The Dynamics of Multi-Agent Reinforcement Learning

Luke Dickens, Krysia Broda and Alessandra Russo[1] [2]

Abstract. Infinite-horizon multi-agent control processes with non-determinism and partial state knowledge have particularly interesting properties with respect to adaptive control, such as the non-existence of Nash Equilibria (NE) or non-strict NE which are nonetheless points of convergence. The identification of reinforcement learning (RL) algorithms that are robust, accurate and efficient when applied to these general multi-agent domains is an open, challenging problem. This paper uses *learning pressure fields* as a means for evaluating RL algorithms in the context of multi-agent processes. Specifically, we show how to model partially observable infinite-horizon stochastic processes (single-agent) and games (multi-agent) within the Finite Analytic Stochastic Process framework. Taking long term average expected returns as utility measures, we show the existence of learning pressure fields: vector fields – similar to the dynamics of evolutionary game theory, which indicate medium and long term learning behaviours of agents independently seeking to maximise this utility. We show empirically that these learning pressure fields are followed closely by policy-gradient RL algorithms.

1 Introduction

Reinforcement Learning (RL) has long shown itself to be a powerful technique for optimising control of large scale control problems. Early research focused on single-agent problems with full state knowledge [10]. A particular feature of these kinds of problems are that the set of optimal control strategies (policies) always contains some members that are completely deterministic, i.e. a functional policy is sufficient for best results. Also, in many other RL applications, including those with partial-state knowledge [7] and/or cooperative multi-agent problems [2] the focus has been on building knowledge structures that allow one to consider the learning outcomes to be deterministic policies.

It has long be realised that, when state knowledge is limited, stochastic policies can outperform the best functional alternatives by an arbitrary degree, even for small single-agent problems [15]. This means that often better control can be achieved if one allows control decisions (*actions*) to be chosen non-deterministically based on current knowledge. In fact, it is widely understood that during the learning phase of on-policy RL techniques – those that interact directly with the system they aim to control – it is essential that policies have some stochastic properties, and this idea is often referred to as the exploration-exploitation trade-off [16]. Not all RL techniques rely on this assumption of the sufficiency of deterministic policies. In particular, gradient based optimisation approaches developed in the last ten years or so are able to explore the space of stochastic policies more freely [1, 3, 11, 14].

For multi-agent problems, where agents are competitive – or more generally non-cooperative, stochastic policies play an even greater role. Nash shows [13] that agents in non-cooperating games require stochastic strategies to avoid being exploited by others. He identifies points in the space of joint strategies, known as Nash Equilibria (NE), where no agent can unilaterally change strategy to improve their chances. These NE can be thought of as the multi-agent analogue of maximum utility in single-agent tasks. However, Bowling and Veloso showed that games of restricted policy can be formulated which have no NE [5]. In such systems, there is no ultimate learning outcome, instead an agent's only option is to continually adapt its (stochastic) policy to avoid being exploited by others. In these *perpetual learning* domains, it is useful to have a means to investigate the dynamic, mid-term learning behaviour of agents.

The paper addresses this by using *learning pressure fields* as a means to study the behaviour of RL algorithms in the context of infinite-horizon partially observable stochastic games. We formulate the Finite Analytic Stochastic Process (FASP) framework, and show how the single-agent partially observable markov decision process (POMDP) and partially observable stochastic games can be modelled as FASPs. Taking long term average expected returns as utility measures, we show the existence of learning pressure fields (LPFs): vector fields, which represent the medium and long term learning behaviour of agents independently seeking to maximise these measures. The LPF is related to the dynamics of evolutionary game theory [9] – in that it represents the combined gradient of adaptation for agents independently optimising within a game. This has been extended to the dynamics of RL algorithms maximising in stochastic games with full state access, see [6] and [17]. This paper extends the domain to partially observable stochastic games, and this demands that agents maximise their long term average expected return (payoff), and not immediate or geometrically discounted payoffs as would otherwise be sufficient. Section 2 formulates the FASP framework and shows that POMDPs and other stochastic games can be modelled as FASPs. Section 3 gives analytic results for FASPs, shows how to find the long term average for any real valued signal, and formalises learning pressure fields. Section 4 runs a series of experiments using four actor-critic algorithms to see whether the LPFs are followed and to evaluate how each algorithm performs. Section 5 concludes the paper with a discussion and an indication of future work.

2 Finite Analytic Stochastic Processes

Probably the most successful family of models in RL is based on the Markov Decision Process (MDP), including partially observable (PO)MDPs [7]; *cooperative* multi-agent MDP derivatives, such as

[1] Imperial College London, UK, email: {luke.dickens03,k.broda,a.russo}@imperial.ac.uk

[2] This research is continuing through participation in the International Technology Alliance sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence.

the decentralised (DEC-)POMDP [2]; and other systems under *adversarial/non-cooperative* control, such as Markov Games [12, 18]. MDP models complement the agent paradigm with the concepts of reward, state, action, and in some cases observation, and many such multi-agent MDP variants have been proposed.

While there is a great deal of variety in these models, they share many underlying properties. Here, we formalise a set of Markov Decision Process derivative frameworks called Finite Analytic Stochastic Processes (FASPs), which allow us to flexibly model single- and multi-agent stochastic processes. We show that the POMDP [7] is an example of a FASP, and present the Simultaneous Multi-Agent (SMA)FASP – a variant of the framework.

Given a FASP where agents follow reactive policies, we can predict the average expected return for any state dependent signal. Using these predictions, we can construct *learning pressure* fields (see Section 3), from which we make further inferences on how two or more optimising agents will interact, even when they are motivated by different utility signals. We first develop some concepts.

Definition 1 (Probability Distribution) *For a set X, a probability distribution over X describes a random variable, and identifies either the probability of each value of that random variable when X is discrete, or the probability of the value falling within a (measurable) subset of X, otherwise. The set of all probability distributions over X is denoted* $\mathrm{PD}(X)$. *For any $d \in \mathrm{PD}(X)$ and $x \in X$, $x \leftarrow\!\!\sim d$ implies x sampled from d; for finite X, d can be written as a vector,* $\mathbf{d} = (d_1, \ldots, d_{|X|})^{\mathsf{T}}$, *where* $d(x_i) = d_i = \Pr(x = x_i \,|x \leftarrow\!\!\sim \mathbf{d})$.

The set of all mappings from one set to a probability distribution over another set is called the stochastic map.

Definition 2 (Stochastic Map) *A stochastic map, m, is a mapping from set X to a probability distribution over a set Y, and the set of all such m is given by* $\Sigma(X \to Y) = \{m \,|m : X \to \mathrm{PD}(Y)\}$

For any measurable subset $Y' \subset Y$, we write $m(Y'|x) = \Pr(y' \in Y' \,|y' \leftarrow\!\!\sim m(x))$. If Y is finite, we can write $m(y|x) = \Pr(y = y' \,|y' \leftarrow\!\!\sim m(x))$ for any $y \in Y$.

This allows us to define our umbrella framework of MDP derivatives, supporting multiple measure (utility) signals and partial observability. Models of this kind are called Finite Analytic Stochastic Processes (FASPs).

Definition 3 (FASP) *A FASP is a tuple $(S, A, O, t, \omega, F, i, \Pi)$, where S, A and O are the finite state, action and observation spaces; $t \in \Sigma(S \times A \to S)$ is the transition function that generates new states from state-action pairs; $\omega \in \Sigma(S \to O)$ is the observation function that generates observations; $F = \{f^1, \ldots, f^N\}$ is the set of measure functions, where each $f^i \in \Sigma(S \to \mathbb{R})$ generates a real valued measure signal, f^i_n, at each time-step n; $i \in \mathrm{PD}(S)$ is the initialisation function generating initial system states; and Π is the set of all available control policies.*

We define a partially observable markov decision process (POMDP) as a subtype of the FASP.

Definition 4 (POMDP) *A POMDP is a FASP $M = (S, A, O, t, \omega, \{r\}, i, \Pi)$ with one measure function $r \in \Sigma(S \to \mathbb{R})$ called the reward function. It is controlled by one agent with actions in A, and observations in O.*

All models in this paper are *state-encapsulated*, meaning here that the observation and reward functions take the state as the only input. This contrasts with models such as the POMDP defined in [7], where rewards and observations can depend on the state and action from the previous time-step, as well as the current state. As shown in [8], state-encapsulation can allow for more concise model descriptions, particularly those with multiple non-cooperative agents, but does not limit the expressibility of the framework.

The Simultaneous Multi-Agent (SMA)FASP is defined as follows,

Definition 5 (SMAFASP) *A SMAFASP is a FASP $M = (S, A, O, t, \omega, \{r_1, \ldots, r_n\}, i, \Pi)$ and a set of agents, G ($|G| = n$). The action space A and observation space O are the Cartesian products of agent specific action and observation spaces respectively, so $A = \bigtimes_{g \in G} A^g$ and $O = \bigtimes_{g \in G} O^g$. The policy space, $\Pi = \bigtimes_{g \in G} \Pi^g$, is the Cartesian product of agent specific policy spaces, where Π^g generates agent specific actions from A^g using previous actions from A^g and observations from O^g. r_g is the reward function for agent $g \in G$.*

We now formulate two small SMAFASP examples, that will be used throughout the paper for illustration and experiments.

Example 1 (See Figure 1) *The Bowling two-step game is the SMAFASP $(S, A, O, t, \omega, \{r_v, r_h\}, i, \Pi)$, with two agents g_v and g_h, four states, $S = \{s_0, s_L, s_R, s_C\}$, the agent specific action sets $A^v = \{U, D\}$ and $A^h = \{L, R\}$ - freely available in all states, and a single joint observation, $O = \{o\}$. The transition function, $t \in \Sigma(S \times A \to S)$, is shown in the figure – arcs are labelled with the action inducing it (or an asterix representing any suitable action), and the associated probability. The observation function is trivial. The reward functions are $r_v = r_1$ and $r_h = r_2$ for agents g_v and g_h respectively, where r_1 and r_2 are defined in Table 1. The initial state is s_0. Each agent g's policy is reactive and strictly stochastic, i.e. $\Pi^g = \Sigma(O^g \to A^g) - \{\pi \,|\exists o \in O^g, \exists a \in A^g$ s.t. $\pi(o|a) = 1\}$.*

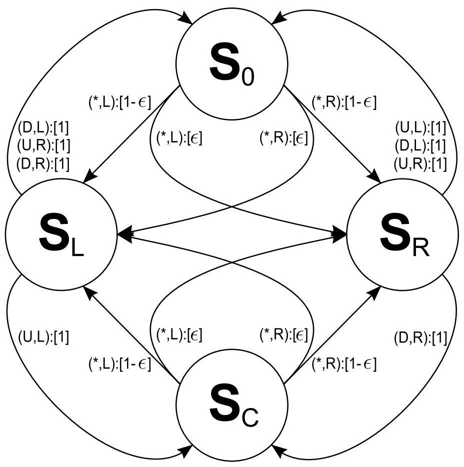

Figure 1. The Bowling two-step game as a SMAFASP.

	s_0	s_L	s_R	s_C
$r_1(.)$	0	0	0	1
$r_2(.)$	0	0	0	-1

Table 1. Reward Schemes for Bowling and Veloso's two-step game.

The Bowling two-step game is equivalent to the example described in [5], and it rewards agent g_v and punishes g_h whenever the system is in state s_C. In [5] there are three states, and the policy is restricted to be the same for every state, without using observations. The value $0 < \varepsilon \ll \frac{1}{2}$ is some small probability, and is arbitrarily set to 0.05, this choice has only a subtle effect on the resulting analysis and is not explored further.

In our second example, the *Inverted two-step game*, we swap the reward functions, r_2 for g_v and r_1 for g_h.

Example 2 (See Figure 1) *The Inverted two-step game is defined as the Bowling two-step, except reward functions $r_v = r_2$ and $r_h = r_1$.*

The Inverted two-step game rewards g_h and punishes g_v each time the system enters state s_c, and is otherwise like the Bowling two-step.

3 Analytics

We begin with a brief definition of a Markov-Process.

Definition 6 *A Markov-Process is a set of n states S and n^2 transition probabilities, where transition probability p_{ij} represents the probability that if the system is in state s_i at some time t, then the sytem will be in state s_j at time $t+1$.*

The transition probabilities p_{ij} form the elements of the transition matrix, P, with $P_{ji} = p_{ij}$; P identifies the Markov-Process.

In general, a policy for a FASP can depend on an arbitrarily rich history of observations and actions upto the present, but for the purposes of clarity we restrict ourselves to reactive policies, π, in the set $\Pi \subseteq \Sigma(O \to A)$; extensions to this are discussed in Section 5. For a reactive policy, $\pi \in \Pi$, the dynamics of this system resolves into a set of state to state transition probabilities, called the State Transition Function, and defined as follows.

Definition 7 (State Transition Function) *Given a FASP $M = (S, A, O, t, \omega, r, i, \Pi)$, with $\Pi = \{f \,|\, f : O \to \mathrm{PD}(A)\}$, the full state transition function is $\tau_M : \Pi \to \Sigma(S \to S)$, where for $\pi \in \Pi$, $\tau_M(\pi) = \tau^\pi_M$ (written as τ^π when M is clear), and, $\forall s, s' \in S$,*

$$\tau^\pi_M(s'|s) = \sum_{o\in O}\sum_{a\in A} \omega(o|s)\pi(a|o)t(s'|s,a)$$

Clearly, for a fixed $\pi \in \Pi$, the FASP behaves as a Markov Process and the state transition function gives the probabilities in the associated transition matrix, T^π, in the following way,

$$T^\pi_{ji} = \tau^\pi(s_j|s_i)$$

Our analytic solution makes use of the *ergodicity property*, defined for Markov-Process, and in turn for FASPs, as follows.

Definition 8 (ergodicity) *A* Markov-Process *with states S is step ergodic, if every state $s \in S$ is* positive recurrent *(guaranteed to be returned to in finite time), aperiodic (the greatest common divisor of all returning paths' lengths is 1), and accessible by all states $s' \neq s$ (will reach s from s' in finite time).*

A FASP *is ergodic, if for any fixed policy $\pi \in \Pi$ it resolves to an ergodic Markov-Chain.*

The occupancy for a Markov-Process or a FASP is a probability distribution, $b \in \mathrm{PD}(S)$, and represents our knowledge about the state of the system, and can be written as a vector, $\mathbf{b}$, with $\mathbf{b}^\intercal \mathbf{1} = 1$.

Lemma 1 *Given some occupancy $\mathbf{b}$ and a state dependent measure signal $r \in \Sigma(S \to \mathbb{R})$, then the expected signal from r is written as*

$$\mathbf{E}\left(r(\mathbf{b})\right) = \mathbf{E}\left(r \,|r \leftsquigarrow r(s), s \leftsquigarrow \mathbf{b}\right) = \sum_{s\in S} b(s)\mathbf{E}\left(r(s)\right)$$

A Markov-Process, P, which is ergodic, has a well defined unique stationary (asymptotic) occupancy, $\mathbf{d}$ satisfying $\mathbf{d} = P\mathbf{d}$, and the system approaches $\mathbf{d}$ at large time-steps, irrespective of the starting occupancy, i.e. $\lim_{n\to\infty} P^n \mathbf{d}_0 = \mathbf{d}, \forall \mathbf{d}_0 \in \mathrm{PD}(S)$ [15].

Clearly, for any ergodic FASP, and fixed policy $\pi \in \Pi$, there is also a policy dependent asymptotic occupancy $\mathbf{d}(\pi)$, written as $\mathbf{d}^\pi$, satisfying $\mathbf{d}^\pi = T^\pi \mathbf{d}^\pi$ and $\lim_{n\to\infty} P^n \mathbf{d}_0 = \mathbf{d}^\pi, \forall \mathbf{d}_0 \in \mathrm{PD}(S)$.

We are interested in maximising the average value for some given reward signal for an infinite trace, and, hence, define the *expected return* for the signal r given policy π as

$$J_r(\pi) = \lim_{N\to\infty} \mathbf{E}\left(\frac{1}{N}\sum_{i=0}^{N} r_i \,\middle|\, \pi\right)$$

From [8] we get the following result.

Theorem 2 (From [8]) *For any ergodic FASP, fixed $\pi \in \Pi$ and $r \in \Sigma(S \to \mathbb{R})$, the expected average signal on r over any infinite trace, approaches the occupancy return for $r(\mathbf{d}^\pi)$ w.p.1, i.e.*

$$J_r(\pi) = \mathbf{E}\left(r(\mathbf{d}^\pi)\right) = \sum_{s\in S} d^\pi(s)\mathbf{E}\left(r(s)\right)$$

A policy space, Π, is *parametrised* by $\boldsymbol{\Theta} \subseteq \mathbb{R}^l$ if there is a surjective function $\mu : \boldsymbol{\Theta} \to \Pi$, so for any $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, then $\mu(\boldsymbol{\theta}) = \pi^{\boldsymbol{\theta}} \in \Pi$. For brevity, $\pi^{\boldsymbol{\theta}}$ is written as $\boldsymbol{\theta}$. Joint policy space $\Pi = \times_{g\in G} \Pi^g$ is parametrised by $\boldsymbol{\Theta} = \times_{g\in G} \boldsymbol{\Theta}^g$ in the obvious way.

For any SMAFASP, each agent g is assumed to have a reward signal r_g, and wishes select $\boldsymbol{\theta}$ to maximise its associated expected return $J_g(\boldsymbol{\theta})$. Having identified this aim, it is natural to consider gradient ascent (policy-gradient) methods. For single-agent policy-gradient methods, as for the POMDP, this involves sampling (or estimating) the gradient of $J_g(\boldsymbol{\theta})$ at the present policy $\boldsymbol{\theta}$, $\boldsymbol{\nabla}_{\boldsymbol{\theta}} J_g$, then updating the improved policy to $\boldsymbol{\theta}' = \boldsymbol{\theta} + \beta.\boldsymbol{\nabla}_{\boldsymbol{\theta}} J_g$ – using suitably small β, and in this way *climb* the gradient to some local maximum.

However, there are two challenges to overcome: firstly there is no immediately available way to sample the gradient, and secondly in a multi-agent environment the expected return function $J_g : \boldsymbol{\Theta} \to \mathbb{R}$ is defined over the entire joint policy $\boldsymbol{\Theta}$, but each agent can at best maximise this function within its own policy space $\boldsymbol{\Theta}_g$.

The first challenge is addressed by RL techniques described as policy-gradient approaches, where the agent estimates the gradient in some way in order to climb it. The GPOMDP method, from [1], estimates the gradient directly, but other actor-critic methods [16] estimate the policy-gradient using intermediate estimates called the value- and Q-functions – details of these can be found in [8]. It is the actor-critic algorithms that we explore in this paper.

The second challenge is perhaps more profound, partly because in general each agent knows nothing about the other agents' choice of policies, and, more ominously, the other agents will exploit any predictable behaviour to their advantage. However, each agent is subject to this same challenge, and must explore action choices and their consequences to find the most advantageous policy with respect to all the others' policies.

3.1 Learning Pressure Fields

For single-agent problems, it is possible to use $\boldsymbol{\nabla}_{\boldsymbol{\theta}} J_g$ to incrementally adapt the policy, but, for multi-agent systems, the policy $\boldsymbol{\theta} = (\boldsymbol{\theta}_g, \boldsymbol{\theta}_{-g}) \in \boldsymbol{\Theta}$ is a joint policy, where agent g's policy part is $\boldsymbol{\theta}_g \in \boldsymbol{\Theta}_g$, and $\boldsymbol{\theta}_{-g}$ refers to the rest of the policy – which g knows nothing of and cannot control. The agent cannot find the full gradient, and instead is limited to finding the partial gradient at $\boldsymbol{\theta}$, $\nabla_{\boldsymbol{\theta}_g} J(\boldsymbol{\theta})$, and using this to improve it's policy with $\boldsymbol{\theta}'_g = \boldsymbol{\theta}_g + \beta.\nabla_{\boldsymbol{\theta}_g} J_g(\boldsymbol{\theta})$ [5, 8]. Since each $\boldsymbol{\Theta}_g$ is orthogonal to every other and together span $\boldsymbol{\Theta}$, we can combine these partial gradients to give a combined gradient of perceived local improvement, or learning pressure vector, given for n agents by the following vector, in block matrix notation.

$$\boldsymbol{\Gamma}(\boldsymbol{\theta}) = \left(\nabla_{\boldsymbol{\theta}_1} J_1(\boldsymbol{\theta})^{\intercal} \quad \nabla_{\boldsymbol{\theta}_2} J_2(\boldsymbol{\theta})^{\intercal} \ldots \nabla_{\boldsymbol{\theta}_n} J_n(\boldsymbol{\theta})^{\intercal}\right)^{\intercal}$$

This defines the learning pressure field (LPF), and we posit that it represents a prediction of the learning behaviour of the multi-agent system, where each agent is a policy-gradient learner [8].

3.2 Solving the Two-Step Game

In both the games defined in Section 2 there is a cycle of two-steps: the first transitions from s_0 to either s_L or s_R, the second transitions to s_0 or s_C. From then at all odd steps i, $s_i = s_L$ or s_R and $s_{i+1} = s_0$ or s_C. We say the process is two-step ergodic.

Definition 9 **(n-step ergodicity)** *A* Markov-Chain *with states S is n-step ergodic, if every state $s \in S$ is* positive recurrent*; has a* periodicity *of n (returns to s can only occur after $k.n$ steps, $k \in \mathbb{N}$); and is accessible by all states $s' \neq s$.*

A FASP *is n-step ergodic, if for any fixed policy, $\pi \in \Pi$, it resolves to an n-step ergodic Markov-Chain.*

Both the Bowling and Inverted two-step games are two-step ergodic. To solve them, we must first parametrise our policy. Recall that there are two agents each with a choice of two actions in a single observation. We parametrise π with $\theta_v, \theta_h \in (0, 1)$, which respectively represent the probabilities of g_v and g_h choosing a_1.

$$\pi_v^{\boldsymbol{\theta}}(a_1|o) = \theta_v \qquad \pi_v^{\boldsymbol{\theta}}(a_2|o) = 1 - \theta_v$$
$$\pi_h^{\boldsymbol{\theta}}(a_1|o) = \theta_h \qquad \pi_h^{\boldsymbol{\theta}}(a_2|o) = 1 - \theta_h$$

Since this game is two-step ergodic, we can find an surrogate to the asymptotic occupancy, called the composite asymptotic occupancy, $\bar{\mathbf{d}}^{\boldsymbol{\theta}}$, this is an unbiased linear combination of $\mathbf{d}^{\boldsymbol{\theta}}_{odd}$ and $\mathbf{d}^{\boldsymbol{\theta}}_{even}$, which represent the asymptotic occupancies at odd and even time-steps respectively. $\mathbf{d}^{\boldsymbol{\theta}}_{odd}$ can be found by fixing some $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ and only observing the resulting process at odd time-steps. This resolves to a Markov-Process with $S = \{s_L, s_R\}$, and transition matrix

$$T^{\boldsymbol{\theta}}_{odd} = \begin{pmatrix} \theta_h + \epsilon - 2\epsilon\theta_h & \theta_h + \epsilon - 2\epsilon\theta_h \\ 1 - \theta_h - \epsilon + 2\epsilon\theta_h & 1 - \theta_h - \epsilon + 2\epsilon\theta_h \end{pmatrix}$$

The associated asymptotic occupancy satisfies

$$\mathbf{d}^{\boldsymbol{\theta}}_{odd} = T^{\boldsymbol{\theta}}_{odd}\mathbf{d}^{\boldsymbol{\theta}}_{odd}$$

A similar approach is used to find $\mathbf{d}^{\boldsymbol{\theta}}_{even}$. We use $\bar{\mathbf{d}}^{\boldsymbol{\theta}} = \frac{1}{2}(\mathbf{d}^{\boldsymbol{\theta}}_{even} + \mathbf{d}^{\boldsymbol{\theta}}_{odd})$, in place of $\mathbf{d}^{\boldsymbol{\theta}}$ in Theorem 2 to predict the expected returns. Figures 2 and 3 show $J_v(\boldsymbol{\theta}) = r_1(\bar{\mathbf{d}}^{\boldsymbol{\theta}})$ and $J_h(\boldsymbol{\theta}) = r_1(\bar{\mathbf{d}}^{\boldsymbol{\theta}})$ respectively for the Bowling two-step, and indicate that agent g_v prefers joint policies $\boldsymbol{\theta} \approx (0, 0)$ or $(1, 1)$ while g_h prefers $\theta_v \approx 1 - \theta_h$.

The partial gradients $\nabla_{\theta_v} J_v(\boldsymbol{\theta})$ and $\nabla_{\theta_h} J_h(\boldsymbol{\theta})$ can be used to construct the LPF, $\boldsymbol{\Gamma}(\boldsymbol{\theta})$ – see Figure 4. These show, for example, that a joint policy $(\theta_v, \theta_h) = (0.2, 0.2)$, induces g_v to lower θ_v, while g_h wants to increase θ_h; this reflects the preferences for g_v and g_h described above. The Bowling two step-game has a non-strict NE at $(\theta_v, \theta_h) = (0.5, 0.5)$, but Figure 4 indicates that policy-gradient methods may converge on this point anyway. In a similar way, we construct the LPF for the Inverse two-step game, see Figure 5. This shows that the Inverse two-step game has no NE.

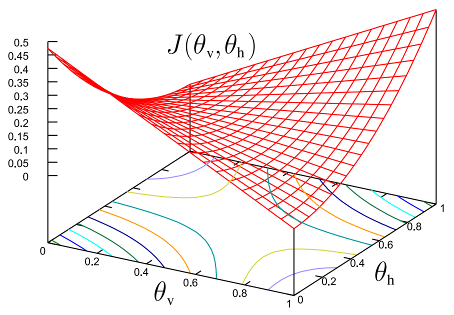

Figure 2. J_v for Bowling two-step game

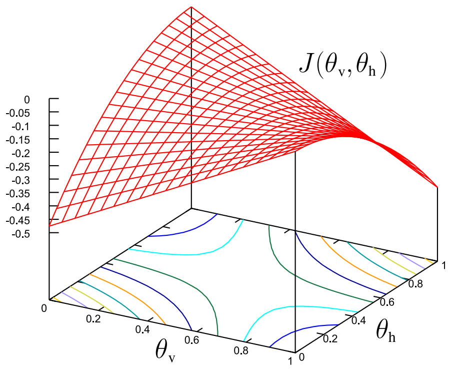

Figure 3. J_h for Bowling two-step game

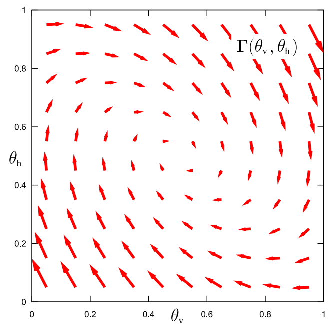

Figure 4. LPF for the Bowling two-step

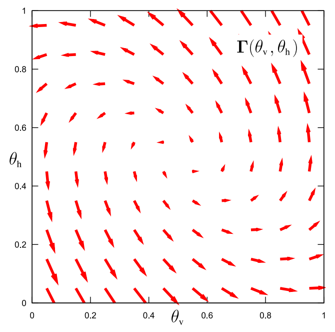

Figure 5. LPF for the Inverted two-step

4 Experiments

We test our predictions with the following actor-critic techniques, as described in [8].

TD: TD(λ) estimation – $\lambda = 0.9$ with QP optimisation.
MTD: MTD(λ) estimation – $\lambda = 0.9$ with QP optimisation.
PMC: PMC estimation – $\gamma = 0.9$ with QP optimisation.
QVP: PMC estimation – $\gamma = 0.9$ with QVP optimisation.

For each of the SMAFASP games from Section 2 and for each ordered combination of two algorithms, we ran thirty-two 100000-step traces in each game, and collected the reward accumulated by agent g_v in each case. Figures 6 and 7 show the mean accumulated rewards for the Bowling and Inverted two-step games respectively; the different algorithm combinations are indicated in the keys, and the results are staggered with a gap of 1000 time-steps for easier reading. For both two-step games, we can see that all the algorithms performed approximately equally, but in the Inverted two-step there is, on the whole, a greater spread of results both within and between tests. However, these graphs do not give us any more information, and while it may be possible to justify the choice of one or other algorithm for use with one of these games, we cannot generalise further about the algorithms' relative performance for unseen games. We cannot even say why there was a difference between the two games tested. Which graph is more practical in terms of generalisation?

For the same experiments, we also collected the joint-policy at each time-step, as represented by the vector $\boldsymbol{\theta} = (\theta_v, \theta_h)$. This allows us to observe how $\boldsymbol{\theta}$ changed with time, and how closely the path through policy space of each experiment (the policy trace) adhered to the LPF, shown in Section 3. First, we looked at both agents using the same algorithm. Figure 8 shows the first 10000-steps of the policy trace from when both agents used the TD(λ) algorithm on the Bowling two-step, $\lambda = 0.9$; two different line styles are used to help differentiate traces, and the policy at step 10000 is shown with a point. We were also able to see how different algorithms fared against one another. Figure 9 shows 10000-step traces, for agents g_v and g_h using PMC and QVP respectively, in the Inverted two-step game. Other algorithm combinations showed similar results.

These policy trace graphs give us more information, and show that all the algorithms did indeed perform equivalently, and support the notion that actor-critic algorithms follow the LPF, even when different algorithms compete against one another. In particular, the Inverted two-step traces seem to get trapped at policy values $\boldsymbol{\theta} \approx (1, 0)$ and $(0, 1)$ on occasion. This phenomena appears to be highly dependent on the initial policy and noise in the system, and not due to particular behavioural patterns. This indicates that the differences in the accumulated rewards for the Inverted two-step may be artifacts of the experimental process, rather than true differences between the algorithms. We have found that even more information can be inferred by looking at stop frame animations of the policy traces, as opposed to static images[3].

5 Discussion

In this paper, we have explained the need for accurate, efficient and robust RL algorithms for multi-agent domains, and for these algorithms to operate effectively in the domain of stochastic policies. This justifies the need to have appropriate tools and methodologies for evaluating such RL algorithms, and that is our focus here.

We began by formulating an umbrella framework, the FASP, under which a number of single- and multi-agent decision processes can be formulated, and then chose to focus on SMAFASPs – one variety of multi-agent FASP. This has given us the opportunity to investigate non-cooperative games, which have been shown to have the widest selection of potential phenomena [5, 8].

We concentrated on infinite-horizon processes with purely reactive policies, i.e. those with no memory. For infinite-horizon FASPs, considering only reactive policies is not as constrictive as it may first appear; the associated policies must be independent of time, as even an agent with a very large memory must eventually forget (fail to remember) some of its interaction history. Finite memories represent an improved but still incomplete historical record, and are therefore fundamentally no different from an observation that provided the same information. Further, in multi-agent non-cooperative problems, where agents compete in a series of episodes, the interaction from previous episodes may inform the current action. So, while the game appears episodic in structure, the on-going engagement between agents is not.

We showed that for small models, we can predict the expected return for any finite measure signal, such as a reward, and were able to construct learning pressure fields (LPF) from the partial gradients of these expected returns. With the LPF we claim to predict the direction of learning even in non-cooperative multi-agent systems, and we justify this both theoretically and experimentally.

In particular, the experiments showed that for our models the LPFs were followed surprisingly closely by all the actor-critic algorithms we selected to investigate, and gave us substantially more information than did the more familiar accumulated reward graphs. Moreover, these trace graphs showed an approximate equivalence between what are substantially different approaches. For instance the more traditional TD(λ) estimation with the stochastic incremental QP optimisation algorithm performs similarly to the fairly radically different PMC estimation approach using the triple table optimisation – see [8] for a complete description of these algorithms.

The next question one might ask is, 'Why would we need more than one actor-critic algorithm, if they all perform approximately equally in these experiments?'

The reasons could be theoretical or experimental and a shortlist is as follows. Firstly, The MTD(λ) and PMC estimation approaches are theoretically more sound for step-size parameters that can vary upwards as well as decay to zero [8]. Such an approach is used by agents in [4] to avoid exploitation by others.

Additionally, the PMC estimation approach gives theoretical bounds on the error in the gradient estimate [8]. This places a limit on the approximation each choice of parameter γ represents in terms of bias to the gradient, and may help us select these parameter values more appropriately. Also, the similarities in structure and performance of the PMC and TD(λ) algorithms seems to suggest there is some equivalence between using TD with λ and PMC with γ, when $\lambda = \gamma$. More experimental results are needed to justify this, and ideally we would like to see an equivalent theoretical result for TD(λ).

The QVP approach, unlike the other algorithms tested here, does not require the agent to keep an explicit representation of the policy, in terms of finite probability distributions. For large action spaces, this could be lighter weight, and may prove more amenable to learning with continuous action spaces [8].

In practice, there could be any number of reasons for one algorithm to be preferred over another. What this paper provides is one way to investigate these and other algorithms in detail, to facilitate this choice.

[3] Examples can be made available on request.

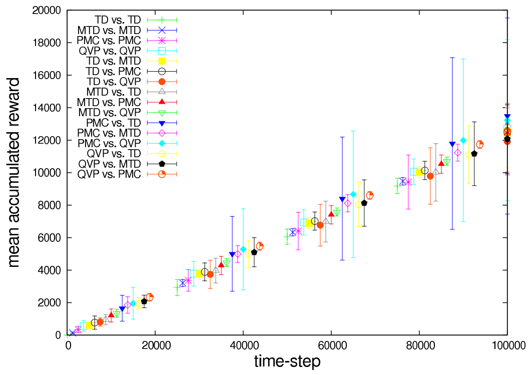

Figure 6. Average Rewards for g_v in Bowling two-step game.

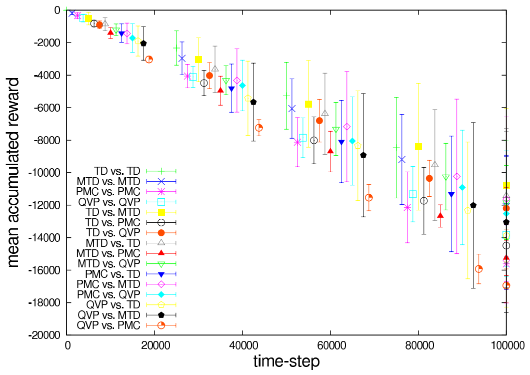

Figure 7. Average Rewards for g_v in Inverted two-step game.

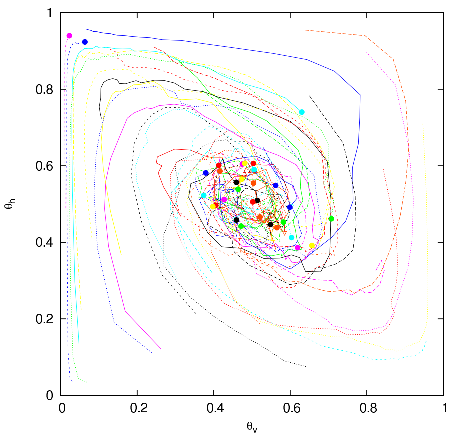

Figure 8. TD versus TD in Bowling two-step game

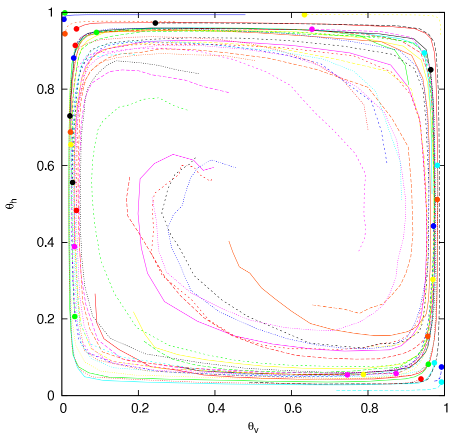

Figure 9. PMC versus QVP in Inverted two-step game

5.1 Future Work

At present, we are investigating how these actor-critic algorithms might be extended to find solutions in systems with more than one measure signal per agent, such as a reward and a risk signal. This might involve staying below one bound for one expected return and maximising the return on the other, or perhaps more interestingly to optimise all expected returns simultaneously in order to find candidate pareto optimal solutions.

Another theoretical advance would include adapting the PMC and MTD algorithms to learn natural (or covariant) gradients as has already been done for the TD(λ) approach [3, 14]. These approaches use the local shape of the gradient function to learn more efficiently.

In terms of experimental results and tools, we would like to develop a measure of deviance from the predicted LPF for experimental traces. This would give a quantitative way to leverage the knowledge contained in the LPF. The experimental testing of RL algorithms has only just begun, and we hope to extract more useful experimental results for RL algorithms with the help of tools like the LPF.

This is a rich seam of research the fruits of which could be powerful and flexible techniques for optimising control, as well as perhaps more insight into the theory of learning.

REFERENCES

[1] J. Baxter and P. Bartlett, ‘Infinite-Horizon Policy-Gradient Estimation’, *Journal of Artificial Intelligence Research*, **15**, (2001).

[2] D. Bernstein, S. Zilberstein, and N. Immerman, ‘The complexity of decentralized control of markov decision processes’, in *UAI*, (2000).

[3] S. Bhatnagar, R. Sutton, M. Ghavamzadeh, and M. Lee, ‘Incremental Natural Actor-Critic Algorithms’, in *NIPS*, (2008).

[4] M. Bowling and M. Veloso, ‘Simultaneous adversarial multi-robot learning’, in *IJCAI*, (2003).

[5] M. Bowling and M. Veloso, ‘Existence of Multiagent Equilibria with Limited Agents’, *Journal of Artificial Intelligence Research*, **22**, (2004).

[6] Michael Bowling and Manuela Veloso, ‘Multiagent learning using a variable learning rate’, *Artificial Intelligence*, **136**, 215–250, (2002).

[7] D. Braziunas, ‘POMDP solution methods: a survey’, Technical report, Department of Computer Science, University of Toronto, (2003).

[8] L. Dickens, *Learning to Act Stochastically*, Ph.D. dissertation, Imperial College (University of London), December 2009.

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

[10] L. Kaelbling, M. Littman, and A. Moore, ‘Reinforcement learning: A survey’, *Journal of Artificial Intelligence Research*, **4**, (1996).

[11] V. R. Konda and J. N. Tsitsiklis, ‘On actor-critic algorithms’, *SIAM J. Control Optim.*, **42**(4), (2003).

[12] M. Littman, ‘Markov games as a framework for multi-agent reinforcement learning’, in *ICML*, (1994).

[13] John Nash, *Non-Cooperative Games*, Ph.D. dissertation, Princeton University, 1950.

[14] J. Peters and S. Schaal, ‘Natural Actor-Critic’, *Neurocomputing*, **71**(7-9), (2008).

[15] S. Singh, T. Jaakkola, and M. Jordan, ‘Learning without State-Estimation in Partially Observable Markovian Decision Processes’, in *ICML*, (1994).

[16] R.S. Sutton and A.G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, 1998.

[17] K. Tuyls, P. J. Hoen, and B. Vanschoenwinkel, ‘An Evolutionary Dynamical Analysis of Multi-Agent Learning in Iterated Games’, *Autonomous Agents and Multi-Agent Systems*, **12**(1), 115–153, (2006).

[18] M. Zinkevich, A. Greenwald, and M. Littman, ‘Cyclic Equilibria in Markov Games’, in *NIPS*, (2005).