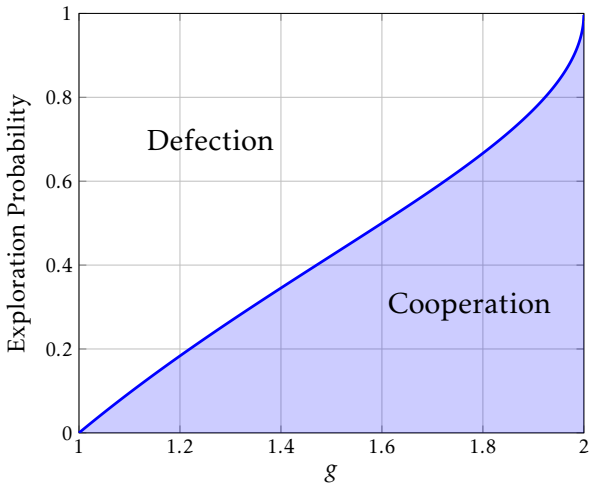
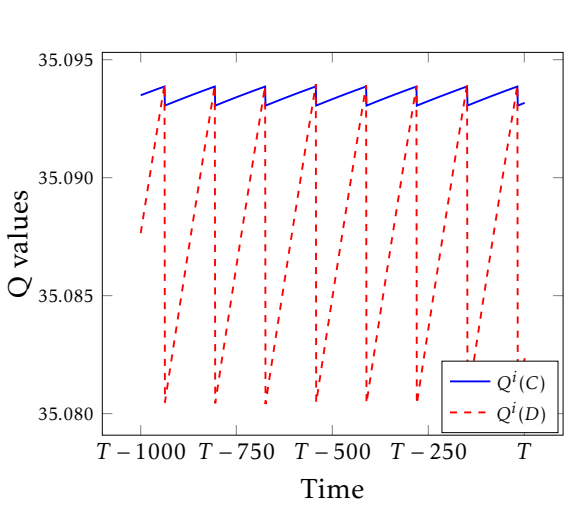


Figure 5: Maximum exploration rate $\underline{\varepsilon}$ to support the cooperative equilibrium, as a function of the growth rate. For all $\varepsilon > \underline{\varepsilon}(g)$ there does not exist a steady-state where both algorithms learn to cooperate.

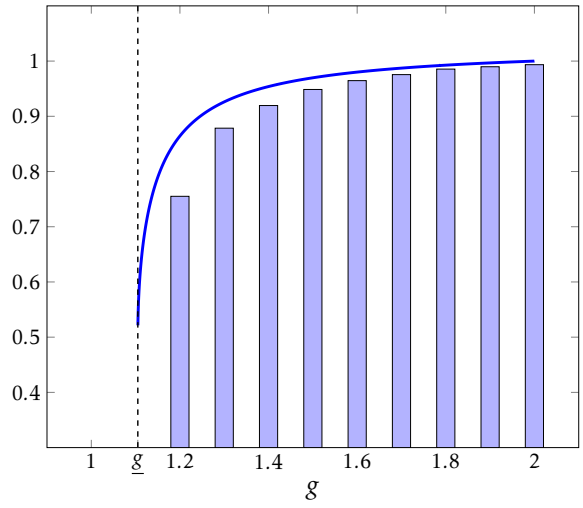
Sustaining cooperation Note that the system of ODE, when simulated with small but discrete time steps, closely mimics the path of play of the discrete Q-learning. This behavior is clearly observed in Figure 6a and can be compared with Figure 2b, shedding some light on how cooperation can be sustained. 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, $Q(D)$ is bound to rise above $Q(C)$. Suppose this happens first to Alice. Once Alice defects, Bob will also defect almost immediately after, because cooperating when Alice defects makes Bob considerably worse off. Joint defection decreases the value of $Q(D)$ for Alice and Bob and reinforces the value of joint cooperation. Effectively, when the exploration rate is too small, the adaptive agents play a symmetric profile of actions too often. We call this effect the *coordination bias*: the estimated value of action a is correct *conditional on the opponent playing (almost always) the same action*. When Alice begins defecting, her experiments with cooperation are too infrequent, and her estimate of the value of cooperation remains biased. Effectively, Alice is too slow to realize the downside from cooperating when Bob defects, and the benefit of defecting when Bob cooperates.

The steady state on the boundary is the continuous-time counterpart of these cooperative cycles in discrete system. The discrete cycles tend to a single steady state in the continuous-time limit. Agents spend some "local time" playing cooperation: we interpret the weights assigned to the fields on the boundary as the fraction of time dedicated to cooperation and defection, respectively.[9] The local time is such that the infinitesimal

[9]This intuition can be made formal using the idea of hysteresis loop around the boundary; see



(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 6

incentives around the stationary point are balanced.

Figure 6b plots this quantity and shows that the proportion of time spent playing the cooperative profile at equilibrium varies with g. We notice that the local time spent cooperating obtained analytically approximates well the estimates from the simulations. Moreover, τ_C is always greater than $\frac{1}{2}$, which squares well with the results from Figure 2a: if the cooperative equilibrium exists, Alice and Bob collude more than 50% of the time, which was the threshold we chose when classifying the simulations' outcomes.

4 Main Results

The mechanics of coordination bias, identified in Section 3, appear tied to the policy of the reinforcer. However, we show that the driving force of inadvertent coupling lies in the algorithm's rate of learning across different actions. First, we show how, in general, learning at differential rates across actions leads to coordination bias. We then establish uniform learning rates as a sufficient condition for convergence.

di Bernardo et al. (2008)

4.1 A Negative Result

The ε-greedy Q-learning analysis in the Prisoner's Dilemma of Section 3 gives us a sense of the mechanics behind collusion by 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. Insufficient exploration impairs the ability of the algorithm to wash away existing bias — discovering the value of the dominant strategy becomes difficult. This phenomenon can be formalized without referring to a specific policy. Observe that the dependence of updates on the policy is completely contained in the learning rates, α. For example, in the case of ε-greedy Q-learning, recall the differential equations within the maximal continuity domain $\omega_{C,C}$:

$$\begin{cases} \dfrac{d\mathbf{Q}_t^A}{dt}(C) = \quad \boldsymbol{\alpha}\dfrac{\boldsymbol{\varepsilon}}{\mathbf{2}} \quad \left[\left(1-\dfrac{\varepsilon}{2}\right)g + \dfrac{\varepsilon}{2}2g + (\gamma-1)\mathbf{Q}_t^A(C)\right] \\ \dfrac{d\mathbf{Q}_t^A}{dt}(D) = \boldsymbol{\alpha}\left(\mathbf{1}-\dfrac{\boldsymbol{\varepsilon}}{\mathbf{2}}\right)\left[\left(1-\dfrac{\varepsilon}{2}\right)2 + \dfrac{\varepsilon}{2}(2+g) + \gamma\mathbf{Q}_t^A(C) - \mathbf{Q}_t^i(D)\right] \end{cases}$$

In the language of reinforcers, the reinforcer's own policy affects the learning rates alone. Each statistic is already estimated conditional on what action has been taken, and the 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 the coordination bias. The coordination bias appears in a Prisoner's Dilemma for any pair of agents using any 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 either of them.

Theorem 2. *Let each agent learn through a greedy reinforcer in a Prisoner's Dilemma. Then, there exist an open set of parameters $x, y, \{\alpha^j(\omega_{k,k})\}_{j,k=C,D}$ for which there exists a pseudo-steady-state exhibiting coordination bias.*

We prove Theorem 2 formally in the Appendix, but we provide a brief sketch of the proof here. The main idea is to construct a pseudo-steady-state of the reinforcer by mim-

icking the construction of the pseudo-steady-state we found in Section 3.1. That is, we want parameters such that the vector fields on the two sides of the boundary between maximal continuity domains are in equilibrium. To do so, we first deal with the case where the local time on the switching surface is equal on either side of the surface. We then show that it is possible to construct a loop in $\mathbb{R}^2$ by varying the parameters α^C, α^D and the value θ^* on the boundary. We generate a homotopy between the loop and the origin by varying these parameters, thus showing that there exists a pseudo-steady-state. Adjusting the local time leads to continuous roto-translations of the homotopy, thus there must exist an open interval of α^C, α^D which generates a pseudo-equilibrium on the switching boundary.

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 Uniform Learning Understands Incentives

Unlike the Prisoner's Dilemma, in many situations dominant strategies also represent the desirable outcome, from the perspective of a market designer or an implementor. For example, an auctioneer who adopts a second-price auction hopes to recover the dominant-strategy outcome in order to maximize its profits.

In particular, the implementation literature focused on dominant-strategy (or strategy-proof) implementation for rational agents. The seminal paper by Dasgupta et al. (1979) provides a rationale for focusing on such strong-powered incentives: social choice functions that are Nash-implementable are also dominant-strategy-implementable when the preference domain is rich. Nothing is gained by relaxing incentives with rational agents, thus dominant-strategy implementation seems extremely desirable. It is a good starting point for a theory of implementation that is robust to the presence of adaptive agents.

We prove that a simple condition guarantees reinforcers will indeed converge on dominant strategies. Reinforcers's relative learning rates must be uniform across all actions. We will 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 an 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. This requirement is easily satisfied by any algorithmic system which adopts a ε-greedy policy, 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. In any game with a dominant 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 steady state, 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. Convergence is guaranteed 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. We propose instead a different interpretation of 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 effectively equalizes the relative rates of Q-learning, and similarly for other adaptive algorithms: 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 provides a powerful result.

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 steady state, 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 the payoff that would have obtained if they had played a different action, treating the

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

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 converge to the (unique) equilibrium.

Corollary 2 provides us with an intuitive interpretation of the learning speeds: different speeds amount to different abilities to work out basic counterfactuals. Following this thread, a natural question is whether uniform learning speeds can help in learning more sophisticated strategies. While elimination of dominated actions is a procedure understood by simple level-one rational agents, higher levels of sophistication can select strategies in more general environments.[11] Dominant strategies do not exist in many settings of strategic interaction, where equilibria instead do require a higher level of rationality.[12] For example, it is well known that players need level-k rationality for dominance-solvable games such that Iterated Elimination of Strictly Dominated Strategies (IESDS) terminates in k rounds.

Our next result shows that reinforcers actually achieve a special form of level-k rationality. We consider here IESDS where a strategy can be eliminated only when it is strictly dominated by another *pure* strategy. Under this assumption, the order of elimination is important in determining which profile(s) of actions survive the procedure, as shown in a simple example by Gilboa et al. (1990). The restriction to domination by pure strategies is necessary because the adaptive algorithms we defined do not deal well with mixed strategies.[13]

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 elimination 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 ε-greedy reinforcer with $RLS(a_i) = RLS(\tilde{a}_i)$ for all $a_i, \tilde{a}_i \in A_i$ for all $i \in N$. Additionally, let Assumption A3 be satisfied with a χ which preserves the rewards' order, i.e. if a_{-i} is the preferred profile of actions of agent i's*

[11] See the seminal paper on level-k rationality by Stahl II and Wilson (1994).

[12] E.g., consider first-price auctions and Bertrand pricing: in both cases, if the players can choose more than two actions, there is no dominant strategy.

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

opponent, $\mathbb{E}[r(a_i, a_{-i})] > \mathbb{E}[r(\tilde{a}_i, a_{-i})]$ *when* $r(a_i, a_{-i}) > r(\tilde{a}_i, a_{-i})$*. Then, there exists a* $\mathcal{T}$ *such that for all* $t \geq \mathcal{T}$ *all* $a_i \in \text{argmax}_a \theta^a(t)$ *are pure-rationalizable by the same IESDS.*

Importantly, Theorem 4 shows that, in a supermodular game, greedy reinforcers will converge to rationalizable strategies. If the equilibrium is unique, then convergence upon the unique equilibrium is guaranteed. In fact, in any pure-dominance-solvable game, reinforcers will converge on the pure-strategy Nash equilibrium.

In the next section we will leverage these result for implementation in hybrid markets. Furthermore, Section 6.2 and 6.3 show how Theorems 3 and 4 can be applied to study analytically environments that received considerable attention in the experimental literature.

5 Learning-Robust Mechanism Design

In this section, we develop an application of the relative learning rates results. We consider the problem of implementation of social choice functions. While strategy-proof implementation is the standard required by rational implementation, guaranteeing robustness of implementation to the presence of adaptive, learning agents, requires further considerations. In fact, in Section 4 we show how traditional strategy-proof implementation is vulnerable to the inadvertent coupling of adaptive agents that prevents convergence on dominant strategies. In this section we apply Theorem 3 to overcome this vulnerability and guarantee learning-robust implementation.

5.1 Model

We introduce our desiderata: mechanisms that guarantee implementation of social choice functions even when participants learn with reinforcers.

Concretely, we consider the basic setting of Dasgupta et al. (1979). Let $\mathcal{X}$ be the set of possible outcomes, and let there be n agents with types $(\lambda_i)_{i=1,\dots,n} \in \bigtimes_{i=1\dots n} \Lambda_i = \Lambda$. Type λ_i determines agent i's preferences $u_i \colon \mathcal{X} \times \Lambda_i \to \mathbb{R}$ over outcomes. A social choice rule (SCR) is a function $f : \Lambda \to \mathcal{X}$, that specifies an outcome for each type profile. Adopting the usual revelation principle, the designer chooses a direct mechanism $g \colon \Lambda \to \mathcal{X}$ such that it assigns an outcome to each possible report of the agents. We say a SCR f is implementable if (i) the set of equilibria $\mathcal{E}_g(\lambda)$ of the game described by g is non-empty for all type profiles λ; and (ii) $g(\mathcal{E}_g(\lambda)) = f(\lambda)$.

The seminal result of Dasgupta et al. (1979) relates Nash implementation with strategy-proof implementation: as long as the domain of preferences of the players is rich, the

two concepts are equivalent. We thus focus on strategy-proof (or dominant-strategy, or straightforward) implementation of a SCR f. That is, we are looking for a direct mechanism such that truth-telling is a dominant-strategy. Equivalently, f is straightforwardly implementable if

$$u_i(f(\lambda_i, \lambda_{-i}), \lambda_i) \geq u_i(f(\eta_i, \lambda_{-i}), \lambda_i) \qquad \forall \eta_i \neq \lambda_i.$$

Dasgupta et al. (1979) provide the following characterization of dominant-strategy implementable social choice rules:

Theorem (4.3.1)**.** *An SCR is truthfully implementable in dominant strategies if and only if it is independently person-by-person monotonic (IPM).*

We depart from the standard setting described so far in two important ways. First, we assume that a subgroup of agents $L \subseteq N$ each act according to the choices of their own reinforcer. Learning requires repeated interaction with the environment, therefore our second point of departure from the standard implementation setup is repetition of the implementation problem. However, we want to avoid incentives arising from repeated interactions, as well as manipulation of the algorithms by forward-looking participants. For this reason we assume that the subset of rational agents $R = N \setminus L$ is short-lived. In each period a fresh group of rational agents arrives and both rational and learning agents choose a report. The implementor then selects the outcome according to the mechanism she selected, payoffs are realized and rational agents leave the game. We call this setting a *hybrid* implementation problem, because rational and learning agents coexist.

We know that, when adopting a truthful strategy-proof mechanism in a hybrid market, rational agents will report truthfully. Will reinforcers learn to play the dominant strategy? As we have shown in Section 3, algorithmic agents may never do. Instead, they may collude together to obtain higher individual payoffs, in spite of the implementor's goals. We thus seek learning-robust truthful mechanisms.

Definition 8. A SCR is *learning-robust* truthfully implementable if:

- Rational agents play truthfully (i.e. report $\lambda_{\text{truthful}}$), and
- Reinforcers converge on the truthful report $\lambda_{\text{truthful}}$.

We say a mechanism is learning-robust if it learning-robust truthfully implements a given SCR.

In a learning-robust mechanism, adaptive agents will learn that reporting truthfully provides 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.

5.2 Feedback Design

In Parkes (2004) the author argues that mechanism design can play an important role in shaping algorithmic systems. The focus of that work is mainly how to improve learning speed and how to select the designer's preferred equilibrium when decision-makers are algorithms. Parkes 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"*. Here we formalize this intuition, and we show that feedback design can make traditional strategyproof settings robust to adaptive algorithmic players.

We do so leveraging our results, which guarantee that equilibrium is restored provided counterfactuals can be computed. Supplying enough information so that algorithms can evaluate counterfactual returns falls onto the mechanism designer. In this imperfect information setting, counterfactual calculations depend critically on the opponents' private information. The designer, who elicits all private information, must partly reveal it ex-post. We refer to this ex-post revelation as feedback provision.

Definition 9. A *feedback statistic* for mechanism f is a factorization of f through partitions. It is composed of the following elements:

- A *signal space* S, which is a partition of Θ_{-i};
- A map $\phi_i \colon \Theta_{-i} \to S$;
- A map $g \colon \Theta_i \times S \to \mathcal{X}$;

such that

- The diagram commutes, i.e. $f(\theta_i, \theta_{-i}) = g(\theta_i, \phi_i(\theta_{-i}))$ for all θ_i, θ_{-i};
- The map ϕ_i is a partition map, i.e. $\phi_i(\theta_i) \ni \theta_i$.

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

A feedback statistic determines the amount of information communicated by the designer *after* the outcome is realized. By accessing ex-post information, the algorithms can now compute counterfactuals: feedback statistics fully reveal what would have happened had a player submit a different report.

Proposition 3. *Under Assumption A3, a mechanism f augmented with feedback structure $(\phi_i)_i$ is* learning-robust *if:*

- *f is an independently person-by-person monotonic SCR; and*
- *$(\phi_i)_i$ is a feedback structure for mechanism f.*

The proof follows from Corollary 2. Notice that there always exists a learning-robust mechanism: it is one where each feedback statistic communicates all opponent's types truthfully.

Corollary 3. *Any SCR f which satisfies IPM is learning-robust truthfully implementable. The learning-robust mechanism which implements f is simply $(f,(\phi_i)_i)$ with $\phi_i\colon \Theta_{-i} \to \Theta_{-i}$ the identity map for all $i = 1,\dots,n$.*

This robust implementation relies on full-revelation: the designer does not hide anything from the participants in the mechanism. However, the full-revelation feedback structure might have some drawbacks. For example, full revelation might not be desirable, particularly when the implementation problem is repeated. Communicating all types also results in large communication complexity, and potentially high computational burden on the agents themselves, who need to re-compute the mechanism's allocation for each possible report. To formalize the desire for reduced communication, we define a partial order on feedback statistics:

Definition 10. Feedback statistic ϕ_i is *more private* than feedback statistic ψ_i, denoted $\phi_i \trianglerighteq \psi_i$, if $\phi_i(\theta_{-i}) \supseteq \psi_i(\theta_{-i})$ for all θ_{-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 statistics are comparable. However, it turns out that under the privacy order maximum and minimum are well-defined: the feedback statistics together with the privacy order form a lattice.

Proposition 4. *Feedback statistics 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 statistic definition. The argument follows from the lattice structure of the set of partitions with the partial order *coarser-than*. Note that $\phi_i^{-1}(\{x\colon x \in \Theta_{-i}\})$ defines a partition of the space Θ_{-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 the both the preimages of ϕ_i and ψ_i. Formally, let $A \subset \phi_i^{-1}(\{x\colon x \in \Theta_{-i}\}) \vee_P \psi_i^{-1}(\{x\colon x \in \Theta_{-i}\})$, then

$$(\phi_i \vee \psi_i)(\theta_{-i}) = (\phi_i \vee \psi_i)(\theta'_{-i}) \text{ for all } \theta_{-i}, \theta'_{-i} \in A$$

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 $B \subset \phi_i^{-1}(\{x\colon x \in \Theta_{-i}\}) \wedge_P \psi_i^{-1}(\{x\colon x \in \Theta_{-i}\})$, then

$$(\phi_i \wedge \psi_i)(\theta_{-i}) = (\phi_i \wedge \psi_i)(\theta'_{-i}) \text{ for all } \theta_{-i}, \theta'_{-i} \in B$$

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

From Proposition 4 it follows that there exist both a minimally- and a maximally-private feedback statistic. The minimally-private statistic is the full-revelation feedback statistic: it reveals all information, and it is clearly less private than any other feedback statistic. The maximally-private rule instead is interesting from the perspective of the market designer.

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

Corollary 4. *The communication complexity of a feedback statistic is monotonically decreasing in the privacy order.*

The maximally private feedback statistic is highly desirable: it has the lowest communication complexity and it maintains the highest level of privacy while supporting truthful implementation. We say a learning-robust mechanism is optimal if it robustly-implements the SCR f and it does so with maximal privacy (and consequently, minimal communication cost) for all agents.[14] Next, we derive the maximally private feedback statistic in the setting of efficient implementation with transferable quasi-linear utility.

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

5.3 Maximally Private Learning-Robust VCG

In this setting let $v_i(x)$ be the value of agent i for outcome x, and assume quasi-linearity in transfers. The goal of the designer is to select the allocation x^* such that

$$x^* = \operatorname*{argmax}_{x \in \mathcal{X}} \sum_{i=1}^{n} v_i(x).$$

Again we focus on truthful reporting mechanisms. One mechanism available to the designer is the well-known VCG mechanism. In particular, the mechanism with the Clarke pivot rule selects the optimal allocation for the given reports and assigns payments for agent i equal to

$$p_i = \sum_{j \neq i} v_j(x^*) - \max_{x \in \mathcal{X}} \sum_{j \neq i} v_j(x).$$

Given these payments, rational agents are incentivized to report truthfully: reporting their true values is a dominant strategy. In order to ensure implementation for an algorithmic system, however, we need to augment VCG with an appropriate feedback structure. The feedback from the traditional VCG is rather limited: with only the knowledge of the price for the implemented outcome, counterfactual returns for other actions are impossible to compute. We define a learnable mechanism, pVCG, and we show that its feedback structure is maximally private.

Definition 11. A pVCG mechanism is a pair of mechanism and feedback structure, such that:

- The allocation and payment function correspond to the VCG allocations and payments; and
- The feedback structure communicates to each agent a vector of personalized prices $(p_i(x))_{x \in \mathcal{X}}$, defined as

$$p_i(x) = \sum_{j \neq i} v_j(x) - \max_{x' \in \mathcal{X}} \sum_{j \neq i} v_j(x').$$

Intuitively, a pVCG mechanism provides each algorithmic player with personalized prices for all possible outcomes that could have been implemented. This menu aggregates the information about other player's valuations and allows to compute counterfactuals. Note that a player's feedback structure does not depend on that player's report. Therefore, each $p_i(x)$ is independent of the vector of reported $(v_i(x))_{x \in \mathcal{X}}$.

Proposition 5. *The pVCG mechanism implements the social optimum even when players bid through greedy adaptive agents.*

Proof. It suffices to show that each algorithm can compute its counterfactual return from reporting any vector $(\hat{v}_i(x))_{x\in\mathcal{X}}$. For any report $(\hat{v}_i(x))_{x\in\mathcal{X}}$, the player can compute which outcome $\hat{x}$ would have been implemented:

$$\hat{x} = \operatorname*{argmax}_{x\in\mathcal{X}} \hat{v}_i(x) + p_i(x)$$

Fixed the other player's reports, $p_i(x)$ represents the sum of other players' valuations for outcome x, up to a constant. It is then easy to compute the counterfactual return for reporting $(\hat{v}_i(x))_{x\in\mathcal{X}}$: player i would obtain $v_i(\hat{x}) + p_i(\hat{x})$. Now, because VCG is a strategy-proof mechanism, Corollary 2 applies and adaptive algorithms learn to play truthfully, thus implementing the social optimum. □

The feedback structure of pVCG is maximally private: for any coarser partition of the outcome space it is a simple exercise to produce a valuation for which the counterfactual is not uniquely determined. Thus, pVCG is the optimal learning-robust VCG mechanism.

Notice how in a second-price auction for a single item, the pVCG mechanism reduces to providing the lowest-bid-to-win to all the losers. Anecdotally, this practice is becoming more common in online auctions, even in games that do not have a dominant strategy (but can be solved by iterated elimination), as pointed out in Banchio and Skrzypacz (2022).

We offer a simple price-theoretic interpretation of pVCG in a unit-demand model with multiple goods for sale.

Example 4. *Assume that the auctioneer is selling goods $x_1,\dots,x_k$ to agents $1,\dots,n$ optimally. The pVCG mechanism requires each agent to communicate their values for each good to the auctioneer. After this communication, each agent is offered a personalized menu: the agent could purchase any goods at the price specified in the menu. Clearly consumers choose the good that maximizes their utility subject to the unit-demand constraint. In addition to consuming their favorite good, they can also infer the utility they could have obtained had they selected a different allocation: the menu provides prices for all goods $x_1,\dots,x_k$. Thus, even adaptive agents would converge on the dominant strategy, by means of counterfactuals evaluations.*

In this example, the feedback of pVCG is an aggregator of market information, which helps agents evaluating the true value of truthful reporting. As suggested by Parkes (2004), price signals serve as learning aids to individual adaptive agents.

6 Extensions

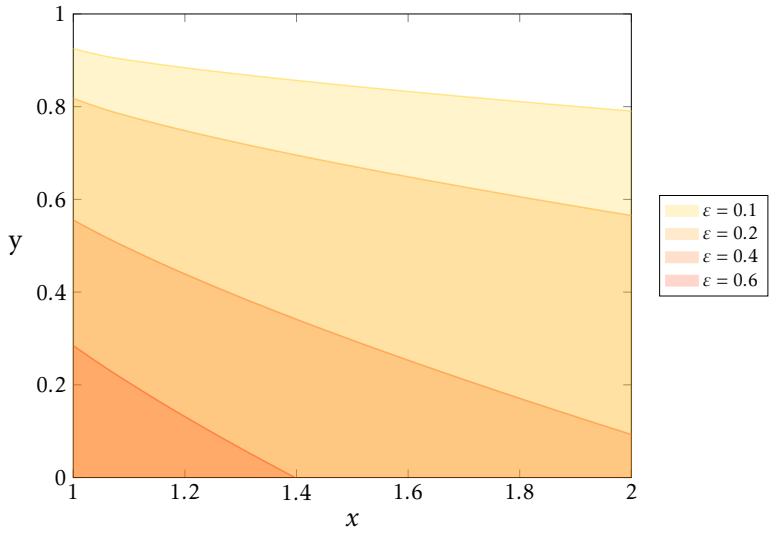
In this section we extend our analytic characterization, and we better position our results vis-a-vis the literature on algorithmic collusion by applying the arguments of Section 4 to two recent papers by Asker et al. (2022) and by Banchio and Skrzypacz (2022).

6.1 General Prisoner's Dilemma

Of course, the contribution game of Section 3 can be seen as a particular one-dimensional parametrization of the Prisoner's Dilemma. A complete parametrization of the incentives in a Prisoner's Dilemma is given in Friedman and Oprea (2012) and reproposed in Figure 7a. 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	Bob 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 7

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

Proposition 6. *There always exists a steady-state $q_D^{eq} \in \omega_{D,D}$. Moreover, there exists a pseudo-steady-state $q_C^{eq} \in \overline{\omega}_{D,D} \cap \overline{\omega}_{C,C}$ only when the following conditions are satisfied:*

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

The conditions for existence of a pseudo-steady-state appear complex, but the visualization in Figure 7b 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 clearly reflects these intuitions for various levels of exploration.

6.2 Bertrand Competition

Consider the setting of Asker et al. (2022). The authors simulate algorithmic competition in a Bertrand oligopoly, and find that collusion depends critically on the synchronicity of the algorithm. We show that their results can be derived analytically, and how competition is enforced only when players have access to counterfactuals with respect to their pricing decisions.

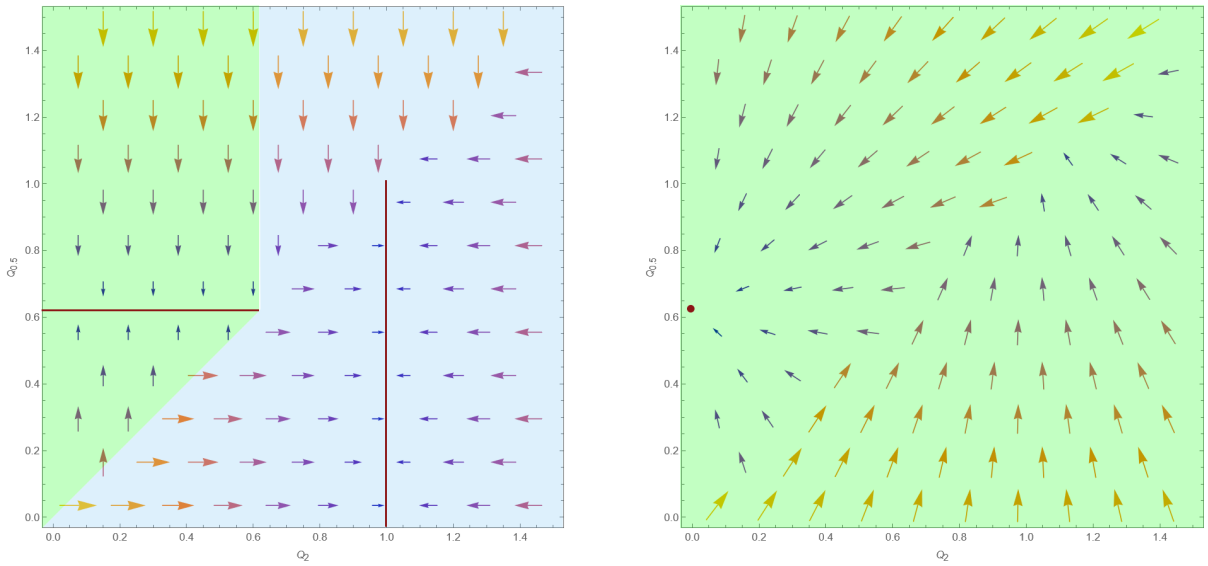
There are two firms, Alice Inc. and Bob Ltd., which face a common demand for their product. For a simple illustrative model, we assume that the market demand is $D(p_A, p_B) = 3 - \min\{p_A, p_B\}$, and if the two firms charge the same price they split demand equally. Suppose that each firm has 0 marginal 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 he could have obtained had he played the other action instead, but holding the opponent's action fixed.

We plot both systems in Figure 8 for $\gamma = 0$.[15] The figures depict the vector fields governing the dynamics of the continuous-time analogous of each Q-learning procedure, with the value of the competitive price on the vertical axis and that of the collusive price on the horizontal axis.

[15] The choice of $\gamma = 0$ reflects the specification of Asker et al. (2022), but we provide the vector fields for general values of γ, and notice that the same intuitions apply.



(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.

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 profoundly 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.

These results are robust to an ε–greedy specification: the *coordination bias* 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 biased during a competitive phase. The persistence of this bias

amounts to insufficient 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 intuition is that they overestimate the value of undercutting the opponent, because they do not internalize 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 equilibrium. Once the firms start 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. These result are not surprising, as Theorem 3 proves that collusion is unsustainable in a dominant-strategy game.

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.

Take for example the full model from Asker et al. (2022). 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 $P = \{p^1, \dots, p^{100}\}$. 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 7. *In a Bertrand oligopoly, if Alice Inc. and Bob Ltd. adopt any greedy reinforcers such that the relative speed of learning is the same across all prices, they either converge on E_1 or E_2.*

Proof. The proof of this proposition consists of verifying the assumptions of Theorem 4. As discussed earlier, Q-learning is a reinforcer, and ε-greedy policies guarantee that Assumption A3 is satisfied. By applying iterated elimination of strictly dominant strategies, only one of two pairs of prices can survive. Those pairs are exactly the equilibrium pairs. □

In particular, this result shows that the parameters of the algorithms are irrelevant for the convergence result. How fast the equilibria are reached will depend on the different discounting or learning rates, but convergence on an equilibrium is guaranteed.

6.3 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.

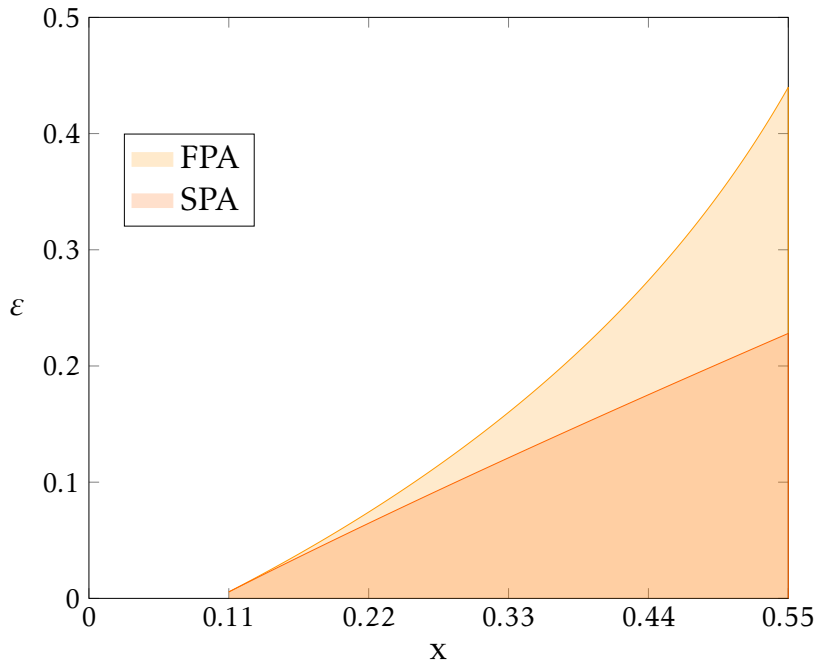
We interpret the cycling between collusive pairs as a form of sliding. To this end, consider the following simpler model, where Alice.com and Bob.net each have access only to two bids. The low bid is 0.1, while the high bid is denoted by x and varies in $[0.1, 0.55]$. Under these assumptions, both games are one-dimensional parametrizations of a Prisoner's Dilemma, and can therefore be mapped to the description of Proposition 6. Figure 9a and 9b shows the payoff matrix for both game formats, while Figure 9c plots the parametric region where a collusive equilibrium exists, for both FPA and SPA.

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

(a) Payoffs of the FPA.

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

(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

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 and coordination bias.

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 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:

Proposition 8. *In a first-price auction, if Alice.com and Bob.net adopt any greedy reinforcers with same relative speed of learning across bids, they converge on either E_1 or E_2.*

The proof is identical to that of Proposition 7 and is therefore omitted. In fact, first-price auctions and homogeneous Bertrand oligopolies are strategically equivalent. Banchio and Skrzypacz (2022) argues that Google's minimum-bid-to-win feedback in their first-price auction acts in the interest of Google, by simultaneously improving convergence and guaranteeing more competitive outcomes. The latter can be seen clearly in our framework, both in Proposition 8 as well as in the implementation problem we presented in Section 5: feedback restores the level-k reasoning which in turn guarantees successful implementation.

7 Conclusion

This paper analyses collusion in games played by online learning algorithms. We take a theoretical perspective and, moving from burgeoning empirical and numerical evidence, we identify the drivers of collusive behaviour in a widely studied class of games. In particular, we first address the issue of the analytical intractability of strategic interaction among algorithms by showing they can be approximated with a system of ODEs. Then we apply this framework to dominant-strategy games, and prove that (ε-)greedy algorithms can learn to collude. We identify the mechanism sustaining collusion in the *coordination bias*: when algorithms realize the benefit of the dominant action too slowly, 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 could help in understanding how AIs reach tit-for-tat strategies when given monitoring technology. As an example of the power of our method, we apply the

techniques to simple auctions in the spirit of Banchio and Skrzypacz (2022), where the same mechanism generates different results across payment rules.

We show that convergence on the dominant action is instead guaranteed for greedy algorithms if learning occurs at uniform rates for all actions. Algorithms that evaluate counterfactual rewards from actions not taken learn simultaneously for all actions, and thus converge on dominant strategies. The involuntary coupling at the heart of collusion may appear whenever learning rates are not uniform — differential learning rates is akin to oversampling bias.

Following these results, we design learning-robust mechanisms, which guarantee implementation in hybrid models where rational and learning agents coexist. Learning-robustness is guaranteed by a new layer in the mechanism design: ex-post feedback provision. We apply the hybrid implementation theory to design a mechanism, *pVCG*, that implements the social optimum in dominant strategies. pVCG provides personalized prices for all outcomes, aggregating the necessary information efficiently and allowing counterfactual evaluations for any report. 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 while recovering 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. Our work provides a theoretical analysis of collusion in games played by online learning algorithms. In doing so we prove a continuous-time approximation technique that can be applied more generally to the study of systems otherwise deemed intractable. We hope that this work will stimulate further analysis of the strategic interaction of learning algorithms. 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 coordination as a key driver of collusion, thus suggesting that offline algorithms may be less prone to collusive behavior.

We focused on implementation with myopic rational participants, but 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, G. W. (1951): "Iterative Solution of Games by Fictitious Play," in *Activity Analysis of Production and Allocation*, 374–376.

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.

Cesa-Bianchi, N., Y. Mansour, and G. Stoltz (2005): "Improved Second-Order Bounds for Prediction with Expert Advice," in *Learning Theory*, Springer Berlin Heidelberg, 217–232.

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.

Dasgupta, P., P. Hammond, and E. Maskin (1979): "The Implementation of Social Choice Rules: Some General Results on Incentive Compatibility," *The Review of Economic Studies*, 46, 185–216.

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.

Friedman, D. and R. Oprea (2012): "A Continuous Dilemma," *American Economic Review*, 102, 337–63.

Gilboa, I., E. Kalai, and E. Zemel (1990): "On the order of eliminating dominated strategies," *Operations Research Letters*, 9, 85–89.

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.

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.

Klos, T., G. J. van Ahee, and K. Tuyls (2010): "Evolutionary Dynamics of Regret Minimization," in *Machine Learning and Knowledge Discovery in Databases*, Springer Berlin Heidelberg, 82–96.

Kühn, K.-U. and S. Tadelis (2018): "The Economics of Algorithmic Pricing: Is collusion really inevitable?" Working paper.

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.

Stahl II, D. O. and P. W. Wilson (1994): "Experimental evidence on players' models of other players," *Journal of Economic Behavior & Organization*, 25, 309–327.

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.

Tuyls, K. and G. Weiss (2012): "Multiagent learning: Basics, challenges, and prospects," *Ai Magazine*, 33, 41–41.

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

Proof of Theorem 1. The existence of a solution for the Cauchy problem is guaranteed by assumption Assumption A1.

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 $\mathbb{R}^{d_i}$. 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 $d_i + 1$, 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 $0 < \tau_1 < \tau_2 < \tau_3 < \dots$. We define the process $\theta^1(t)$ as

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

for all times $t \geq 0$. The process $\theta^1(t)$ is a compound Poisson process, cadlag and Markov. Naturally, its $d_i + 1$ component always coincides with the iteration k. On average, 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 d_i components is[16]

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

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

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

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

$$\mu^n(x, dz) = \mathbb{P}\Big\{\theta^n(\tau^n) \in dz | \theta^n(0) = x\Big\}$$

where τ^n is the first exit time of Q^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

$$F^n(x)_m = n \int \frac{\alpha}{n} T_m(x, Y)\mu = \int \alpha\, T_m(x, Y)\mu.$$

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 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 $T(\theta, Y, t)$ must be bounded for all t. Let $M = \sup_{t,\theta\in E} T(\theta, Y, t)$ 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 $|F(x)-F(y)| \leq M|x-y|$ 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 $X(t,x_0), 0 \leq t \leq T, x_0 \in E$ 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 $\eta > 0$

$$\lim_{n\to\infty} P\Big\{\sup_{t\leq T} |X^n(t) - X(t,x_0)| \geq \eta\Big\} = 0$$

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

- since $F^n = F$ is linear, 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

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

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:

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

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

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

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:

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

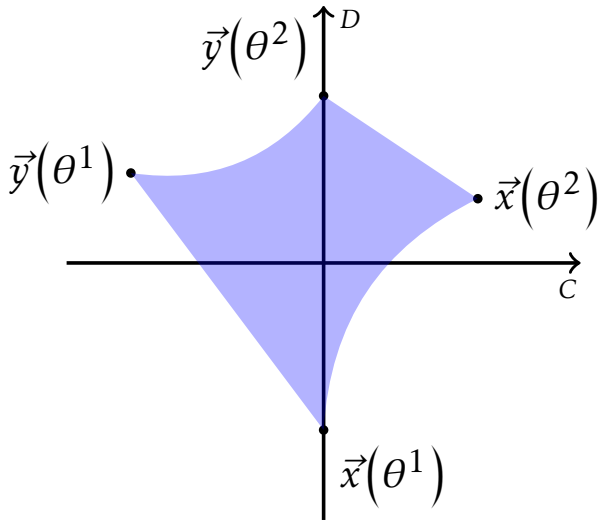


Figure 10: Homotopy

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

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

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 $H\colon [0,1]\times[0,1]\to\mathbb{R}^2$.

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

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.

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

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

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:

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

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:

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

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

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

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 $j\neq i$.[17] 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 differen-

[17] 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.

tial 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 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.

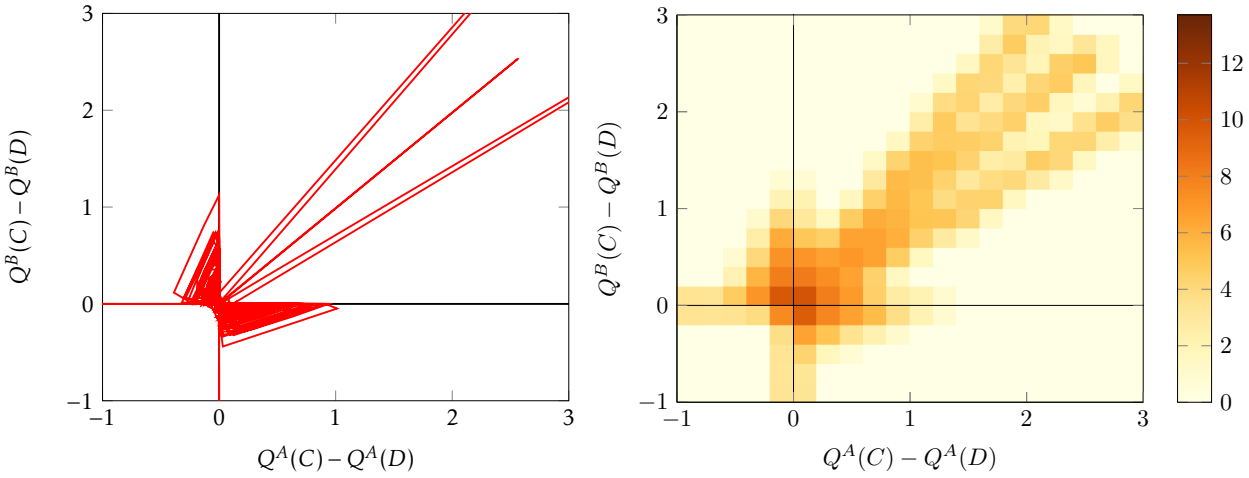


Figure 11: A chaotic trajectory of the system of Equation (4) in a 2-D representation. On the left, we depict the motion in the difference space. Agents appear chaotically attracted to a butterfly-like motion around the origin. The figure on the right represents the logarithmic density of time spent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

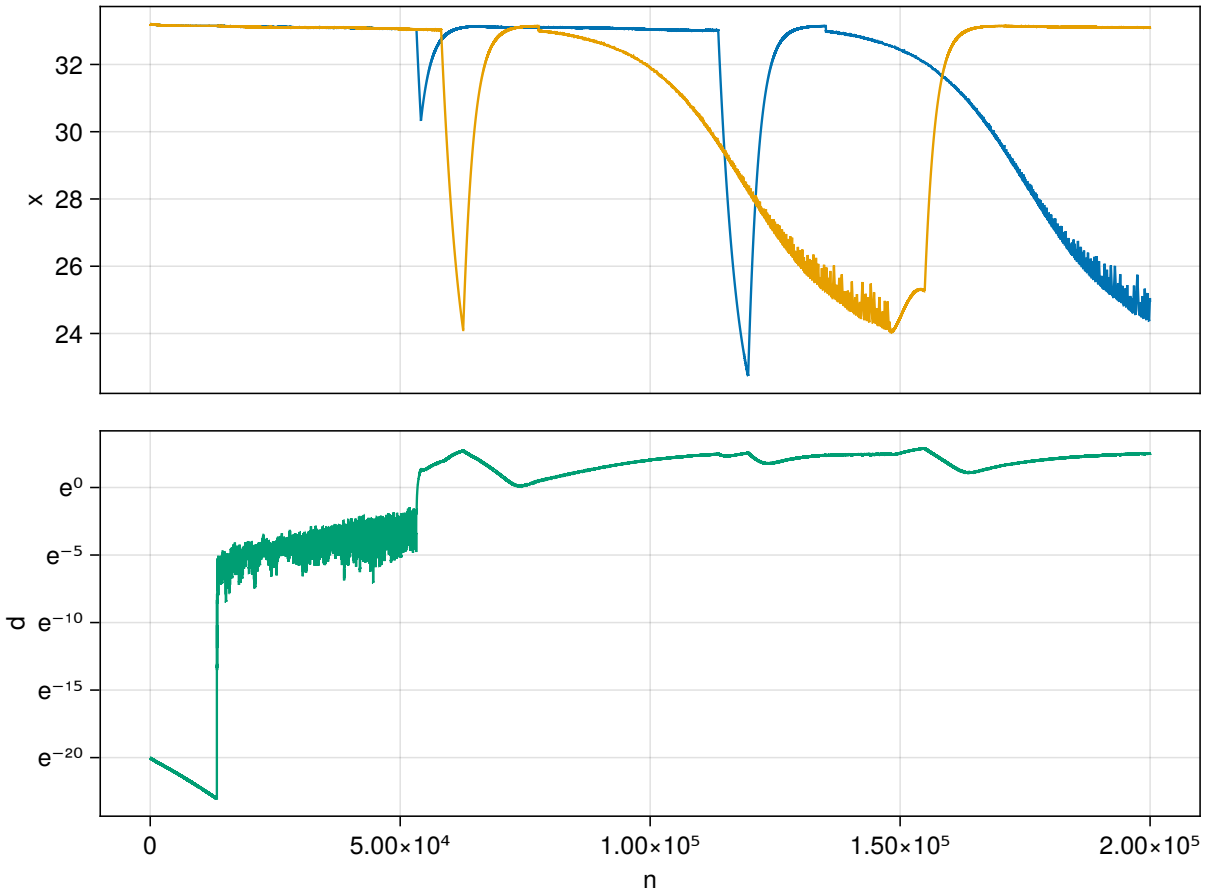


Figure 12: The top figure depicts the evolution of (the first component of) two trajectories with minimal perturbations of their initial conditions. The bottom figure represents the absolute distance between these two trajectories, in logarithmic scale. After an initial convergence, hitting the boundary 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.

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 6b: 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 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.