Multi-agent Learning and the Reinforcement Gradient

Michael Kaisers, Daan Bloembergen, and Karl Tuyls

Department of Knowledge Engineering, Maastricht University

Abstract. This article shows that seemingly diverse implementations of multi-agent reinforcement learning share the same basic building block in their learning dynamics: a mathematical term that is closely related to the gradient of the expected reward. Gradient Ascent on the expected reward has been used to derive strong convergence results in two-player two-action games, at the expense of strong assumptions such as full information on the game that is being played. Variations of Gradient Ascent, such as *Infinitesimal Gradient Ascent* (IGA), *Win-or-Learn-Fast* IGA, and *Weighted Policy Learning* (WPL), assume a known value function for which the reinforcement gradient can be computed directly. In contrast, independent multi-agent reinforcement learning algorithms that assume less information on the game being played such as *Cross learning*, variations of *Q-learning* and *Regret minimization* base their learning on feedback from discrete interactions with the environment, requiring neither an explicit representation of the value function nor its gradient. Despite this much stricter limitation on information available to these algorithms, they yield dynamics which are very similar to Gradient Ascent and exhibit equivalent convergence behavior. In addition to the formal derivation, directional field plots of the learning dynamics in representative classes of two-player two-action games illustrate the similarities and strengthen the theoretical findings.

1 Introduction

Recent multi-agent learning survey papers and publications at agents and machine learning conferences make clear that the number of newly proposed multi-agent learning algorithms is constantly growing. Many domain-specific problems are tackled by modifying or refining the learning algorithms in question for the task at hand. An overview of well-established multi-agent learning algorithms with their various purposes is given in [5]; it demonstrates the need for a comprehensive understanding of their similarities and differences. The diversity of learning algorithms makes it imperative to specify the assumptions (*learning bias*) which precede any discussion [6]. Within the scope of this article, agents can only observe their own actions and payoffs (i.e., learning with *Minimal Information*).

Q-learning has been linked to a dynamical system which allows decomposing the learning dynamics into exploitation and exploration terms [20]. This reveals that the exploitation terms, which move the behavior toward higher payoff, are

M. Cossentino et al. (Eds.): EUMAS 2011, LNAI 7541, pp. 145–159, 2012.

equivalent to Cross Learning as described in [3]. The remaining terms allow for inferior strategies to be explored. This article extends this decomposition, and further relates Cross Learning to the most fundamental concept of learning to increase payoff – learning along the reinforcement gradient.

Specifically, two independent branches of multi-agent learning research can be distinguished based on their respective assumptions and premises. The first branch assumes that the value function of the game is known to all players, which is then used to update the learning policy based on *Gradient Ascent*. Notable algorithms in this branch include Infinitesimal Gradient Ascent (IGA) [17], the variation Win or Learn Fast IGA (WoLF) [4] and the Weighted Policy Learner [1]. The second branch of multi-agent learning is concerned with learning in unknown environments, using interaction-based *Reinforcement Learning*, and contains those algorithms which have been shown to be formally connected to the replicator equations of *Evolutionary Game Theory*. In this case, the learning agent updates its policy based on a sequence of $\langle action, reward \rangle$ pairs that indicate the quality of the actions taken. Notable algorithms include Cross Learning [7], Regret Minimization [13], and variations of Q-learning [11,21]. This article demonstrates inherent similarities between these diverse families of algorithms by comparing their underlying learning dynamics, derived as the continuous time limit of their policy updates. These dynamics have already been investigated for algorithms from each family separately [1,3,4,11,13,17], however, they have not yet been discussed in context of the relation to each other, and the origin of their similarity has not been discussed satisfactorily.

The remainder of this article is structured as follows: Section 2 formally introduces Gradient Ascent, Reinforcement Learning, and the concepts of Evolutionary Game Theory that are used in the analysis. This analysis is presented in Section 3 and starts with a comparison of the evolutionary dynamics of reinforcement learning. Representative two-player two-action games serve as an illustrative example for the comparison of these dynamics to gradient ascent. This comparison is then generalized to two-player normal form games. Section 4 emphasizes the practical differences between reinforcement learning and gradient ascent, and sketches the merits of evolutionary game theory. Finally, Section 5 concludes the article.

2 Background

This section provides an overview of the basic concepts of Gradient Ascent, Reinforcement Learning, and Evolutionary Game Theory, that are necessary to understand the remainder of this paper. This overview is not meant to be complete; references are provided for those readers that want to delve deeper into these diverse fields.

2.1 Gradient Ascent

The idea of gradient ascent (or decent) is a well known optimization technique in the field of Machine Learning. If an appropriate objective function can be

defined, the learning process can be directed in the direction of its gradient in order to find an optimum. This concept can be adapted for multi-agent learning by having the learning agents' policies follow the gradient of their expected payoff. Naturally, this approach assumes that the expected payoff function is known to the learners, which is not generally feasible in practice.

One algorithm that implements gradient ascent is **Infinitesimal Gradient Ascent** (IGA) [17], in which a learner updates its policy by taking infinitesimal steps in the direction of the gradient of its expected payoff. It has been proven that in two-player two-action games IGA either converges to a Nash equilibrium or the asymptotic expected payoff of the two players converges to the expected payoff of a Nash equilibrium. A discrete time algorithms using a finite decreasing step size is shown to share these properties as well.

A learner's policy $\pi(t) = \{\pi_1, \pi_2, \ldots, \pi_n\}$ denotes a probability distribution over its n possible actions at time t, where π_i is the probability of selecting action i, i.e., $\forall i : 0 \leq \pi_i \leq 1$, and $\sum_i \pi_i = 1$. Take $V(\pi) : \mathbb{R}^n \to \mathbb{R}$ to be the value function that maps a policy to its expected payoff. The policy update rule for IGA can now be defined as

$$\begin{aligned} \Delta\pi_i(t) &\leftarrow \alpha \frac{\partial V(\pi(t))}{\partial \pi_i(t)} \\ \pi_i(t+1) &\leftarrow projection(\pi_i(t) + \Delta\pi_i(t)) \end{aligned} \tag{1}$$

where α denotes the learning step size. The intended change $\Delta\pi(t)$ may take π outside of the valid policy space, in which case it is projected back to the nearest valid policy by the *projection* function.

Win or Learn Fast. (WoLF) [4] is a variation on IGA which uses a variable learning rate. The intuition behind this scheme is that an agent should adapt quickly if it is performing worse than expected, whereas it should be more cautious when it is winning. The modified learning rule of IGA-WoLF is

$$\begin{aligned} \Delta\pi_i(t) &\leftarrow \frac{\partial V(\pi(t))}{\partial \pi_i(t)} \begin{cases} \alpha_{min} \text{ if } V(\pi(t)) > V(\pi^*) \\ \alpha_{max} \text{ otherwise} \end{cases} \\ \pi_i(t+1) &\leftarrow projection(\pi_i(t) + \Delta\pi_i(t)) \end{aligned} \tag{2}$$

where π^* is an arbitrary Nash equilibrium policy. This means that, next to the assumption that the value function is known to all agents, WoLF also assumes that the agents have knowledge of at least one strategy which is part of a Nash equilibrium of the learning problem.

The **Weighted Policy Learner** (WPL) [1] is a second variation of IGA that also modulates the learning rate, but in contrast to WoLF-IGA does not require knowledge of Nash equilibria. The update rule of WPL is defined as

$$\begin{aligned} \Delta\pi_i(t) &\leftarrow \alpha \frac{\partial V(\pi(t))}{\partial \pi_i(t)} \begin{cases} \pi_i(t) & \text{if } \frac{\partial V(\pi(t))}{\partial \pi_i(t)} < 0 \\ 1 - \pi_i(t) & \text{otherwise} \end{cases} \\ \pi_i(t+1) &\leftarrow projection(\pi_i(t) + \Delta\pi_i(t)) \end{aligned} \tag{3}$$

where the update is weighted either by π_i or by $1 - \pi_i$ depending on the sign of the gradient. This means that π is driven away from the boundaries of the policy space. The *projection* function is slightly different from IGA, in that the policy is projected to the closest valid policy that lies at distance ϵ within the boundary of the policy space. Note that while WPL does not require the Nash equilibria to be known, it still requires that the gradient of the value function is known.

2.2 Reinforcement Learning

Reinforcement Learning starts from a different premise than gradient ascent. Instead of assuming full knowledge of the value function, a reinforcement learning agent learns from scratch by repeatedly interacting with its environment. After taking an action, the agent perceives the resulting state of the environment and receives a reward that captures the desirability of that state and the cost of the action. While the single-agent reinforcement learning problem is well defined as a Markov decision process, the multi-agent case is more complex. As state transitions and rewards are influenced by the joint action of all agents, the Markov property is no longer satisfied from a single agents' point of view. In essence, each agent is chasing its optimal policy, which depends on what the other agents do – and since they change as well, all agents chase a moving target. Nevertheless, single-agent reinforcement learning algorithms have been shown to produce good results in the multi-agent case [5]. Three independent reinforcement algorithms are selected for this article: the policy iterator Cross Learning; and the value iterators Regret Minimization and Q-learning.

This article considers the special case of stateless reinforcement learning, which facilitates the analysis of the algorithms and enables natural comparison to the similarly stateless gradient ascent algorithms. One of the most basic reinforcement learning algorithms is **Cross Learning** [3,7], which updates its policy π based on the reward r received after taking action j:

$$\pi_i(t+1) \leftarrow \begin{cases} r(t) + [1 - r(t)]\,\pi_i(t) & \text{if } i = j \\ [1 - r(t)]\,\pi_i(t) & \text{otherwise} \end{cases} \tag{4}$$

In this case, no projection function is needed as a valid policy is ensured by the update rule as long as the rewards are normalized, i.e., $0 \leq r \leq 1$. Cross Learning is closely related to Finite Action-set Learning Automata (FALA) [15,19]. In particular, it is equivalent to a learning automaton with a linear reward-inaction (L_{R-I}) scheme and a learning step size of 1.

The notion of **Regret Minimization** (RM) forms the basis for a different type of reinforcement learning algorithms. In the Polynomial Weights algorithm [2], the learner calculates the loss l_i of taking action i rather than the best action in hindsight as $l_i = r^* - r$ where r is the reward received, and r^* is the optimal reward. The learner maintains a set of weights w for its actions, updates these weights according to the perceived loss, and derives a new policy by normalization:

$$\begin{pmatrix} A_{11}, B_{11} & A_{12}, B_{12} \\ A_{21}, B_{21} & A_{22}, B_{22} \end{pmatrix}$$

Fig. 1. General payoff bi-matrix (A, B) for two-player two-action games

$$\begin{aligned} w_i(t+1) &\leftarrow w_i(t)\left[1 - \alpha l_i(t)\right] \\ \pi_i &= \frac{w_i}{\sum_j w_j} \end{aligned} \tag{5}$$

Like Cross Learning, this algorithm ensures a valid policy as long as the rewards are normalized. There is however a difference in information requirements: Regret Minimization requires to know the optimal reward in hindsight.

Arguably the best-known reinforcement learning algorithm is **Q-learning**. Q-learning estimates the expected discounted future reward achievable for every action from the learner's current state [21]. In the stateless learning problem, learning the expected discounted future reward is equivalent to learning the expected instantaneous reward. The action-value function Q is updated at each step according to

$$Q_i(t+1) \leftarrow Q_i(t) + \alpha\left[r(t) - Q_i(t)\right] \tag{6}$$

after which a new policy can be derived. Various schemes exist to derive the policy, which mainly differ in the way they balance exploration and exploitation. The Boltzmann scheme [18] allows controlling this balance using a temperature parameter τ:

$$\pi_i = \frac{e^{Q_i \cdot \tau^{-1}}}{\sum_j e^{Q_j \cdot \tau^{-1}}} \tag{7}$$

A high temperature promotes exploration, whereas a low temperature favors exploitation and generates a close-to-greedy policy. Q-learning using the Boltzmann scheme ensures a valid policy independent of the reward range, and does not require the reward function to be known.

2.3 Evolutionary Game Theory

Game theory models strategic interactions in the form of games. Each player has a set of actions, and a preference over the joint action space which is captured in the numerical payoff signal. For two-player games, the payoffs can be represented by a bi-matrix (A, B), that gives the payoff for the row player in A, and the column player in B (see Figure 1). In this example, the row player chooses one of the two rows, the column player chooses on of the columns, and the outcome of this joint action determines the payoff to both. The goal for each player is to come up with a strategy (a probability distribution over its actions) that maximizes its expected payoff in the game.

It is assumed that the players are hyper-rational, in the sense that each player purely tries to maximize its own payoff, and assumes the others are doing likewise. Under this assumption, the Nash equilibrium prescribes what players will

reasonably choose to do. A set of strategies forms a Nash equilibrium if no single player can do better by unilaterally switching to a different strategy [8]. In other words, each strategy in a Nash equilibrium is a best response against all other strategies in that Nash equilibrium.

Classical game theory requires that full information about the game is available to each player, which together with the assumption of hyper-rationality does not reflect the dynamical nature of most real world environments [9]. Evolutionary Game Theory (EGT) replaces the assumption of rationality by concepts like natural selection and mutation from evolutionary biology [14]. Since EGT relies on relaxed assumptions, it provides a solid basis to study the decision making process of bounded rational players in an uncertain environment.

Central to evolutionary game theory are the replicator dynamics, that describe how a population of candidate strategies evolves over time. Suppose that each player is represented by a population consisting of pure strategies. The fact that a player plays action i with probability π_i can then be translated as a fraction π_i of the population playing pure strategy i. In this evolutionary setting, the fitness of each candidate strategy is defined by its expected payoff against a randomly selected individual from the opponent's population. The reproduction rate of each strategy depends on the difference between its individual fitness and the average fitness of the whole population: if a strategy does better than average, its population share will increase; if it does worse, it will decrease.

In a two-player game with payoff bi-matrix (A, B), where the two players use the strategies π and σ respectively, the fitness of row player's i^{th} candidate strategy can be calculated as $\sum_j A_{ij}\sigma_j$. Similarly, the average fitness of the population is defined as $\sum_i \pi_i \sum_j A_{ij}\sigma_j$. In matrix form, this leads to the following dynamical system describing the change over time in the frequency distribution of the candidate strategies:

$$\begin{aligned} \dot{\pi}_i &= \pi_i \left[e_i A \sigma^T - \pi A \sigma^T \right] \\ \dot{\sigma}_i &= \sigma_i \left[\pi B e_i^T - \pi B \sigma^T \right] \end{aligned} \tag{8}$$

where e_i is the i^{th} unit vector. These differential equations are the replicator dynamics that encode the evolutionary concept of selection based on fitness.

2.4 Linking Reinforcement Learning to Evolutionary Game Theory

Multi-Agent Learning and Evolutionary Game Theory share a substantial part of their foundation, in that they both deal with the decision making process of bounded rational agents, or players, in uncertain environments. The link between these two field is not only an intuitive one, but was made formal with the proof that the continuous time limit of Cross Learning converges to the replicator dynamics [3].

Recall the update rule of Cross Learning given in Equation 4. This update rule can be rewritten to take the form $\pi_i(t+1) = \pi_i(t) + \Delta\pi_i(t)$ as

$$\pi_i(t+1) \leftarrow \pi_i(t) + \begin{cases} r(t) - r(t)\pi_i(t) & \text{if } i = j \\ -r(t)\pi_i(t) & \text{otherwise} \end{cases}$$

Note that the probability π_i of action i is affected both if i is selected and if another action j is selected, and let r_i or r_j be the reward received for taking action i or j respectively. The expected change $\mathrm{E}[\Delta\pi_i(t)]$ can now be calculated as

$$E[\Delta\pi_i(t)] = \pi_i(t)[r_i(t) - r_i(t)\pi_i(t)] + \sum_{j\neq i} \pi_j[-r_j(t)\pi_i(t)]$$

$$= \pi_i(t)\left[r_i(t) - \sum_j \pi_j(t) r_j(t)\right]$$

If the discrete updates are considered learning steps of time 1, then the continuous limit of this process can be taken as $\pi_i(t+\delta) = \pi_i(t) + \delta\Delta\pi_i(t)$, with $\lim \delta \to 0$. This yields a continuous system which can be expressed with a partial differential equation as

$$\dot{\pi}_i = \pi_i\left[f_i - \sum_j x_j f_j\right] \text{ where } f_i = \mathrm{E}[r_i]$$

which is the single-population replicator dynamic. In a two-agent scenario with rewards defined by the payoff bi-matrix (A, B) and agents following policies π and σ, the multi-population replicator dynamics of Equation 8 turn up again. This can be seen by replacing the expected reward $f_i = e_i A\sigma$ for the first agent.

3 Analysis

This section presents an overview of the dynamics of the different algorithms, and highlights their similarities. The discussion is limited to the domain of two-player normal form games for the sake of clarity. First, the evolutionary game theoretic models that have been derived for Cross Learning, Frequency Adjusted Q-learning and Regret Minimization are described and compared. Next, the similarities between these evolutionary dynamics and the gradient ascent algorithms are derived for two-player two-action games. In addition, the various dynamics are visualized in directional field plots. Finally, these findings are generalized to two-player normal-form games.

3.1 Evolutionary Dynamics of Reinforcement Learning

Cross Learning (CL) was the first algorithm to be linked to a dynamical system from evolutionary game theory [3]. As described in Section 2.4, the learning dynamics of CL in the limit of an infinitesimal update step approach the replicator dynamics of Equation 8, which is reiterated here for one player:

$$\dot{\pi}_i = \pi_i\left[e_i A\sigma^T - \pi A\sigma^T\right]$$

The link between a policy learner like CL and a dynamical system in the policy space may be rather straight forward. However, the link has been extended to

value based learners as well. A model of Q-learning with the Boltzman update scheme has been proposed in [20], given the additional assumption of updating all actions simultaneously. The variation **Frequency-Adjusted Q-learning** (FAQ-learning) [11] implements this model by modulating the update rule inversely proportional to π_i, thereby approximating simultaneous action updates.

$$Q_i(t+1) \leftarrow Q_i(t) + \frac{1}{\pi_i}\alpha\left[r_i(t) - Q_i(t)\right]$$

This update rule corresponds to a dynamical system that can be decomposed into the replicator dynamics and terms for randomization, where the temperature parameter τ tunes the balance between the two. The replicator dynamics enforce the selection of knowingly better actions (exploitation), while randomization corresponds to mutations (exploration). In brief, the FAQ-learning dynamics are a weighted average between CL and exploration.

$$\dot{\pi}_i = \frac{\alpha\pi_i}{\tau}\underbrace{\left[e_i A\sigma^T - \pi A\sigma^T\right]}_{\text{selection}} - \alpha\pi_i\underbrace{\left[\log\pi_i - \sum_k \pi_k \log\pi_k\right]}_{\text{mutation}}$$

Recently, the evolutionary framework has also been extended to the Polynomial Weights algorithm, which implements **Regret Minimization** [2,13]. Despite the great difference in update rule and policy generation (see Eq. 5), the infinitesimal limit has been linked to a dynamical system with CL dynamics in the nominator.

$$\dot{\pi}_i = \frac{\alpha\pi_i\left[e_i A\sigma^T - \pi A\sigma^T\right]}{1 - \alpha\left[\max_k e_k A\sigma^T - \pi A\sigma^T\right]}$$

The denominator can be interpreted as a learning rate modulation dependent on the best action's relative fitness.

3.2 Similarities in Two-Player Two-Action Games

For two-agent two-action games, the dynamics can be simplified. Let $h = (1, -1)$, $\pi = (x, 1-x)$ and $\sigma = (y, 1-y)$. The dynamics are completely described by the pair $(\dot{x}, \dot{y})$, which denote the probability changes of the first actions. For CL in self-play, this leads to the following simplified form:

$$\dot{x} = x(1-x)\left[yhAh^T + A_{12} - A_{22}\right]$$
$$\dot{y} = y(1-y)\left[xhBh^T + B_{21} - B_{22}\right]$$

Here, the payoff matrices A and B are again taken from the bi-matrix given in Figure 1. The second player's update $\dot{y}$ is completely analogous to $\dot{x}$, and will be omitted in the subsequent discussion. Similarly, for FAQ-learning the simplified dynamics read

$$\dot{x} = \alpha x(1-x)\left(\tau^{-1}\left[yhAh^T + A_{12} - A_{22}\right] - \log\frac{x}{1-x}\right)$$

The dynamics of RM are slightly more complex. To simplify the notation for two-action games, let $\eth = e_1 A\sigma^T - e_2 A\sigma^T = yhAh^T + A_{12} - A_{22}$. The numerator is equal to Cross Learning with an additional step size parameter α, and the denominator depends on which action gives the highest reward. This can be derived from the gradient: the first action will be maximal iff $\eth > 0$. If the first action is maximal, the denominator can be worked out to read $1 - \alpha(1-x)\eth$. Similarly, when the second action is maximal the denominator reads $1+\alpha\eth$. The dynamics of RM in two action games can then be written as follows:

$$\dot{x} = \alpha x(1-x)\eth \cdot \begin{cases} (1+\alpha x\eth)^{-1} & \text{if } \eth < 0 \\ (1-\alpha(1-x)\eth)^{-1} & \text{otherwise} \end{cases}$$

For **Gradient Ascent**, the update rule can be worked out in a similar fashion. The main term in this update rule is the gradient of the expected reward, which in two player two-action games can be written as

$$\begin{aligned} \frac{\partial V(x,y)}{\partial x} &= \frac{\partial}{\partial x}(x, 1-x) A \begin{pmatrix} y \\ 1-y \end{pmatrix} \\ &= y(A_{11} - A_{12} - A_{21} + A_{22}) + A_{12} - A_{22} \\ &= yhAh^T + A_{12} - A_{22} \\ &= \eth \end{aligned}$$

This reduces the dynamics of the update rule for IGA in two-player two-action games to

$$\dot{x} = \alpha\eth$$

The extension of the dynamics of IGA to IGA-WoLF and WPL are straightforward. Table 1 lists the dynamics of the six discussed algorithms: IGA, WoLF, WPL, CL, FAQ and RM. It is immediately clear from this table that *all algorithms have the same basic term in their dynamics*: the gradient $\eth$. Depending on the algorithm, the gradient is scaled with a learning speed modulation. FAQ-learning yields the only dynamics that additionally add exploration terms to the process.

In order to illustrate the similarities between the algorithms, their dynamics are visualized in representative classes of two-player two-action games. Three distinct classes can be identified [9]. The first class consists of games with one pure Nash equilibrium, such as the Prisoner's Dilemma. The second class of games has two pure and one mixed Nash equilibrium, such as the Battle of the Sexes. Finally, the third class of games has only one mixed Nash equilibrium; an example is the Matching Pennies game. The payoff bi-matrices of these games are presented in Figure 2.

Since the dynamics in two-player two-action games are completely described by the pair $(\dot{x}, \dot{y})$ as described above, it is possible to plot the dynamics in the unit square that makes up the joint policy space. The dynamics can be visualized using a directional field plot, where each arrow indicates the direction of change at that point (x, y) in the policy space.

Table 1. This table shows an overview of the learning dynamics, rewritten for the specific case of two-agent two-action games. For simplicity, the common gradient is abbreviated $\eth = \left[yhAh^T + A_{12} - A_{22}\right]$.

Algorithm	$\dot{x}$
IGA	$\alpha\eth$
WoLF	$\eth \cdot \begin{cases} \alpha_{min} & \text{if } V(x,y) > V(x^e,y) \\ \alpha_{max} & \text{otherwise} \end{cases}$
WPL	$\alpha\eth \cdot \begin{cases} x & \text{if } \eth \text{ ¡ } 0 \\ (1-x) & \text{otherwise} \end{cases}$
CL	$\alpha x(1-x)\ \eth$
FAQ	$\alpha x(1-x)\left[\eth \cdot \tau^{-1} - \log \frac{x}{1-x}\right]$
RM	$\alpha x(1-x)\ \eth \cdot \begin{cases} (1+\alpha x\eth)^{-1} & \text{if } \eth < 0 \\ (1-\alpha(1-x)\eth)^{-1} & \text{otherwise} \end{cases}$

$$\begin{array}{c} \\ C \\ D \end{array}\!\!\begin{array}{c} \begin{array}{cc} C & D \end{array} \\ \begin{pmatrix} \frac{3}{5},\frac{3}{5} & 0,1 \\ 1,0 & \frac{1}{5},\frac{1}{5} \end{pmatrix} \end{array} \qquad \begin{array}{c} \\ O \\ F \end{array}\!\!\begin{array}{c} \begin{array}{cc} O & F \end{array} \\ \begin{pmatrix} 1,\frac{1}{2} & 0,0 \\ 0,0 & \frac{1}{2},1 \end{pmatrix} \end{array} \qquad \begin{array}{c} \\ H \\ T \end{array}\!\!\begin{array}{c} \begin{array}{cc} H & T \end{array} \\ \begin{pmatrix} 1,0 & 0,1 \\ 0,1 & 1,0 \end{pmatrix} \end{array}$$

Prisoner's Dilemma Battle of the Sexes Matching Pennies

Fig. 2. Normalized payoff matrices for three representative two-player two-action games

Figure 3 shows the learning dynamics of the different algorithms in the Prisoner's Dilemma, Battle of the Sexes, and Matching Pennies game. The dynamics of RM are not shown, as they are visually indistinguishable from CL. Figure 3 illustrates the high similarity between all algorithms in the first two games. They all share the same convergence properties, and follow similar trajectories. The dynamics of IGA and WoLF in the Prisoners' Dilemma show the need for the *projection* function to prevent the update from taking the policies π and σ out of the valid policy space.

In the Matching Pennies, IGA and CL both cycle around the Nash equilibrium. The other three algorithms all spiral inwards and eventually converge, but do so in a different manner. The dynamics of WoLF clearly show the effect of the *win or learn fast* scheme, switching between the two discrete learning step values at $x = 0.5$ and $y = 0.5$.

3.3 Generalization to Two-Player Normal Form Games

The previous section has show the gradient $\eth$ to be a basic building block of several learning algorithms in two-action games, where the policy can be written as $\pi = \{x, 1-x\}$. This section extends the definition to the more general case of two-player normal form games, where each player has a finite discrete set of actions, and $\pi = \{\pi_1, \pi_2, \ldots, \pi_n\}$ such that $\sum_i \pi_i = 1$ and $\forall \pi_i : 0 \leq \pi_i \leq 1$. To ensure satisfying the first constraint the gradient needs to be normalized such that $\sum_i \eth_i = 0$, where $\eth_i$ is the i^{th} component of the gradient, i.e., $\eth_i$ is the

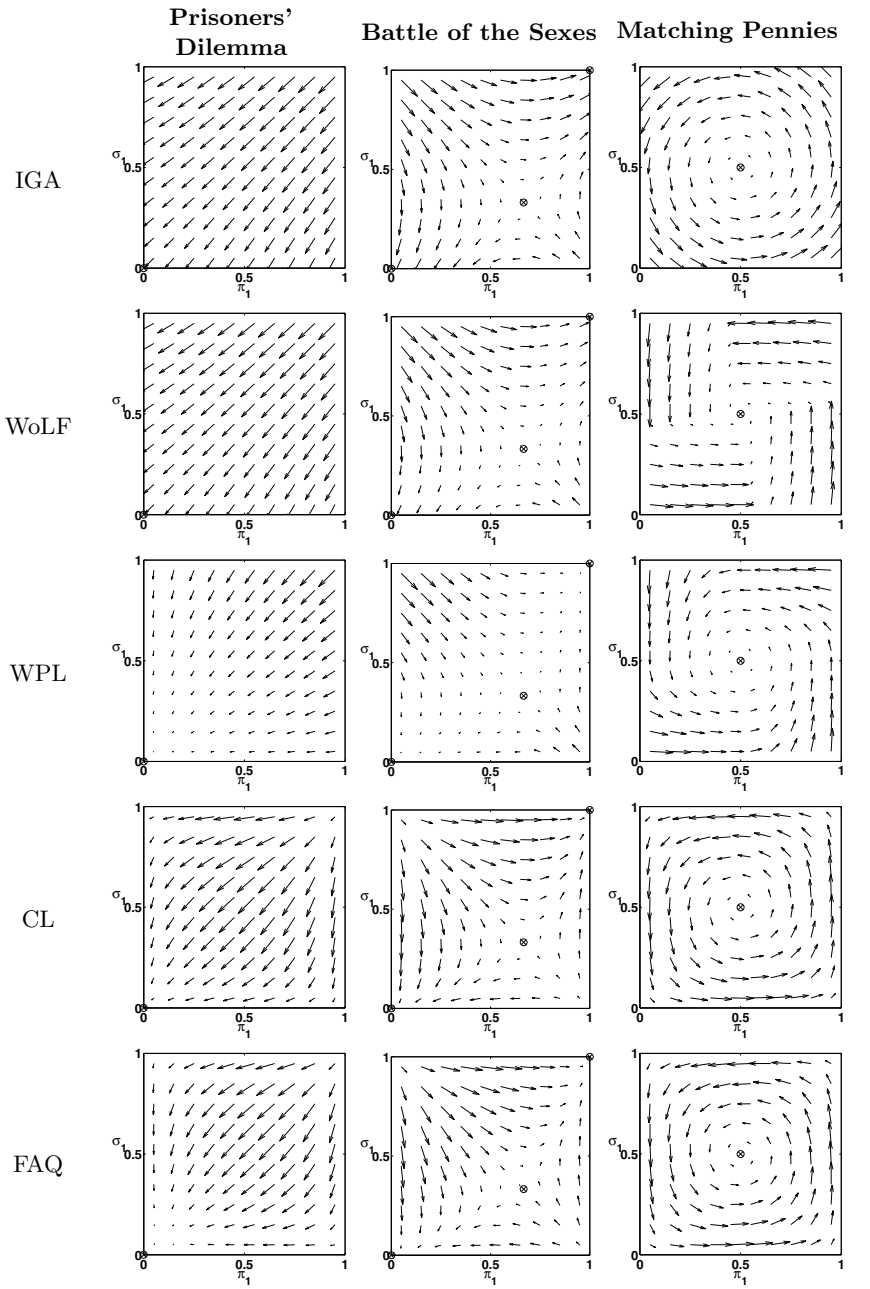

Fig. 3. This figure shows the learning dynamics of the various algorithms in the Prisoners' Dilemma, Battle of the Sexes, and Matching Pennies. The Nash Equilibria are indicated with ⊗.

partial derivative of the value function with respect to π_i. The second constraint is satisfied either by projecting π to the (closest point in the) valid policy space, or by making $\dot{\pi}_i$ itself dependent on π, analogous to the different policy update rules of the algorithms.

Recall that the value function in two-player normal form games is defined as $V(\pi, \sigma) = \pi A \sigma^T$. The i^{th} element of the gradient can be calculated as the partial derivative of V with respect to π_i. Let e_i denote the i^{th} unit vector, the differential with respect to π_i can then be defined as δe_i. However, recall that $\dot{\pi}$ needs to be normalized in order to stay on the tangent space of π. This can be guaranteed by projecting δe_i onto the tangent space using the orthogonal projection function $\Phi(\zeta) = \zeta - \frac{1}{n}\sum_j \zeta_j$ [16]. This gradient can now be written as

$$\begin{aligned}\frac{\partial V(\pi,\sigma)}{\partial \pi_i} &= \lim_{\delta \to 0} \frac{[\pi + \Phi(\delta e_i)]\, A\sigma^T - \pi A \sigma^T}{\delta} \\ &= \Phi(e_i) A \sigma^T \\ &= e_i A \sigma^T - \frac{1}{n}\sum_j e_j A \sigma^T\end{aligned}$$

which closely resembles the replicator dynamics (see Equation 8). As explained in Section 2, IGA and WoLF use a projection function to ensure a valid policy (i.e., to satisfy $\forall \pi_i : 0 \leq \pi_i \leq 1$). Similarly, their dynamical models need to be projected back to the valid policy space. CL, FAQ and RM take another approach, and ensure validity of the policy update by making the update rule proportional to π. Incorporating proportional updating into the gradient-based policy update rule yields

$$\pi_i(t+1) \leftarrow \pi_i(t) + \pi_i \frac{\partial V(\pi,\sigma)}{\partial \pi_i}$$

The projection function Φ which projects δe_i to the tangent space of π needs to change as well in order to properly map the weighted gradient. Intuitively, this can be achieved by using a weighted mean instead of a standard mean, such that $\hat{\Phi}(\zeta, w) = \zeta - \sum_j w_j \zeta_j$ where w is a normalized weight vector. Using $w = \pi$, this leads to the following dynamics:

$$\begin{aligned}\dot{\pi}_i &= \pi_i \lim_{\delta \to 0} \frac{\left[\pi + \hat{\Phi}(\delta e_i, \pi)\right] A\sigma^T - \pi A \sigma^T}{\delta} \\ &= \pi_i \lim_{\delta \to 0} \frac{\pi A \sigma^T + \hat{\Phi}(\delta e_i, \pi) A \sigma^T - \pi A \sigma^T}{\delta} \\ &= \pi_i \hat{\Phi}(e_i, \pi) A \sigma^T \\ &= \pi_i [e_i A \sigma^T - \sum_j \pi_j e_j A \sigma^T] \\ &= \pi_i [e_i A \sigma^T - \pi A \sigma^T]\end{aligned}$$

These resulting dynamics are exactly the replicator dynamics of Equation 8, which shows that Cross Learning is equal to Gradient Ascent with proportional

updates. This provides a strong link between the two families of algorithms, gradient ascent on the one hand and independent multi-agent reinforcement learning on the other.

All of the algorithms described in this section reveal dynamics that follow the reinforcement gradient. The terms of the gradient appear to be the foundation of multi-agent reinforcement learning, with learning rate modulations and some deviations for the sake of exploration and coordination.

4 Discussion

Gradient Ascent on the expected reinforcement assumes that the gradient is known or can be computed by the agent. This is typically not the case in reinforcement learning problems. The merits of Gradient Ascent are more theoretical – it allows to provide convergence guarantees at the cost of stronger assumptions. Recently, similar guarantees have also been derived for evolutionary models of independent multi-agent reinforcement learning. These guarantees either draw on well established models from evolutionary biology, or study newly derived variations. For example, the cyclic behavior of the replicator dynamics is a well studied phenomenon [10], while the dynamics of FAQ-learning have been thoroughly analyzed in two-agent two-action games showing convergence to Nash equilibria [12]. In addition, the findings presented in this article highlight the commonalities of gradient ascent and reinforcement learning. Future research can build on this basis and further unite the two parallel streams of literature.

5 Conclusions

This article relates two seemingly diverse families of algorithms within the field multi-agent learning: gradient ascent and independent reinforcement learning. The main contributions can be summarized as follows: First, it is shown that the replicator dynamics are a prime building block of various types of independent reinforcement learning algorithms, such as Cross Learning, Regret Minimization, and Q-learning. Second, the replicator dynamics are shown to relate to the gradient of the expected reward, which forms the basis of Gradient Ascent. Both the replicator dynamics and gradient ascent base their update on the difference between the expected reward of an action and the average expected reward over all actions. The difference lies in the weight given to each action's update: gradient ascent assumes uniform weights as given by the gradient, whereas the replicator dynamics use the action-selection probabilities as weights. The theoretical comparison is complimented by a visualization of the different learning dynamics in representative two-agent two-action games – a class in which their similarity is particularly compelling.

In sum, this article structures a highly diversified field such as multi-agent learning. The number of proposed learning algorithms is continuously increasing, and we deem recognizing persistent principles such as learning along the

reinforcement gradient crucial to the integrity of the field. This approach provides the basis for an analysis of the inherent capabilities but also limitations of what is learnable with independent reinforcement learning in multi-agent games. Eventually, we are seeking to establish lower bounds on the performance in multi-agent games similar to Probably Approximately Correct Learning guarantees in single-agent learning.

References

1. Abdallah, S., Lesser, V.: A multiagent reinforcement learning algorithm with non-linear dynamics. Journal of Artificial Intelligence Research 33(1), 521–549 (2008)
2. Blum, A., Mansour, Y.: Learning, regret minimization and equilibria. Cambridge University Press (2007)
3. Börgers, T., Sarin, R.: Learning through reinforcement and replicator dynamics. Journal of Economic Theory 77(1) (November 1997)
4. Bowling, M., Veloso, M.: Multiagent learning using a variable learning rate. Artificial Intelligence 136, 215–250 (2002)
5. Busoniu, L., Babuska, R., De Schutter, B.: A comprehensive survey of multiagent reinforcement learning. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 38(2), 156–172 (2008)
6. Crandall, J.W., Ahmed, A., Goodrich, M.A.: Learning in repeated games with minimal information: The effects of learning bias. In: Twenty-Fifth AAAI Conference on Artificial Intelligence (2011)
7. Cross, J.G.: A stochastic learning model of economic behavior. The Quarterly Journal of Economics 87(2), 239 (1973)
8. Gibbons, R.: A Primer in Game Theory. Pearson Education (1992)
9. Gintis, H.: Game Theory Evolving, 2nd edn. University Press, Princeton (2009)
10. Hofbauer, J., Sigmund, K.: Evolutionary Games and Population Dynamics. Cambridge University Press (2002)
11. Kaisers, M., Tuyls, K.: Frequency adjusted multi-agent Q-learning. In: Proc. of 9th Intl. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010), May 10-14, pp. 309–315 (2010)
12. Kaisers, M., Tuyls, K.: Faq-learning in matrix games: Demonstrating convergence near nash equilibria, and bifurcation of attractors in the battle of sexes. In: Proceedings of the Workshop on Interactive Decision Theory and Game Theory (2011)
13. Klos, T., van Ahee, G.J., Tuyls, K.: Evolutionary Dynamics of Regret Minimization. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) ECML PKDD 2010, Part II. LNCS, vol. 6322, pp. 82–96. Springer, Heidelberg (2010)
14. Maynard Smith, J., Price, G.R.: The logic of animal conflict. Nature 246(2), 15–18 (1973)
15. Narendra, K.S., Thathachar, M.A.L.: Learning automata - a survey. IEEE Transactions on Systems, Man, and Cybernetics 4(4), 323–334 (1974)
16. Sandholm, W.H.: Population Games and Evolutionary Dynamics. The MIT Press, Cambridge (2010)
17. Singh, S., Kearns, M., Mansour, Y.: Nash convergence of gradient dynamics in general-sum games. In: Proc. of the 16th Conference on Uncertainty in Artificial Intelligence, pp. 541–548 (2000)

18. Sutton, R., Barto, A.: Reinforcement Learning: An introduction. MIT Press, Cambridge (1998)
19. Thathachar, M.A.L., Sastry, P.S.: Varieties of learning automata: An overview. IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics 32(6), 711–722 (2002)
20. Tuyls, K., Jan't Hoen, P., Vanschoenwinkel, B.: An evolutionary dynamical analysis of multi-agent learning in iterated games. Autonomous Agents and Multi-Agent Systems 12, 115–153 (2006)
21. Watkins, C.J.C.H., Dayan, P.: Q-learning. Machine Learning 8(3), 279–292 (1992)