Stability of learning dynamics in two-agent, imperfect-information games

Butterworth, John M. and Shapiro, Jonathan L.

2009

MIMS EPrint: **2009.32**

ISSN 1749-9097

Stability of Learning Dynamics in Two-Agent, Imperfect-Information Games

John M. Butterworth
School of Computer Science
University of Manchester
jbutterworth@cs.man.ac.uk

Jonathan L. Shapiro
School of Computer Science
University of Manchester
jls@cs.man.ac.uk

ABSTRACT

One issue in multi-agent co-adaptive learning concerns convergence. When two (or more) agents play a game with different information and different payoffs, the general behaviour tends to be oscillation around a Nash equilibrium. Several algorithms have been proposed to force convergence to mixed-strategy Nash equilibria in imperfect-information games when the agents are aware of their opponent's strategy. We consider the effect on one such algorithm, the lagging anchor algorithm, when each agent must also infer the gradient information from observations, in the infinitesimal time-step limit. Use of an estimated gradient, either by opponent modelling or stochastic gradient ascent, destabilises the algorithm in a region of parameter space. There are two phases of behaviour. If the rate of estimation is low, the Nash equilibrium becomes unstable in the mean. If the rate is high, the Nash equilibrium is an attractive fixed point in the mean, but the uncertainty acts as narrow-band coloured noise, which causes dampened oscillations.

Categories and Subject Descriptors

I.2.6 [**Computing Methodologies**]: Artificial Intelligence—*Learning*

General Terms

Theory

Keywords

Co-adapting agents, game theory, reinforcement learning

1. INTRODUCTION

A challenging problem in learning algorithm design is the problem of convergence in learning algorithms for multi-agent co-adaptation. In many multi-agent scenarios, agents must produce behaviours independently (or autonomously), and the fitness of a given agent depends both on the behaviour it chooses and also on the behaviours chosen by the other agents. In the simplest case (considered here) there are two agents who are each choosing actions in order to try to optimise the payoff they receive, but the payoff each achieves when performing an action depends on the actions performed by the other agent. The agents try to learn the actions which optimise their payoff. When the agents employ population-based algorithms, such a situation is called co-evolutionary learning. For general learning algorithms, we shall call such a situation "co-adaptive learning".

In order to determine whether a co-adaptive learning algorithm is effective, we need to define what constitutes a solution to the learning problem. If only one agent is learning and the others are fixed, the problem reduces to an optimisation problem. In this case, the goal of the learning agent is to learn the *best-response* to the other agents, which is a behaviour that optimises the learning agent's payoff. An effective learning algorithm should find such an optimum in this situation.

When all agents are learning simultaneously, game theory[1] provides a formal description for multi-agent interactions, as well as a precise definition of solution points. Game theory defines a Nash equilibrium point as one where all agents are playing best-response to all the other agents. In other words, a Nash equilibrium point is one from which no agent can deviate unilaterally and improve its payoff. The celebrated result of Nash is that finite games always have such equilibria [12], but only when the possibility of probabilistic or "mixed" behaviours is included. It is sensible to assume that when all agents are endowed with the same learning algorithm, the system should converge to a Nash equilibrium. In this state, all agents will be performing in a way which is *mutually* optimal.

Unfortunately, devising co-adapting learning algorithms that converge to an attracting or asymptotically stable state can be problematic even for the simplest structure of actions and rewards [20]. The issue is particularly acute when learning agents must employ mixed strategies in order to locate a Nash equilibrium point. One instance where the convergence problem can occur is a two-player imperfect-information game. In this paper, we consider two-player games in which states during play are only partially observed, and agents must learn best-response to their opponent's strategy. Environments of this nature often require agents to undertake strategies containing a stochastic element in order to act optimally, and we consider only the hard problem of converging to mixed-strategy solutions. We study the convergence of gradient ascent in the small time-

[1]more specifically, non-cooperative game theory

step limit. We show that a specific algorithm which is stable when the opponents strategies are known, can become unstable when those strategies must be learned or estimated.

2. RELATED WORK

The issue of non-convergence in adaptive games has a long history, in a variety of learning approaches. The issue was first raised by Crawford, who showed in a series of papers that for a two-player game endlessly repeated, seemingly natural adaptation rules did not converge to mixed-strategy Nash equilibria: in zero-sum matrix games [4], in general-sum games [5], and in evolutionary games [6]. The problem of non-convergence in this context is sometimes referred to as the "Crawford Puzzle".

Much theoretical work in evolutionary learning in games has been done in the limit of infinite population, selection only, and continuous time, a limit also known as the replicator dynamics. For symmetric games, there is a special class of Nash equilibria, called evolutionary stable strategies (ESS), for which the learning dynamics converges. However, many games do not contain ESS equilibria; for example in zero-sum games the replicator dynamics is typically oscillatory or divergent. For asymmetric two-player games, which includes the games considered here, there are no asymptotically stable fixed points in the replicator dynamics [9, 10].

Another class of learning algorithms include those based on gradient ascent. Here the agent employs a probabilistic strategy, and learns by moving the probabilities in a direction which most increases its payoff. This moves the behaviour in a direction towards, but not all the way to, the best-response behaviour. Gradient ascent dynamics is similar to the evolutionary dynamics, in that it does not converge to mixed-strategy Nash equilibria in general. An in-depth study which reveals this was performed by Singh et al., in which the behaviour of infinitesimal gradient ascent (IGA) learning on simple two-player two-action games [20] was shown to result either in limit-cycle behaviour around a Nash equilibrium, or divergence. Other studies of IGA were carried out by Zinkevich [24].

Several solutions to dampen the oscillatory or divergent behaviour observed by seemingly natural adaptation rules have been proposed. Selten [18] introduced the concept of "anticipatory learning" in which two agents learning via gradient ascent do not update with respect to the current strategy of the opponent, but update instead to the *anticipated* strategy of the opponent. The anticipated strategy is obtained by calculating the next gradient ascent update of the opponent. This converges to mixed-strategy equilibrium points, but requires learning agents not only to possess the opponent's precise strategy upon each update, but also the opponent's update method.

Related to this is the stabilisation method used in this paper — the *lagging-anchor algorithm* of Dahl [7] for two-player games of imperfect information. The basis of the algorithm is gradient ascent, but with an added attractor on each update - the player's "lagging anchor". The anchor is simply a weighted average of all of the strategies that the player has employed during learning so far. The anchor acts by lagging behind the present state of the player's strategy, drawing the strategy towards it slightly. By doing so, the anchor dampens the oscillation found in basic gradient ascent, drawing the players toward the mixed-strategy Nash equilibrium. Dahl showed that for a subset of matrix games, the lagging anchor algorithm achieves exponential convergence when the agents are aware of the opponent's strategy [7].

More recent work on a related problem was performed by Bowling and Veloso [2]. They developed an algorithm for stochastic games - games where the only imperfection in an agent's state information is due to simultaneous moves by the players. Their algorithm was based on the principle that an agent should learn slowly if "winning" and learn quickly if "losing". They devised a method for agents to estimate whether to use the fast or slow learning rates without the need of knowing a Nash payoff, combining the technique with policy hill-climbing. Another algorithm for stochastic games was proposed by Conitzer and Sandholm [3]. They developed an algorithm for repeated games; stochastic games with a single state. The algorithm works on the principle that it plays a predetermined Nash equilibrium strategy unless it finds other players are stationary, in which case it adapts to the best-response strategy. We assume that a Nash equilibrium cannot be computed *a priori*, and so a group of learning agents cannot rely on a commonly chosen equilibrium before playing the game.

In the past, it has often been different communities who have studied the different approaches: population-based co-evolution and gradient-based or other reinforcement learning co-adaptation. However, there seem to be similarities in the results, and there are benefits in viewing the two approaches in a unified way. One of the great successes of reinforcement learning was the development of a backgammon-playing program by Tesauro [22]. It used TD-learning and self-play to achieve top-10 human level performance However, it has been argued that what was essential was not the form of the learning algorithm, but the fact that the system learned through self-play and co-adaptation [13]. Hence, it has been proposed that self-play can be used to compute effective actions in multi-agent environments. This can only be true if we find effective algorithms which can converge to useful behaviours.

3. RELATIONS TO OTHER WORK IN EVOLUTIONARY LEARNING AND REINFORCEMENT LEARNING

As FOGA is a venue for reporting work on evolutionary algorithms, and since we consider only two agents and do not use population-based methods, it is worth stating why we think this work is relevant and how it fits into other work. This is done in this section.

We reiterate the main feature of this work; we study the co-adaptation of two agents where the fitness of each depends on the behaviour of both. We are interested in learning algorithms which can converge to Nash equilibria that require mixed strategies for both players, which means that the agents need to be able to learn to produce probabilistic behaviour. We also assume that the two agents have different and incomplete information about the environment. This makes it difficult for an agent to infer the strategy that its opponent is using from the opponent's behaviour. It has been difficult to devise learning methods which learn probabilistic behaviour for multiple agents, and is particularly difficult in the case of imperfect information.

Let us start with the following question: what learning algorithm should an individual learning agent employ in the situation described above? The learning algorithm could be population-based, in which case the agent's mixed strategy would be represented as a population of pure strategies and the learning algorithm could be a genetic algorithm. Alternatively, the mixed strategy could be represented as a vector of probabilities, and stochastic gradient ascent [23] or some other reinforcement learning algorithm could be used. The key point is that learning probabilistic behaviour in heterogeneous agents is a challenging and unsolved problem in either approach. For example, consider the simple case where there are two agents and each agent has a single binary choice to make and there is no hidden information. Assume there is a Nash equilibrium in which each agent has a non-zero probability of each action. In the replicator dynamics, the Nash equilibrium is a fixed point, but it is not asymptotically stable (the dynamics will not converge to it). The fixed point is either a saddle point — stable in some directions and unstable in others, or is stable but not asymptotically stable — the dynamics oscillates (see, for example, section 10.4, page 119 of Hofbauer and Sigmund [10]). This is true in general for two-player, asymmetric games using the replicator equation. The replicator dynamics is equivalent to a genetic algorithm in two limits: the limit of a large population and the continuous-time limit, which is essentially the limit of a large number of generations with low selection pressure per generation. There is no evidence that relaxing these assumptions stabilises the fixed points, and oscillations in heterogeneous two-population co-evolving genetic algorithms have been shown experimentally (for example [16]). These oscillations may also partially explain the "mediocre states" phenomena found in early attempts at co-evolving problem-solvers and their test problems [8, 19]

Now consider the same two-agent, two-action, fully-mixed equilibrium problem using gradient ascent learning, also in the continuous-time limit. The result is exactly the same. Interior fixed points are either saddles or are stable but not asymptotically stable (i.e. they are oscillatory). This has been shown by Singh et al [20]. In that work, the authors assume the gradient can be computed exactly by the agent, which requires knowledge of the opponent's strategy, which we don't assume here. However, that unrealistic assumption is not the cause of the lack of asymptotic stability, and several authors have modified gradient-based algorithms to induce stability. The lagging anchor algorithm of Dahl [7] is one such algorithm; it stabilizes gradient ascent learning, although it requires the ability to compute the gradient. One of the contributions of our work is to study how the algorithm behaves when combined with gradient estimation methods.

Neither of the basic mechanisms of learning, evolutionary or gradient-based, will converge to mixed strategies. Another possible class of algorithms from the reinforcement learning community are value-estimation methods (e.g. Q-learning or TD-learning) [21]. These algorithms are not designed to learn probabilistic strategies, and modifications have been proposed to solve this, but none that work in the general case. In addition, it has been shown that Q-learning using a soft-max policy during learning is equivalent in the continuous-time limit to the replicator equations with a memory term [17]. The memory term does not stabilize the mixed equilibria. Rather, in examples studied, it destabilizes the oscillatory trajectories and can cause them to become chaotic.

Thus, returning to the original question: what learning algorithm should we use in this co-evolutionary setting? There is no easy answer. We consider gradient-estimation methods, because we can extend Dahl's algorithm which works for gradient ascent, and because it has the simplicity of linearity in normal-form games. Extension to population-based algorithms would be more difficult.

With respect to this work's place in with other work on co-evolution, another important distinction is between "competitive co-evolution" and "cooperative co-evolution". The current work falls under the first heading. The goal here is to address the question of which learning algorithm to put into an individual learning agent which is interacting with other agents on whose behaviour its fitness depends. We do not assume that the strategies or learning behaviour of the other agent(s) are known or can be controlled. The learning problem is viewed as being local to the agent. This is quite general. Such a learning agent can compete with agents similar to itself, or compete with instances of the problem it is trying to solve. It can be also be used in adversarial environments (e.g. a virus detector or spam filter competing with virus or spam designers) or a game-playing program competing with human players or programs designed by other programmers. For this reason, we do not assume that the player can use any information about its opponent that is not observable. In particular, we assume that the strategy used by the opponent cannot be observed; only the opponent's actions can be observed.

A lot of the work on co-evolving systems in the EA community is about building a global system to solve a problem out of co-evolutionary parts. This is called "cooperative co-evolution" [14]. In some approaches, each population represents a component of the global solution and the fitness of an individual is the average of fitnesses of itself combined with all (or a sample) of those sub-populations contributing different components. In this scenario, there is no reason that the learners cannot use information about the other systems. These systems have a global designer and are designed to satisfy a global goal. In addition, most of the work on cooperative co-evolution searches for pure strategies (although in the case of function optimization, they may be on a continuous space). In games with complete information and deterministic strategies, the classical methods, such as genetic algorithms and TD-learning, do work.

4. PROBLEM DEFINITION

In their study on multi-agent learning, Bowling and Veloso proposed two desirable criteria that a rational learning algorithm should fulfil [2]:

1. Against a stationary opponent, the algorithm should converge to the best-response strategy.

2. In self-play, the algorithm converges to a fixed point at which each agent is playing best-response to the other; i.e. a Nash equilibrium[2].

[2]Bowling and Veloso propose a more general criterion here, where the algorithm is required to converge to a stationary policy in general. They specify this is possible when the algorithm is pitted against another 'useful' learning algorithm (with similarly rational behaviour), or a stationary opponent as in the first criterion. Since we cannot prove that the algorithm converges against all other algorithms considered rational, a necessary but not sufficient condition for this behaviour is that the algorithm will converge against itself.

These criteria motivate the current work. It is clear that a learning agent should execute the first requirement when trying to maximise its reward, since playing best-response maximises the expected payoff for the agent by taking advantage of any suboptimal play by the opponent. The second criterion states that a group of agents, all of which are attempting to maximise their reward, and all using the same resources for learning, should converge to a stable fixed point where no agent may improve any further.

In addition to the above objectives, we believe the goal of a learning agent should be to optimize the amount of time spent playing best-response; hence maximising the payoff it receives. There is a weaker condition than those stated, which is; when playing against a slow-learning opponent (or quasi-stationary one), the agent should track the opponent's strategy and play best-response against it for as much of the time as possible. In other words, if the agent has sufficient learning resources to track its opponent, it should spend as much time as possible playing best-response. It may not be able to spend all its time doing so, because it may require some time to learn what the best-response is.

It must be recognised, however, that for any learning agent it may be possible to create another learning agent which can learn sufficiently quickly to defeat it. It would be interesting to find the learning agent for which no other learning agent can take advantage of in this way. However, this might not be possible. All we can do is consider learning algorithms which cannot be defeated by other learning algorithms which we can think of, a necessary but not sufficient condition for the agent to learn in this manner against arbitrary opponents.

A system which satisfies the above objectives will only converge to certain Nash equilibria; those with a special property which we will call *local learnability.* A Nash equilibrium is by definition a point at which no improvement is possible for any agent with the other agents strategies held fixed. We are interested in equilibria which are attractive when all strategies deviate slightly. Call an equilibrium point locally unlearnable if there exists a direction in the product of the strategy spaces of the agents which allows all agents to improve their payoff with a small move in that direction. A Nash equilibrium is locally learnable if it is not unlearnable. Whereas a Nash equilibrium disallows improvement through unilateral and global changes of strategies, locally learnable Nash equilibria also rule out improvement through multilateral local changes[3].

We assume a two-player game in which each player observes only part of the environment, $\mathcal{E}_k$ is observed by player k. Thus, an agent has information which is hidden from the other agent. The agents may have different objectives, and therefore unrelated payoff functions, which are denoted U_i for the *expected* payoff to the ith agent. Other assumptions we make are as follows:

1. An agent is aware of the rules of the game governing his possible action space.
2. Agents play a game repeatedly, learning and adapting their behaviour.

[3]Note that all mixed-strategy equilibria in zero-sum games are locally learnable since no direction can exist in which all players may adjust their strategies and each receive a greater payoff. In general-sum games, unlearnable mixed-strategy equilibria are possible.

3. At the end of each game, each payoff received is observed by the player who receives it.
4. There is a locally-learnable genuinely-mixed Nash equilibrium.
5. Each game is treated independently by each agent (e.g. by taking an independent draw from its probabilistic strategy).
6. Agents are unaware of the opponent's strategy.

An example of such a game is poker. Poker is a turn-based game, so the agent sees what the opponent does at each turn and responds accordingly. The hidden information is the private cards which are seen only by the player who holds them. The equilibrium solutions to poker which are known only in very simplified games are almost always mixed, which correspond to some degree of "bluffing". However, poker is a zero-sum game; one player's winnings are the others losses. We do not assume here that the game is zero sum.

We will consider two mathematical representations for player strategies from game theory. These are known as *normal form* and *extensive form.* A pure strategy is one in which a given action is always taken when faced with the same observable state. In normal form, a pure strategy is represented as a list of choices for *all* possible decisions which may be faced during play. A mixed strategy (one in which actions may be taken probabilistically), is then represented as a probability function over the set of pure strategies. In extensive form, we assign a probability function over all of the allowed actions from each (observably equivalent) state. It is more natural for a learning agent to express its strategy in extensive form, and also its model of the opponent, because single actions at decision nodes are precisely what are encountered during play. Normal form is preferred for analysis however, yielding simpler mathematics.

The learning algorithms we consider will be based on gradient ascent on the payoff function. In order for an agent to compute this gradient, it needs to know: the mathematical form of the payoff function, the strategy its opponent is playing, and all hidden information. Although this has been assumed in previous papers, we do not think this is realistic. The main focus of this paper is to investigate the convergence when the gradient needs to be estimated. However, we start by assuming that the gradient can be estimated by each player, to set notation and methods of analysis. Even in this case, convergence is an issue, as is shown in the next section.

5. CONVERGENCE WITH PERFECT KNOWLEDGE OF THE OPPONENT

To illustrate the issue of convergence and stability in gradient ascent learning, we first consider the situation where both agents know their opponent's strategy. We define a basic gradient ascent algorithm and analyse it in the infinitesimal time-step limit. This analysis builds on the approach developed in Singh et al. [20]. In a real game with knowledge of the opponent's strategy, there may be better ways to find the most effective best response than gradient ascent. This serves to illustrate the method and the issues.

5.1 Gradient ascent learning

Let $\mathbf{v}$ be the vector of parameters denoting the probabilistic strategy of player 1, and $U_1(\mathbf{v}, \mathbf{w})$ be a function which returns the expected payoff to player 1 if playing strategy $\mathbf{v}$ whilst the opponent plays strategy $\mathbf{w}$. The expectation is over the stochasticity of the environment as well as the agent behaviours. We restrict $\mathbf{v}$ and $\mathbf{w}$ to the spaces representing valid strategies for the players; they must represent valid probabilities.

We can now express the updates for both players using simple gradient ascent as follows:

$$\begin{aligned} \mathbf{v}_{t+\tau} &= \mathbf{v}_t + \tau \nabla_{\mathbf{v}} U_1(\mathbf{v}_t, \mathbf{w}_t) \ , \\ \mathbf{w}_{t+\tau} &= \mathbf{w}_t + \tau \nabla_{\mathbf{w}} U_2(\mathbf{v}_t, \mathbf{w}_t) \ , \end{aligned} \tag{1}$$

where $\nabla_{\mathbf{v}} U_1(\mathbf{v}_t, \mathbf{w}_t)$ is the directional derivative of the payoff function for player 1, with respect to his strategy $\mathbf{v}$, similarly evaluated for player 2. The step-size, τ could be 1.0 in an algorithm, but we will consider the small τ limit in analysing this system. It is clear that a learning agent following such an update rule adjusts its strategy in a direction which increases its expected payoff. In case equation (1) takes the strategy outside the probability simplex, it is projected back to the boundary.

At a Nash equilibrium, a necessary requirement is that neither agent may improve its payoff locally. Therefore, a Nash equilibrium must be a fixed point for a system obeying these dynamics, since a fixed point must possess the property that the gradient for both players is zero.

We are interested in the case where the strategies are genuinely mixed, which means at least two strategies for each player have non-zero probability. We project our equations into that space and do not consider the dynamics of the deterministic parts of the strategy. In normal form representation, the payoff functions are quadratic forms,

$$U_1(\mathbf{v}_t, \mathbf{w}_t) = \mathbf{v}_t^T \mathbf{E_1} \mathbf{w}_t + \mathbf{J_1} \mathbf{w}_t + K \ , \tag{2}$$

where $\mathbf{E}_k$ is a matrix of expected payoff to player k when player 1 plays pure strategy i and player 2 plays pure strategy j, and the i $(j)th$ component of the vector $\mathbf{v}$ $(\mathbf{w})$ is the probability that player 1 (2) plays pure strategy i (j). In the above, linear terms in player 1's strategy have been removed by a shift in the zero point to the Nash equilibrium, and the superscript T denotes matrix transpose.

5.2 Infinitesimal gradient ascent (IGA) does not converge

In the limit $\tau \to 0$, (1) and (2) yield,

$$\frac{d}{dt}\begin{pmatrix} \mathbf{v}_t \\ \mathbf{w}_t \end{pmatrix} = \begin{pmatrix} \mathbf{0} & \mathbf{E_1} \\ \mathbf{E_2} & \mathbf{0} \end{pmatrix}\begin{pmatrix} \mathbf{v}_t \\ \mathbf{w}_t \end{pmatrix} \ . \tag{3}$$

As the above is a linear set of first-order differential equations, of the form $d\mathbf{x}/dt = \mathbf{E}\mathbf{x}$, the solutions will be modes defined by the eigenvectors of $\mathbf{E}$. The character of each solution is determined by the corresponding eigenvalue. Imaginary parts of eigenvalues lead to oscillations; while the negative real parts lead to exponential decay, and the positive real parts result in exponential growth.

Since $\mathbf{E}$ is real and has zero trace, its eigenvalues come in complex conjugate pairs and the real parts must sum to zero. Thus, either: 1) all of the eigenvalues are pure imaginary, or 2) some of the eigenvalues have positive real part. Case 1 results in dynamics which oscillate around the Nash equilibrium but are not attracted toward it. Case 2 corresponds to dynamics which deviate away from the equilibrium. Case 2 also corresponds to the locally unlearnable situation described in Section 4. Thus, from here on, we will consider only matrices with pure imaginary eigenvalues. Hence, IGA does not result in a learning algorithm which converges to the equilibrium point even when the agents know each others strategies. It is marginally stable (neither converges nor diverges). This was shown by Singh et al. for two-action strategies [20].

The above analysis assumed normal form representation of the strategy. The analysis can be extended to extensive-form representations, but only locally by Taylor expanding around the equilibrium point to linear order. Let $\mathcal{F}_v$ be the map from the extensive form representation to the normal form representation for player 1, and let $\partial\mathcal{F}_v$ be its derivative, and likewise for player 2. If these maps are invertible and their Jacobians are non-zero[4], then the payoff equations can be expanded around the equilibrium value. The result is of the same form as (3), where $\mathbf{E_1}$ is transformed by the congruency transformation $\partial\mathcal{F}_{\mathbf{v}}{}^T \mathbf{E}_1 \partial\mathcal{F}_{\mathbf{v}}$ and likewise for $\mathbf{E}_2$. This system has the same structure locally, and so the same conclusions about stability can be drawn. However, the transient behaviour may be very different.

5.3 The lagging anchor: a mechanism for stabilising gradient ascent

In the previous section we showed that even when the players know each others strategies, gradient ascent in the small time-step size limit did not converge to the Nash equilibrium, either in normal or extensive form representations. The difficulty of convergence in multi-player games has been known for some time, and several mechanisms have been proposed to address this, including: WoLF [2], anticipatory learning [18] and the lagging anchor algorithm [7], which we now consider. We analyse this algorithm here using the same techniques as in the previous section, before providing our extension and analysis of the algorithm to when it must learn the gradient *without* knowledge of the opponent's strategy in Section 6.

Two additional terms are required for the definition of the lagging anchor algorithm to extend basic gradient ascent learning. Firstly, we use $\bar{\mathbf{v}}$ to denote the anchor for player 1's strategy $\mathbf{v}$, similarly for player 2. Secondly, we now introduce a new parameter, the anchor drawing factor, represented by η. The update equations now become:

$$\begin{aligned} \mathbf{v}_{t+\tau} &= \mathbf{v}_t + \tau \nabla_{\mathbf{v}} U_1(\mathbf{v}_t, \mathbf{w}_t) + \tau\eta(\bar{\mathbf{v}}_t - \mathbf{v}_t) \ , \\ \mathbf{w}_{t+\tau} &= \mathbf{w}_t + \tau \nabla_{\mathbf{w}} U_2(\mathbf{v}_t, \mathbf{w}_t) + \tau\eta(\bar{\mathbf{w}}_t - \mathbf{w}_t) \ , \\ \bar{\mathbf{v}}_{t+\tau} &= \bar{\mathbf{v}}_t + \tau\eta(\mathbf{v}_t - \bar{\mathbf{v}}_t) \ , \\ \bar{\mathbf{w}}_{t+\tau} &= \bar{\mathbf{w}}_t + \tau\eta(\mathbf{w}_t - \bar{\mathbf{w}}_t) \ . \end{aligned} \tag{4}$$

Now, on each learning iteration, a strategy is adjusted to increase expected payoff as before, but is also drawn toward its anchor by a factor η. In addition, each anchor is drawn toward its corresponding strategy in proportion to the difference between the current strategy and the anchor.

It is assumed that $\tau\eta < \frac{1}{2}$, so that it is not necessary to restrict $\bar{\mathbf{v}}$ and $\bar{\mathbf{w}}$ to valid strategies. A point $(\mathbf{v}, \mathbf{w}, \bar{\mathbf{v}}, \bar{\mathbf{w}})^T$ is only a fixed point of the lagging anchor dynamics if $(\mathbf{v}, \mathbf{w})^T$

[4]In general, normal form is over-parametrized whereas extensive form is not; and this over-parametrization must be removed for the map to be invertible.

is a fixed point of the gradient ascent dynamics, and also $\mathbf{v} = \bar{\mathbf{v}}$ and $\mathbf{w} = \bar{\mathbf{w}}$.

5.4 Infinitesimal Lagging Anchor (ILA)

Dahl showed that due to the dampening of the oscillation caused by the anchors, these dynamics achieve exponential convergence to genuinely-mixed strategy equilibrium points in a subset of matrix games [7]. This can be shown in the infinitesimal time-step limit as follows. We assume that the IGA matrix $\mathbf{E}$ in (3) has pure imaginary eigenvalues $\pm i b_n$, since this corresponds to local learnability. Here n indexes the eigenvalues. The differential equation corresponding to the lagging anchor equation has the following coupling matrix,

$$\left(\begin{array}{c:c} \mathbf{E} - \eta\mathbf{I} & \eta\mathbf{I} \\ \hdashline \eta\mathbf{I} & -\eta\mathbf{I} \end{array}\right) ,$$

where $\mathbf{E}$ is the matrix for IGA and $\mathbf{I}$ is the identity matrix. Eigenvalues of this partitioned matrix can be found using the identity,

$$\det\left(\begin{array}{c:c} \mathbf{A} & \mathbf{B} \\ \hdashline \mathbf{C} & \mathbf{D} \end{array}\right) = \det\mathbf{D}\det\left(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C}\right) , \qquad (5)$$

to simplify the characteristic equation $\det(\mathbf{M} - \lambda\mathbf{I}) = 0$, which factorises into quadratic equations involving the eigenvalues of IGA. The result is that each complex conjugate pair of IGA eigenvalues $\pm i b_n$ splits into four,

$$-\eta \pm \sqrt{\eta^2 - (b_n/2)^2} \pm i\frac{b_n}{2} , \qquad (6)$$

the real part of which is always negative. Thus, ILA always converges to the Nash equilibrium.

6. STABILITY OF GRADIENT ASCENT USING AN UNBIASED ESTIMATE OF THE GRADIENT

In the above, we found that even with perfect knowledge of the opponent's strategy, gradient ascent is only marginally stable; it does not converge to locally-learnable mixed-strategy equilibria (in the infinitesimal time-step limit). However, using the lagging anchor mechanism it always converges. From here on we remove the assumption that the agent knows the opponent's strategy. The player must learn how to respond to the opponent by observing results of repeated play. We show under a fairly generic set of assumptions, gradient ascent is always unstable in the mean.

We assume that after each game, the player can obtain information through observations which allows the player to infer a contribution to the gradient which is unbiased (two ways to do this will be explained momentarily). This means that the player observes a random variable whose mean is the gradient of that player's payoff function. However, because this contribution will come from only one game, it will be very noisy. Although the estimate is unbiased, it has a large variance, too large to be useful. Thus, it is necessary to average this gradient by accumulating it over several games.

To accumulate these contributions and smooth the estimate, we use what we believe to be the standard approach from on-line stochastic approximation [15]. If $\tilde{q}_t$ is the estimate of a quantity whose real value is q_t, and $\mathcal{O}_t$ is an observation which gives unbiased information, i.e.

$$\langle \mathcal{O}_t \rangle = q_t , \qquad (7)$$

where $\langle\cdot\rangle$ denotes expectation, a standard on-line update equation is,

$$\tilde{q}_{t+\tau} = (1 - \tau\phi_t)\tilde{q}_t + \tau\phi_t\mathcal{O}_t . \qquad (8)$$

Here the time-step τ is introduced so that the infinitesimal limit can be taken. If the value of q was constant in time, it would be appropriate to reduce the modelling rate $\tau\phi_t$ over time, e.g. by using an on-line updating procedure for counts or pseudo-counts in the case of Bernoulli variables. However, in the current situation, the player will be estimating quantities which depend on the opponent's behaviour which will change as the two players co-adapt. Thus the modelling rate must not reduce too quickly; a standard approach is to keep it constant (see, for example [21]). We assume that from now on, and drop the t subscript from ϕ.

If $\tau\phi = 1$, the estimate is based on information from the most recent observation only; the estimate is unbiased by assumption but is very noisy. As $\tau\phi$ decreases, the variance in the estimate is expected to decrease in proportion to ϕ, but $\tilde{q}$ is not unbiased because it is based on observations after which q may have changed.

6.1 Opponent modelling

The most straightforward way for the player to infer gradient information from game play is to explicitly model the player's strategy. This requires that the player can observe the opponent's actions and, after each game has completed, the hidden state information. (We do not assume that the information is observed during play, because that would change the nature of the game.) In order for a learning agent to be capable of explicitly modelling its opponent, we must introduce two additional assumptions to those in Section 4:

1. Agents are aware of the action spaces of both players.
2. An agent can observe the actions taken by the opponent during play.

The agents play a game repeatedly, learning their opponent's strategy on-line. The agents then use the opponent model, which is an estimate of the true behaviour of the opponent, in their update calculations. The goal of opponent modelling is to track the probabilistic behaviour of the opponent in real time, so as to play best-response to the strategy being executed.

Each player has a variable which represents the current estimate of the opponent's strategy. So, player 1 has a vector $\tilde{\mathbf{w}}$ which is the estimate of player 2's strategy vector $\mathbf{w}$. The players update their estimates by observing the opponent's actions and using equation (8). Since the player knows what action the opponent used and after the fact the hidden state, the player observes a single unbiased sample from the probabilistic strategy. In computing the gradient, the estimate $\tilde{\mathbf{w}}$ rather than the actual strategy $\mathbf{w}$ is used, i.e. $\nabla_{\mathbf{v}} U_1(\mathbf{v}_t, \tilde{\mathbf{w}}_t)$ in equation (1).

6.2 Stochastic gradient ascent

It is also possible to estimate the gradient without assuming that the player observes hidden environment or has an explicit model of its payoff function. The idea is to estimate the gradient directly from the payoffs received in each game. Consider this from the perspective of player 1.

If the strategies of the two players and the environment was fully known, the utility to each player would be determined. Let α and β denote the pure strategies played by players 1 and 2 respectively in a particular game with environment $\mathcal{E}$. (The environment variable $\mathcal{E}$ describes the entire environment, the parts seen by both players, as well as the parts private to one or the other.) The payoff to player 1 in this particular game is $u_1(\alpha, \beta, \mathcal{E})$. The probability of the two players playing these strategies depends on their parameters, and on the environment. Thus, we can write that the expected payoff to player 1 obeys,

$$U_1(\mathbf{v}_t, \mathbf{w}_t) = \sum_{\alpha\beta\mathcal{E}} P(\alpha, \beta|\mathbf{v}_t, \mathbf{w}_t, \mathcal{E}) P(\mathcal{E}) u_1(\alpha, \beta, \mathcal{E}) \ . \quad (9)$$

An obvious point is that one could use the payoff of a single game, $u_1(\alpha, \beta, \mathcal{E})$, as an unbiased estimate of the expected payoff $U_1(\mathbf{v}_t, \mathbf{w}_t)$. This is because the former occurs with probability $P(\alpha, \beta|\mathbf{v}_t, \mathbf{w}_t, \mathcal{E})P(\mathcal{E})$, so averaging $u_1(\alpha, \beta, \mathcal{E})$ is equivalent to taking the sum in equation (9).

The gradient of the payoff with respect to the player 1 parameter v_j is a similar sum,

$$\begin{aligned} &\nabla_{\mathbf{v}_t} U_1(\mathbf{v}_t, \mathbf{w}_t) \\ &= \sum_{\alpha\beta\mathcal{E}} P(\mathcal{E}) \nabla_{\mathbf{v}_t} P(\alpha, \beta|\mathbf{v}_t, \mathbf{w}_t, \mathcal{E}) u_1(\alpha, \beta, \mathcal{E}) \end{aligned} \quad (10)$$

Since each game samples one term in the sum, approximating the sum by individual terms is an example of stochastic gradient ascent. However, in order to make the estimate unbiased, we must approximate the gradient by,

$$\nabla_{\mathbf{v}_t} U_1 \approx u_1(\alpha, \beta, \mathcal{E}) \frac{\nabla_{\mathbf{v}_t} P(\alpha, \beta|\mathbf{v}_t, \mathbf{w}_t, \mathcal{E})}{P(\alpha, \beta|\mathbf{v}_t, \mathbf{w}_t, \mathcal{E})}. \quad (11)$$

The denominator is needed to correct the probability of reaching each term in the gradient sum, equation (10). Here we use $\approx$ to denote our approximation or estimate. This is our estimate of the gradient to use in equation (8). It can also be written,

$$\nabla_{\mathbf{v}_t} U_1 \approx u_1(\alpha, \beta, \mathcal{E}) \nabla_{\mathbf{v}_t} \log P(\alpha, \beta|\mathbf{v}_t, \mathbf{w}_t, \mathcal{E}).$$

(This approach is closely related to the REINFORCE algorithm proposed by Williams [23] and shown to give unbiased estimates of gradient ascent.)

This approach does not at first sight appear practical, however. The probability in the above equation, appears to be unknown to a player, because it depends upon the opponent's strategy. However, in both normal form and extensive form, the ratio in equation (11) *is* observable. This is because the probability $P(\alpha, \beta|\mathbf{v}_t, \mathbf{w}_t, \mathcal{E})$ factorises into factors which depend only on one player's strategy or the other's and is linear in components of the parameter $\mathbf{v}_t$. The factorisation is due to the fact that each player makes decisions independently of the other player when conditioned on the observed behaviour up to the point of the decision. The factors depending on the opponent's strategy cancel in the ratio.

Thus, to get an unbiased estimate of the gradient sums, one needs to play a game, observe the payoff at the end of the game and multiply this by a weight given by the gradient of the probability as given in equation (11) to estimate the gradient. Each player maintains the vector of estimates $\mathbf{g}_i$ which are computed from equation (11) and are updated using equation (8). This is then used instead of the gradient in equation (1).

Using either method for estimating the gradient, either by estimating the opponent's strategy from its behaviour or by estimating the gradient using stochastic gradients, and taking the infinitesimal limit, one gets a stochastic differential equation. We can derive the dynamics for the mean behaviour. Because the system is linear in normal form, or by approximating it as linear around the Nash equilibrium in extensive form, the mean of the dynamical equations describes the dynamics for the mean of the variables. In the infinitesimal time-step limit, using the same methods used in Section 5.4, the eigenvalues of this system can be expressed in terms of the eigenvalues of the original IGA system. Opponent modelling and stochastic gradient ascent are essentially equivalent in that both methods result in the same characteristic equation and the same eigenvalues. Each complex conjugate pair of IGA, $\pm ib_n$ splits into four,

$$\frac{-\phi \pm \sqrt{\phi^2 \pm i4b_n\phi}}{2} \ ; \quad (12)$$

the real parts of these are respectively:

$$\begin{array}{c} \frac{-\phi}{2} + \frac{\sqrt{2}}{2}\sqrt{\phi^2 + \sqrt{\phi^4 + 16{b_n}^2\phi^2}} \ , \\ \frac{-\phi}{2} - \frac{\sqrt{2}}{2}\sqrt{\phi^2 + \sqrt{\phi^4 + 16{b_n}^2\phi^2}} \ . \end{array} \quad (13)$$

For all positive ϕ the first one is positive and the second is negative. Since there are always eigenvalues with positive real parts in the coupling matrix, using stochastic approximation to update the estimates of the gradient always results in unstable dynamics in the mean and in the small time-step limit. Therefore, the system diverges from the Nash equilibrium.

6.3 Stabilising estimated gradients with the lagging anchor algorithm

Since use of an estimated gradient destabilises gradient ascent and the lagging anchor algorithm stabilises it, it is natural to combine the two in the hope of producing a stable algorithm. Dahl does not consider the task of estimating the gradient without knowledge of the opponent's strategy [7], and here we extend the algorithm to allow this, analysing the resulting dynamics. This new algorithm which combines lagging anchor with gradient estimation that requires no knowledge of the opponent's strategy may compete against an arbitrary opponent, since observations during play are all it requires to learn. This allows the algorithm to be applied more generally, but we still wish to retain the desired behaviours specified in Section 4.

Strategy updates and estimate updates are as before, but now there are anchors associated with the strategy variables but not with the estimates. Again the characteristic equation holds for opponent modelling and stochastic gradient ascent.

This system can be analysed using the techniques of previous sections. This set of stochastic equations is solved in the mean by taking the infinitesimal time-step limit and finding the eigenvalues. The matrices form a 3×3 block matrix system, rather than the 2×2 blocks shown in Section 5.4. Opponent modelling and stochastic gradient both yield the same characteristic equation. The characteristic equation factorises into cubic equations involving the eigenvalues of IGA and the two parameters, ϕ and η. Thus, each eigenvalue of IGA splits into three, the solutions (for λ) of,

$$\lambda^3 + (2\eta + \phi)\lambda^2 + \phi(2\eta \pm ib_n)\lambda \pm ib_n\eta\phi = 0 \ . \quad (14)$$

It is found that the eigenvalues can have real parts which are either positive or negative, depending on the value of the parameter η and the opponent modelling rate ϕ. The dividing point is a phase boundary, which is where the real part of the eigenvalue associated with the largest b_n becomes 0. Figure 1 shows this boundary and the two phases. This figure also shows asymptotic approximations to the curve: for large η it is approximately $\phi = 2\eta$, independent of the eigenvalues of IGA. This observation can facilitate setting these parameters for general problems — η needs to be set sufficiently large, then ϕ can be taken as 3η or similar. For small η it is approximately $\phi = b_n^2/\eta$.

Thus, the lagging anchor algorithm with estimated gradient can have two behaviours. In the region labelled "divergent in the mean", the expected values of the strategies diverge away from the Nash equilibrium. It will oscillate, but with increasing amplitude (oscillations due to the imaginary part of the eigenvalues; divergence due to the positive real parts). In the region labelled mean convergence, the expected values of the strategies converge to the Nash equilibrium. However, this does not ensure that the system converges in individual runs. The noise caused by the probabilistic behaviours being modelled results in dampened (or noisy) oscillations about the equilibrium. Figure 6.4 shows results of a simulation.

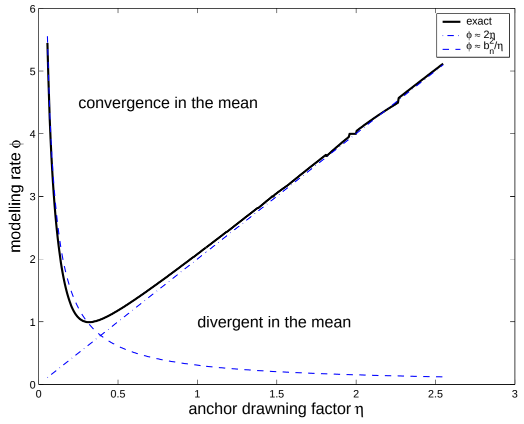

Figure 1: The phase boundary between divergent behaviour and convergence in the mean for ILA with estimated gradient.

6.4 The nature of the phase transition

In the previous section we found a phase transition in the mean behaviour of lagging anchor gradient ascent, using an estimated gradient. The mean behaviour is stable in one region of parameter space and unstable in the other. However, mean behaviour does not necessary tell us about what happens in the real system, particularly in the "stable region" since stability in the mean does not necessarily imply stability in individual runs. In fact, simulations show the divergent nature of the unstable region, but in the stable region the system is still noisy and it is hard to tell from simulations whether the system is oscillating.

To address this question, we consider a temporal covariance function

$$\mathcal{C}(\Delta) = \left\langle \left\langle (\mathbf{x}(t+\Delta) - \langle \mathbf{x}(t+\Delta)\rangle)(\mathbf{x}(t) - \langle \mathbf{x}(t)\rangle)^T \right\rangle \right\rangle_t .$$

Here $\mathbf{x}$ is the column vector of variables, superscript T de-

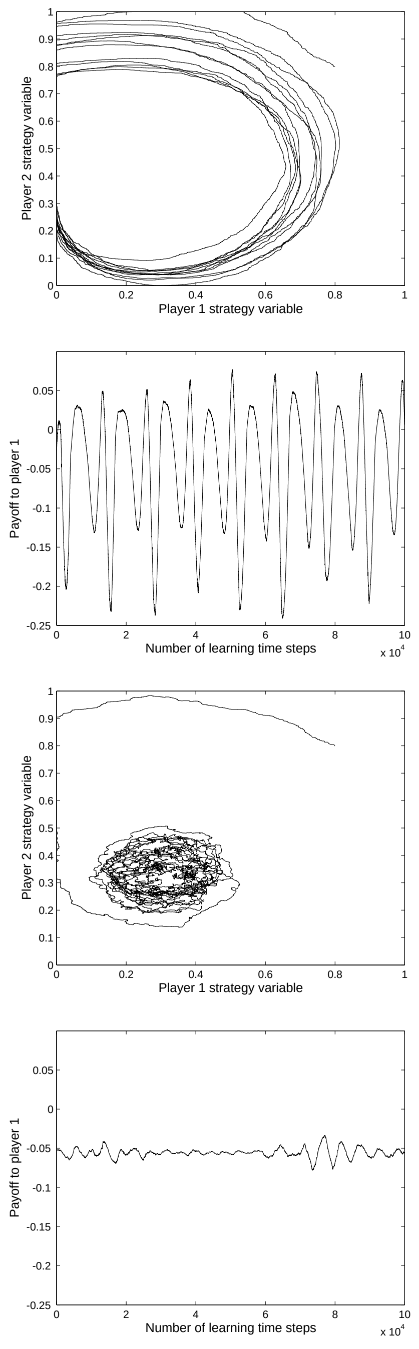

Figure 2: Results of lagging anchor algorithm with opponent modelling on Kuhn poker [11] using an extensive form representation. Strategy plots show single runs of 150,000 games for strategy variables against each other: one for each player, whose equilibrium is at $(1/3, 1/3)$. Payoff plots show the payoff to player 1 over the same run of games. Top plots are in the divergent phase ($\phi = 1.0$). Bottom plots are in the phase which converges in the mean ($\phi = 9.0$). Other parameters are for all plots: $\tau = 0.01$; $\eta = 2.0$. The phase boundary is located at $\phi = 4$.

notes matrix transpose and $\langle \cdot \rangle_t$ denotes time average. Thus $\mathcal{C}(0)$ is the covariance matrix and $\mathcal{C}(\cdot)$ generally shows the correlation between variables at different times. If the system behaves like white noise, this will be a delta-function at $\Delta = 0$; if the system is periodic, $\mathcal{C}$ will show peaks separated by Δ equal to the period of oscillations.

The temporal correlation function obeys the same differential equation with respect to Δ as that of the mean of the variables with respect to t. It has the same eigenvalues. As the phase boundary is approached from above, there is only one eigenvalue pair with a positive real part; where that real part changes sign defines the phase boundary. The inverse of the real part of that eigenvalue is a diverging timescale as the boundary is approached.

Let $\lambda = a \pm ib$ denote this eigenvalue and $\mathbf{e}^r \pm i\mathbf{e}^i$ the associated eigenvector. A property of the mean dynamics is that if the variables start in the space spanned by the two real vectors $\mathbf{e}^r$ and $\mathbf{e}^i$, it will stay in that subspace. This holds for the temporal covariance, which obeys the same dynamics. A useful statistic is the trace of the covariance projected into this subspace. Call that $\Pi\mathcal{C}(\Delta)$, it obeys

$$\Pi\mathcal{C}(\Delta) = e^{a\Delta}\left[A\cos(b\Delta) + B\sin(b\Delta)\right] \quad . \tag{15}$$

Here A and B are constants. Thus, as a changes sign, this goes from diverging with time constant $1/a$ to decaying with characteristic time $1/|a|$. When $a = 0$ it oscillates with period $2\pi/b$. This result shows that in the mean convergent region, where $a < 0$, there is still temporal correlation in the form of narrow-band coloured noise, which is found in damped harmonic motion [1]. In fact, the system is much like an damped harmonic oscillator coupled to a white noise source. It continues to oscillate because the noise continuously kicks the system.

7. CONCLUSIONS

We have considered two agents competing in a general-sum game of imperfect information using gradient ascent, in the infinitesimal time-step limit. The learning dynamics around a mixed-strategy Nash equilibrium are oscillatory, but can be made stable using the lagging anchor algorithm. This assumes the opponent strategies are known and the gradient can be computed. In the more realistic situation in which the gradient must be inferred from observation, we show that gradient ascent is always unstable in the small time-step limit. Using lagging anchor, however, there is a phase transition in parameter space. The gradient estimate lags the actual gradient due to the need to accumulate observations. If the lag time is too great, the Nash equilibrium becomes unstable, otherwise it is stable in the mean, but stochasticity introduces damped oscillations.

8. REFERENCES

[1] Moshe Bitterman. *The Noisy Oscillator*. World Scientific, 2005.

[2] Michael H. Bowling and Manuela M. Veloso. Multiagent learning using a variable learning rate. *Artificial Intelligence*, 136(2):215–250, 2002.

[3] Vincent Conitzer and Tuomas Sandholm. Awesome: A general multiagent learning algorithm that converges in self-play and learns a best response against stationary opponents. *Machine Learning*, 67(1-2):23–43, 2007.

[4] Vincent P. Crawford. Learning the optimal strategy in a zero-sum game. *Econometrica*, 42(5):885–891, 1974.

[5] Vincent P. Crawford. Learning behavior and mixed-strategy nash equilibria. *Journal of Economic Behavior and Organization*, 6(1):69–78, 1985.

[6] Vincent P. Crawford. Learning and mixed-strategy equilibria in evolutionary games. *Journal of Theoretical Biology*, 140:537–550, 1989.

[7] Fredrik A. Dahl. The lagging anchor algorithm: Reinforcement learning in two-player zero-sum games with imperfect information. *Machine Learning*, 49(1):5–37, 2002.

[8] Sevan G. Ficici and Jordan B. Pollack. Challenges in coevolutionary learning: Arms-race dynamics, open-endedness, and mediocre stable states. In Christopher Adami, editor, *Proceedings of the Sixth International Conference on Artificial Life*, pages 238–247. MIT Press, 1998.

[9] Drew Fudenberg and David K. Levine. *The Theory of Learning in Games*. MIT Press, 1998.

[10] Josef Hofbauer and Karl Sigmund. *Evolutionary Games and Population Dynamics*. Cambridge University Press, 1998.

[11] H. W. Kuhn. A simplified two-person poker. In *Contributions to the Theory of Games*, volume 1, pages 97–103. Princeton University Press, 1950.

[12] John F. Nash. Equilibrium points in n-person games. In *Proceedings of the National Academy of Sciences*, volume 36, pages 48–49, 1950.

[13] Jordan B. Pollack and Alan D. Blair. Why did TD-Gammon work? In *Advances in Neural Information Processing Systems 9*. MIT Press, 1996.

[14] Mitchell A. Potter and Kenneth A. De Jong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8:1–29, 2000.

[15] H. Robbins and S. Munro. A stochastic approximation method. *Ann. Math. Stat.*, 1951.

[16] David Salamon and Peter Salamon. Cycling co-evolution resulting from genetic adaptation in two-person zero-sum games. *Open Systems & Information Dynamics*, 12, Number 3 / September, 2005(3):265–271, 2005.

[17] Yuzuru Sato and James Crutchfield. Coupled replicator equations for the dynamics of learning in multiagent systems. *Physical Review E*, 67(1):015206(4), 2003.

[18] Reinhard Selten. Anticipatory learning in games. In *Game Equilibrium Models*, volume 1, New York, 1991. Springer-Verlag.

[19] J. L. Shapiro. Does data-model co-evolution improve generalization performance of evolving learners? In *LNCS*, volume 1498, pages 540–549, 1998.

[20] Satinder Singh, Michael Kearns, and Yishay Mansour. Nash convergence of gradient dynamics in general-sum games. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 541–548. Morgan Kaufmann, 2000.

[21] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[22] G. J. Tesauro. TD-gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computation*, 6(2):215–219, 1994.

[23] Ronald J. Williams. Simle statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 1992.

[24] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *International Conference on Machine Learning (ICML)*, pages 928–936, Washington, DC, USA, 2003.