Anticipatory Fictitious Play

Alex Cloud , **Albert Wang** and **Wesley Kerr**
Riot Games
kacloud@gmail.com, {alwang, wkerr}@riotgames.com

Abstract

Fictitious play is an algorithm for computing Nash equilibria of matrix games. Recently, machine learning variants of fictitious play have been successfully applied to complicated real-world games. This paper presents a simple modification of fictitious play which is a strict improvement over the original: it has the same theoretical worst-case convergence rate, is equally applicable in a machine learning context, and enjoys superior empirical performance. We conduct an extensive comparison of our algorithm with fictitious play, proving an optimal $O(t^{-1})$ convergence rate for certain classes of games, demonstrating superior performance numerically across a variety of games, and concluding with experiments that extend these algorithms to the setting of deep multiagent reinforcement learning.

1 Introduction

Matrix games (also known as *normal-form games*) are an abstract model for interactions between multiple decision makers (or *players*). Fictitious play (FP) [Brown, 1951] is a simple algorithm for two-player matrix games. In FP, each player starts by playing an arbitrary strategy, then proceeds iteratively by playing the best strategy against the average of what the other has played so far. In some cases, such as two-player zero-sum games, the average strategies are guaranteed to converge to a pair of strategies such that neither player has an incentive to deviate from those strategies, i.e. a Nash equilibrium.

Although linear programming can be used to efficiently compute Nash equilibria of matrix games [Adler, 2013; Shoham and Leyton-Brown, 2008], fictitious play remains a topic of interest for at least two reasons. First, it serves as a model for how humans might arrive at Nash equilibria in real-world interactions [Luce and Raiffa, 1989; Brown, 1951; Conlisk, 1993a]. Second, FP is extensible to real-world games which are too large and complicated to be represented as linear programs in practice. Our work is primarily motivated by this second reason.

The initial step towards extending FP to real-world games was by [Kuhn, 1953], which established the equivalence of normal-form games (represented by matrices) and extensive-form games (represented by trees with additional structure). Loosely speaking, this means that results which apply for matrix games may also apply to much more complicated decision-making problems, such as ones that that incorporate temporal elements or varying amounts of hidden information.

Leveraging this equivalence, [Heinrich *et al.*, 2015] proposed an extension of FP to the extensive-form setting, full-width extensive-form fictitious play (XFP), and proved that it converges to a Nash equilibrium in two-player, zero-sum games. [Heinrich *et al.*, 2015] also proposed Fictitious Self Play (FSP), a machine learning approximation to XFP. In contrast to XFP, which is intractable for real-world games whose states cannot be enumerated in practice, FSP relies only on basic operations which can be approximated in a machine learning setting, like averaging (via supervised learning) and computing best responses (via reinforcement learning). In this way, FSP provides a version of fictitious play suitable for arbitrarily complex two-player, zero-sum games. Not long after the introduction of FSP, [Lanctot *et al.*, 2017] presented Policy Space Response Oracles (PSRO), a general framework for fictitious-play-like reinforcement learning (RL) algorithms in two-player, zero-sum games. These ideas were employed as part of the groundbreaking AlphaStar system that defeated professional players at StarCraft II [Vinyals *et al.*, 2019].

We introduce anticipatory fictitious play (AFP), a simple variant of fictitious play which is also amenable to reinforcement learning. In contrast to FP, where players iteratively update to exploit an estimate of the opponent's strategy, players in AFP update proactively to respond to the strategy that the opponent would use to exploit them.

We prove that AFP is guaranteed to converge to a Nash equilibrium in two-player, zero-sum games and establish an optimal convergence rate for two (narrow) classes of games that are of particular interest in learning for real world games [Balduzzi *et al.*, 2019], a class of "cyclic" games and a class of "transitive" games. Numerical comparisons suggest that in AFP eventually outperforms FP on virtually any game, and that its improvement over FP improves as games get larger. Finally, we propose a reinforcement learning version of AFP that is implementable as a one-line modification of an RL implementation of FP, such as FSP. These algorithms are compared on two stochastic games.

1.1 Related Work

Aside from the literature on fictitious play and its extension to reinforcement learning, there has been substantial work on "opponent-aware" learning algorithms. These algorithms incorporate information about opponent updates and are quite similar to anticipatory fictitious play.

In the context of evolutionary game theory, [Conlisk, 1993a] proposed an "extrapolation process," whereby two players in a repeated game each forecast their opponents' strategies and respond to those forecasts. Unlike AFP, where opponent responses are explicitly calculated, the forecasts are made by linear extrapolation based on the change in the opponent's strategy over the last two timesteps. [Conlisk, 1993b] proposed two types of "defensive adaptation," which are quite similar in spirit to AFP but differ in some important details; most importantly, while they consider the opponent's empirical payoffs at each step, they do not respond directly to what the opponent is likely to play given those payoffs.

[Shamma and Arslan, 2005] proposed derivative action fictitious play, a variant of fictitious play in the continous time setting in which a best response to a forecasted strategy is played, like in [Conlisk, 1993a]. The algorithm uses a derivative-based forecast that is analogous to the discrete-time anticipated response of AFP. However, their convergence results rely on a fixed, positive entropy bonus that incentivizes players to play more randomly, and they do not consider the discrete-time case.

[Zhang and Lesser, 2010] proposed Infinitesimal Gradient Ascent with Policy Prediction, in which two policy gradient learning algorithms continuously train against a forecast of the other's policy. Their algorithm represents the core idea of AFP, albeit implemented in a different setting. However, their proof of convergence is limited to $2x2$ games. [Foerster *et al.*, 2018] and [Letcher *et al.*, 2018] take this idea further, modifying the objective of a reinforcement learning agent so that it accounts for how changes in the agent will change the anticipated *learning* of the other agents. This line of research is oriented more towards equilibrium finding in general-sum games (e.g. social dilemmas), and less on efficient estimation of equilibria in strictly competitive two-player environments.

2 Preliminaries

A (finite) *two-player, zero-sum game* (2p0s game) is represented by a matrix $A \in \mathbb{R}^{m\times n}$, so that when player 1 plays i and player 2 plays j, the players observe payoffs $(A_{i,j}, -A_{i,j})$ respectively. Let $\Delta^k \subset \mathbb{R}^k$ be the set of probability vectors representing distributions over $\{1, \dots, k\}$ elements. Then a *strategy* for player 1 is an element $x \in \Delta^m$ and similarly, a strategy for player 2 is an element $y \in \Delta^n$.

A *Nash equilibrium* in a 2p0s game A is a pair of strategies (x^*, y^*) such that each strategy is optimal against the other, i.e.,

$$x^* \in \arg\max_{x\in\Delta^m} x^\intercal A y^* \quad \text{and} \quad y^* \in \arg\min_{y\in\Delta^n} (x^*)^\intercal A y.$$

The Nash equilibrium represents a pair of strategies that are "stable" in the sense that no player can earn a higher payoff by changing their strategy. At least one Nash equilibrium is guaranteed to exist in any finite game [Nash Jr, 1950].

Nash equilibria in 2p0s games enjoy a nice property not shared by Nash equilibria in general: in 2p0s games, if (x_1, y_1) and (x_2, y_2) are Nash equilibria, then (x_2, y_1) is a Nash equilibrium. Consequently, in a 2p0s game, it is natural to define a *Nash strategy* to be one that occurs as part of some Nash equilibrium. Note that the aforementioned property does not hold in general, so typically it is only valid to describe collections of strategies (one per player) as equilibria.

A *solution* to a 2p0s game A is a pair of strategies (x^*, y^*) such that

$$\min_{y\in\Delta^n} (x^*)^\intercal A y = \max_{x\in\Delta^m} x^\intercal A y^*.$$

We say $v^* = (x^*)^\intercal A y^*$, which is unique, the *value* of the game. In 2p0s games, Nash equilibria are equivalent to solutions [Shoham and Leyton-Brown, 2008], which is why we use the same notation. Finally, the *exploitability* of a strategy is the difference between the value of the game and the worst-case payoff of that strategy. So the exploitability of $x \in \Delta^m$ is $v^* - \min x^\intercal A$, and the exploitability of $y \in \Delta^n$ is $\max Ay - v^*$.

2.1 Fictitious Play

Let $e_1, e_2, \dots$ denote the standard basis vectors in $\mathbb{R}^m$ or $\mathbb{R}^n$. Let $\mathtt{BR}_A^k$ be the best response operator for player k, so that

$$(\forall y \in \Delta^n) \quad \mathtt{BR}_A^1(y) = \{e_i \in \mathbb{R}^m : i \in \arg\max Ay\};$$
$$(\forall x \in \Delta^m) \quad \mathtt{BR}_A^2(x) = \{e_j \in \mathbb{R}^n : j \in \arg\min x^\intercal A\}.$$

Fictitious play is given by the following process. Let $x_1 = \overline{x}_1 = e_i$ and $y_1 = \overline{y}_1 = e_j$ be initial strategies for some i, j. For each $t \in \mathbb{N}$, let

$$x_{t+1} \in \mathtt{BR}_A^1(\overline{y}_t); \qquad y_{t+1} \in \mathtt{BR}_A^2(\overline{x}_t);$$
$$\overline{x}_{t+1} = \frac{1}{t+1}\sum_{k=1}^{t+1} x_k; \qquad \overline{y}_{t+1} = \frac{1}{t+1}\sum_{k=1}^{t+1} y_k.$$

In other words, at each timestep t, each player calculates the strategy that is the best response to their opponent's average strategy so far. [Robinson, 1951] proved that the pair of average strategies $(\overline{x}_t, \overline{y}_t)$ converges to a solution of the game by showing that the exploitability of both strategies converge to zero.

Theorem 1. (Robinson, 1951) If $\{(x_t, y_t)\}_{t\in\mathbb{N}}$ is a FP process for a 2p0s game with payoff matrix $A \in \mathbb{R}^{m\times n}$, then

$$\lim_{t\to\infty} \min \overline{x}_t^\intercal A = \lim_{t\to\infty} \max A\overline{y}_t = v^*,$$

where v^* is the value of the game. Furthermore, a bound on the rate of convergence is given by

$$\max A\overline{y}_t - \min \overline{x}_t^\intercal A = O(t^{-1/(m+n-2)})$$

where $a = \max_{i,j} A_{i,j}$.[1]

[1] Robinson did not explicitly state the rate, but it follows directly from her proof, as noted in [Daskalakis and Pan, 2014] and explicated in the supplementary material, Section B.1.

3 Anticipatory Fictitious Play

Although FP converges to a Nash equilibrium in 2p0s games, it may take an indirect path. For example, in Rock Paper Scissors (RPS) with best response tiebreaking towards the minimum strategy index, the sequence of average strategies $\{\overline{x}_1, \overline{x}_2, \dots\}$ orbits the Nash equilibrium, slowly spiraling in with decreasing radius, as shown on the left in Figure 1. This tiebreaking scheme is not special; under random tiebreaking, the path traced by FP is qualitatively the same, resulting in slow convergence with high probability, as shown in Figure 2.

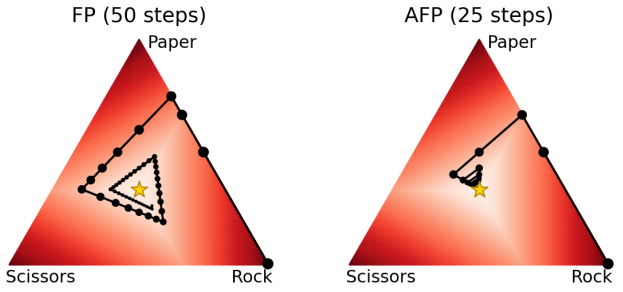


Figure 1: A visualization of the first 50 steps of FP $(\overline{x}_1^{\text{FP}}, \overline{x}_2^{\text{FP}}, \dots, \overline{x}_{50}^{\text{FP}})$ and first 25 steps of AFP $(\overline{x}_1^{\text{AFP}}, \overline{x}_2^{\text{AFP}}, \dots, \overline{x}_{25}^{\text{AFP}})$ on Rock Paper Scissors. This corresponds to an equal amount of computation per algorithm (50 best responses). Ties between best response strategies were broken according to the ordering 'Rock,' 'Paper,' 'Scissors.' The Nash equilibrium is marked by a star. The shading indicates the exploitability of the strategy at that point, with darker colors representing greater exploitability.

As illustrated by the RPS example, FP may spend many steps playing the same strategy while moving away from equilibrium, thus slowing its rate of convergence. Motivated by this observation, we propose *anticipatory fictitious play* (AFP), a version of fictitious play that "anticipates" the best response an adversary might play against the current strategy, and then plays the best response to an average of that and the adversary's current average strategy. (Simply responding directly to the opponent's response does not work; see the supplementary material, Section A.) Alternatively, one can think of AFP as a version of FP that "forgets" every other best response it calculates.

AFP is given by the following process. For some $i \in \{1, \dots, m\}$ and $j \in \{1, \dots, n\}$, let $x_1 = \overline{x}_1 = e_i$ and $y_1 = \overline{y}_1 = e_j$ be initial strategies for each player. For each $t \in \mathbb{N}$, define

$$
\begin{aligned}
&x'_{t+1} \in \text{BR}_A^1(\overline{y}_t); && y'_{t+1} \in \text{BR}_A^2(\overline{x}_t);\\
&\overline{x}'_{t+1} = \tfrac{t}{t+1}\overline{x}_t + \tfrac{1}{t+1}x'_{t+1}; && \overline{y}'_{t+1} = \tfrac{t}{t+1}\overline{y}_t + \tfrac{1}{t+1}y'_{t+1};\\
&x_{t+1} \in \text{BR}_A^1(\overline{y}'_{t+1}); && y_{t+1} \in \text{BR}_A^2(\overline{x}'_{t+1});\\
&\overline{x}_{t+1} = \frac{1}{t+1}\sum_{k=1}^{t+1} x_k; && \overline{y}_{t+1} = \frac{1}{t+1}\sum_{k=1}^{t+1} y_k. \qquad (1)
\end{aligned}
$$

Here, x'_{t+1} and y'_{t+1} are the best response to the opponent's average strategy. They are the strategies that FP would have played at the current timestep. In AFP, each player "anticipates" this attack and defends against it by calculating the opponent's average strategy that include this attack ($\overline{x}'_t$ and $\overline{y}'_t$), and then playing the best response to the anticipated average strategy of the opponent.

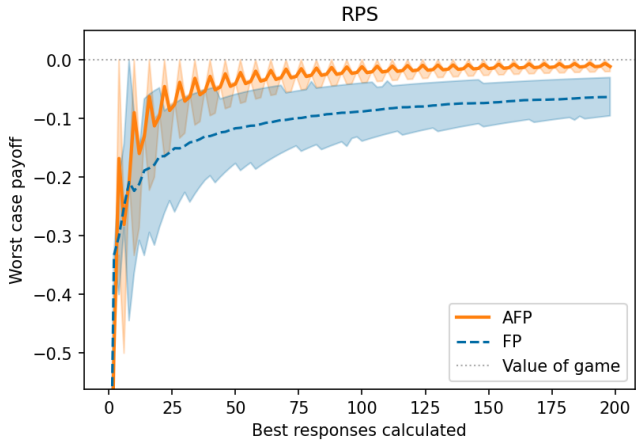


Figure 2: Comparison of FP and AFP performance ($\min \overline{x}_t^\intercal A$) on RPS with random tiebreaking. The highlighted region depicts the 10th and 90th percentiles across 10,000 runs. All variation is due to randomly sampled tiebreaking. The value of the game is $v^* = 0$.

In Figure 1, we see the effect of anticipation geometrically: AFP "cuts corners," limiting the extent to which it overshoots its target. In contrast, FP aggressively overshoots, spending increasingly many steps playing strategies that take it further from its goal. The effect on algorithm performance is pronounced, with AFP hovering near equilibrium while FP slowly winds its way there.

Of course, RPS is a very specific example. The rest of the paper is dedicated to understanding AFP's performance in greater generality. We begin by proving that AFP converges to a Nash equilibrium.

Proposition 1. If $\{(x_t, y_t)\}$ is an AFP process for a 2p0s game with payoff matrix $A \in \mathbb{R}^{m \times n}$, the conclusion of Theorem 1 holds for this process. Namely, AFP converges to a Nash equilibrium, and it converges no slower than the rate that bounds FP.

Proof. (Idea) Generalize the original proof of Theorem 1. We work with accumulating payoff vectors $U(t) = tA^\intercal \overline{x}_t$ and $V(t) = tA\overline{y}_t$. In the original proof, a player 1 strategy index $i \in \{1, \dots, m\}$ is called *eligible* at time t if $i \in \arg\max V(t)$ (similarly for player 2). We replace eligibility with the notion of *E-eligibility*, satisfied by an index $i \in \arg\max[V(t) + E]$, for any $E \in \mathbb{R}^m$ with $\|E\|_\infty < \max_{i,j} |A_{i,j}|$. Essentially, an index is E-eligible if it corresponds to a best response to a perturbation of the opponent's history $\overline{y}_t$. Treating the in-between strategies in AFP, $\overline{x}'_t$ and $\overline{y}'_t$, as perturbations of $\overline{x}_t$ and $\overline{y}_t$, it follows that AFP satisfies the conditions for the generalized result. A complete proof is given in the supplementary material, Section B. □

4 Application to Normal-Form Games

Proposition 1 establishes that AFP converges and that AFP's worst-case convergence rate satisfies the same bound as FP's, where the worst-case is with respect to games and tiebreaking rules. The next proposition shows that for two classes of games of interest, AFP not only outperforms FP, but attains an optimal rate. In both classes, our proofs will reveal that AFP succeeds where FP fails because AFP avoids playing repeated strategies. The results hold for general applications of FP and AFP rather than relying on specific tiebreaking rules.

The classes of games that we analyze are intended to serve as abstract models of two fundamental aspects of real-world games: transitivity (akin to "skillfullness:" some ways of acting are strictly better than others) and nontransitivity (most notably, in the form of strategy cycles like Rock $<$ Paper $<$ Scissors $<$ Rock). Learning algorithms for real-world games must reliably improve along the transitive dimension while accounting for the existence of strategy cycles; see [Balduzzi *et al.*, 2019] for further discussion.

For each integer $n \geq 3$, define payoff matrices C^n and T^n by

$$C^n_{i,j} = \begin{cases} 1 & \text{if } i = j+1 \mod n; \\ -1 & \text{if } i = j-1 \mod n; \\ 0 & \text{otherwise;} \end{cases} \quad \text{and}$$

$$T^n_{i,j} = \begin{cases} (n-i+2)/n & \text{if } i = j+1; \\ -(n-i+2)/n & \text{if } i = j-1; \\ 0 & \text{otherwise,} \end{cases}$$

for $i, j \in \{1, \ldots, n\}$. The game given by C^n is a purely cyclic game: each strategy beats the one before it and loses to the one after it; C^3 is Rock Paper Scissors. For each C^n, a Nash strategy is $[n^{-1}, \ldots, n^{-1}]^\intercal$. The game given by T^n could be considered "transitive:" each strategy is in some sense better than the last, and $[0, \ldots, 0, 1]^\intercal$ is a Nash strategy. The payoffs are chosen so that each strategy i is the unique best response to $i-1$, so that an algorithm that learns by playing best responses will progress one strategy at a time rather than skipping to directly to strategy n.[2]

The following proposition establishes a convergence rate of $O(t^{-1})$ for AFP applied to C^n and T^n. This rate is optimal within the class of time-averaging algorithms, because the rate at which an average changes is t^{-1}. Note: we say a random variable $Y_t = \Omega_p(g(t))$ if, for any $\epsilon > 0$, there exists $c > 0$ such that $P[Y_t < cg(t)] < \epsilon$ for all t.

Proposition 2. FP and AFP applied symmetrically to C^n and T^n obtain the rates given in Table 1. In particular, if $\{x_t, x_t\}_{t \in \mathbb{N}}$ is an FP or AFP process for a 2p0s game with payoff matrix $G \in \{C^n, T^n\}$ with tiebreaking as indicated, then $\max G\overline{x}_t = R(t)$. Tiebreaking refers to the choice of $x_{t+1} \in \arg\max \mathrm{BR}^1_G(\overline{x}_t)$ when there are multiple maximizers. The "random" tiebreaking chooses between tied strategies independently and uniformly at random. For entries marked with "arbitrary" tiebreaking, the convergence rate holds no matter how tiebreaks are chosen.

[2]Note that the definition of transitivity in [Balduzzi *et al.*, 2019] is much stronger. Using it here would result in a game with a single dominant strategy, with nothing interesting to learn.

Algorithm	Game G	Tiebreaking	Rate $R(t)$	Caveats
FP	C^n	random	$\Omega_p(t^{-1/2})$	
AFP	C^n	arbitrary	$O(t^{-1})$	$n = 3, 4$
FP	T^n	arbitrary	$\Omega(t^{-1/2})$	$t < t^*(n)$
AFP	T^n	arbitrary	$O(t^{-1})$	

Table 1: Convergence rates for FP and AFP on C^n and T^n.

Proof. Full proofs of all cases are provided in the supplementary material, Section C. □

The proofs of Proposition 2 establish a theme: FP can be slow because it spends increasingly large amounts of time progressing between strategies (playing $x_t = x_{t+1} = \cdots = x_{t+k}$ with k increasing as t increases), whereas AFP avoids this. (Return to Figure 1 for a visual example.)

Some further comments on the results: we only obtain the $O(t^{-1})$ rate for AFP applied to C^n in the $n = 3, 4$ case. We conjecture that: (i) for a specific tiebreaking rule, AFP has the same worst-case rate as FP but with a better constant, (ii) under random tiebreaking, AFP is $O_p(t^{-1})$ for all n. This is reflected in numerical simulations for large n, as shown in the supplementary material, Section D.

Our results are noteworthy for their lack of dependence on tiebreaking: worst-case analyses of FP typically rely on specific tiebreaking rules; see [Daskalakis and Pan, 2014], for example. As for the "$t < t^*(n)$" caveat for FP applied to T^n, this is an unremarkable consequence of analyzing a game with a pure strategy equilibrium (all probability assigned to a single strategy). We write $t^*(n)$ to indicate the first index at which FP plays e_n. Both FP and AFP will play e_n forever some finite number of steps after they play it for the first time, thus attaining a t^{-1} rate as the average strategy "catches up" to e_n. Our result shows that until this point, FP is slow, whereas AFP is always fast. As before, AFP's superior performance is reflected in numerical simulations, as shown in the supplementary material, Section D.

4.1 Numerical Results

In order to compare FP and AFP more generally, we sample large numbers of random payoff matrices and compute aggregate statistics across them. Matrix entries are sampled as independent, identically distributed, standard Gaussian variables (note that the shift- and scale-invariance of matrix game equilibria implies that the choice of mean and variance is inconsequential). Since FP and AFP are so similar, and AFP computes two best responses per timestep, it is natural to wonder: is AFP's superior performance just an artifact of using more computation per timestep? So, in order to make a fair comparison, we compare the algorithms by *the number of best responses calculated* instead of the number of timesteps (algorithm iterations). Using the worst-case payoff as the measure of performance, we compare FP and AFP based on the number of responses computed and based on matrix size in Figures 3 and 4.

The result is that AFP is clearly better on both counts. Although FP is better for a substantial proportion of 30×30 games at very early timesteps t, AFP quickly outpaces FP,

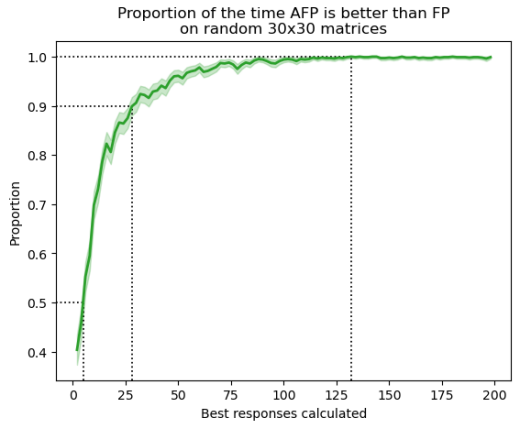


Figure 3: For 1,000 randomly sampled (30,30) matrices A, the proportion of the time that $\min(\overline{x}^{\text{AFP}}_{r/2})^\intercal A \geq \min(\overline{x}^{\text{FP}}_{r})^\intercal A$ for $r = 2, 4 \ldots, 200$. A 95% Agresti-Coull confidence interval [Agresti and Coull, 1998] for the true proportion is highlighted. Note that after only about six best responses, AFP is better half the time, and by 130, AFP is better than FP essentially 100% of the time.

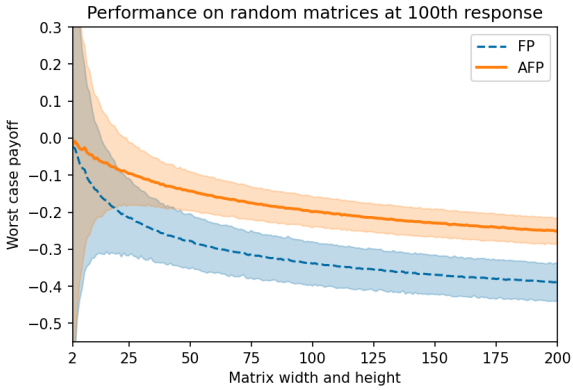


Figure 4: Average performance of FP vs. AFP at the 100th best response (timestep 100 for FP, timestep 50 for AFP) as matrix size is varied. All matrices are square. Highlighted regions show the 10th and 90th percentiles.

eventually across each of 1,000 matrices sampled. In terms of matrix size, FP and AFP appear to have equivalent average performance for small matrices, but quickly grow separated as matrix size grows, with AFP likely to be much better.

5 Application to Reinforcement Learning

We apply reinforcement learning (RL) [Sutton and Barto, 2018] versions of FP and AFP in the context of a (two-player, zero-sum, symmetric) *stochastic game* [Shapley, 1953], defined by the tuple $(\mathcal{S}, \mathcal{O}, \mathcal{X}, \mathcal{A}, \mathcal{P}, \mathcal{R}, p_0)$, where $\mathcal{S}$ is the set of possible states of the environment, $\mathcal{O}$ is the set of possible observations received by an agent, $\mathcal{X} : \mathcal{S} \to \mathcal{O} \times \mathcal{O}$ gives the observations for each player based on the current state, $\mathcal{A}$ is the set of available actions, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{A} \to \Delta(\mathcal{S})$ defines the transition dynamics for the environment given each player's action, $\mathcal{R} : \mathcal{S} \to \mathbb{R} \times \mathbb{R}$ defines the reward for both players such that $\mathcal{R}(s_t) = (r_t, -r_t)$ are the rewards observed by each player at time t, and $p_0 \in \Delta(\mathcal{S})$ is the initial distribution of states, such that $s_0 \sim p_0$. Let $\mathcal{H}$ be the set of possible sequences of observations. Then a *policy* is a map $\pi : \mathcal{H} \to \Delta(A)$. An *episode* is played by iteratively transitioning by the environment according to the actions sampled from each players' policies at each state. Players 1 and 2 earn *returns* $(\sum_t r_t, -\sum_t r_t)$. The reinforcement learning algorithms we consider take sequences of observations, actions, and rewards from both players and use them to incrementally update policies toward earning greater expected returns. For background on reinforcement learning, see [Sutton and Barto, 2018]. For details on machine learning approximations to FP, see [Heinrich *et al.*, 2015]. Table 2 gives a high-level overview of the relationship.

Normal-form game	Extensive-form game
Strategy	Policy
Payoff $A_{i,j}$	Expected return $E_{\pi_i,\pi_j}(\sum_t r_t)$
Best response	Approximate best response by RL
Strategy mixture $\sum \alpha_i x_i$, $\sum \alpha_i = 1, \alpha_i \geq 0$	At start of episode, sample policy π_i with probability α_i. Play entire episode with π_i.

Table 2: The normal-form game analogies used to extend FP and AFP to reinforcement learning.

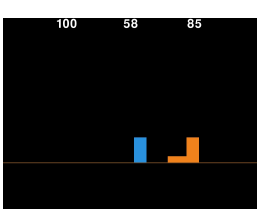
We use two environments, our own TinyFighter, and Running With Scissors, from [Vezhnevets *et al.*, 2020].

5.1 Environments

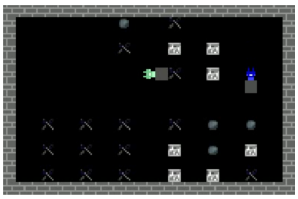
TinyFighter is a minimal version of an arcade-style fighting game shown in Figure 5a. It features two players with four possible actions: `Move Left`, `Move Right`, `Kick`, and `Do Nothing`. Players are represented by a rectangular body and when kicking, extend a rectangular leg towards the opponent.

Kicking consists of three phases: Startup, Active, and Recovery. Each phase of a kick lasts for a certain number of frames, and if the Active phase of the kick intersects with any part of the opponent (body or leg), a hit is registered. When a hit occurs, the players are pushed back, the opponent takes damage, and the opponent is stunned (unable to take actions) for a period of time. In the Startup and Recovery phases, the leg is extended, and like the body, can be hit by the opponent if the opponent has a kick in the active phase that intersects the player. The game is over when a player's health is reduced to zero or when time runs out.

Player observations are vectors in $\mathbb{R}^{13}$ and contain information about player and opponent state: position, health, an 'attacking' indicator, a 'stunned' indicator, and how many frames a player has been in the current action. The observation also includes the distance between players, time remaining, and the direction of the opponent (left or right of self). The game is partially observable, so information about the opponent's state is hidden from the player for some number of frames (we use four, and the game runs at 15 frames per second). This means a strong player must guess about the distribution of actions the opponent may have taken recently and to respond to that distribution; playing deterministically will



(a) TinyFighter, where players dance back and forth, attempting to land kicks on each other in order to reduce the other's health from 100 to zero.



(b) Running With Scissors, a 2D gridworld where players collect items before tagging each other to earn reward based on the proportion of items in their inventory.

Figure 5: Screenshots of multiagent RL environments.

allow the opponent to exploit the player and so a stochastic strategy is required to play well.

Running With Scissors (RWS) is a spatiotemporal environment with partial observability with potential for nontransitive relationships between policies. As shown in Figure 5b, RWS is a 2D gridworld with a few types of entities: two agents; three types of items: rock, paper, and scissors, which can be picked up by the agents to add to their inventories; and impassable walls. In addition to moving around and picking up items, agents in RWS have a "tag" action, which projects a cone in front of them for a single frame. If the cone hits the other agent, the episode ends and each agent receives rewards based on the payoff matrix C^3 according to the ratios of each item in their inventory. Agents can only see their own inventory and a 5×5 grid in front of them and can remember the last four frames they've seen, so in order to perform well they must infer what items the opponent has picked up. [Vezhnevets *et al.*, 2020] and [Liu *et al.*, 2022] (Appendix B.1.) feature further discussion of the environment.

5.2 Adapting FP and AFP to Reinforcement Learning

Neural Population Learning (NeuPL) [Liu *et al.*, 2022] is a framework for multiagent reinforcement learning wherein a collection of policies is learned and represented by a single neural network and all policies train continuously. For our experiments, we implement FP and AFP within NeuPL, as shown in Algorithm 1. For reference, we also include a simple RL version of FP and AFP in the style of PSRO in the supplementary material, Section F.

In NeuPL-FP/AFP, the opponent sampler $\mathfrak{O}$ determines the distributions of opponents that each agent faces and is the only difference between the FP and AFP implementations. We have, for each $t > 1$,

$$\mathfrak{O}^{\text{FP}}(t) = \text{Uniform}(\{1, 2, 3, \ldots, t-1\}), \text{ and}$$
$$\mathfrak{O}^{\text{AFP}}(t) = \text{Uniform}(\{k < t : k \text{ odd}\} \cup \{t-1\}).$$

These distributions are depicted in Figure 6. Just as each step of FP involves computing a best response to an average against all prior strategies, sampling from $\mathfrak{O}^{\text{FP}}(t)$ corresponds to training agent t uniformly against the prior policies; just as AFP can be thought of "forgetting" every other index, $\mathfrak{O}^{\text{AFP}}(t)$ trains learner index t uniformly against every odd

Algorithm 1 NeuPL-FP/AFP

$\mathfrak{O} \in \{\mathfrak{O}^{\text{FP}}, \mathfrak{O}^{\text{AFP}}\}$ ▷ Input: FP or AFP opponent sampler.
$\{\Pi_\theta(t) : \mathcal{H} \to \Delta(\mathcal{A})\}_{t=1}^{n}$ ▷ Input: neural population net.
for batch $b = 1, 2, 3, \ldots,$ **do**
  $B \leftarrow \{\}$
  while per-batch compute budget remains **do**
    $T_{\text{learner}} \sim \text{Uniform}(\{1, \ldots, n\})$
    $T_{\text{opponent}} \sim \mathfrak{O}(T_{\text{learner}})$
    $D_{\text{learner}} \leftarrow \text{PLAYEP}(\Pi_\theta(T_{\text{learner}}), \Pi_\theta(T_{\text{opponent}}))$
    $B \leftarrow B \cup D_{\text{learner}}$
  end while
  $\Pi_\theta \leftarrow \text{REINFORCEMENTLEARNINGUPDATE}(B)$
end for

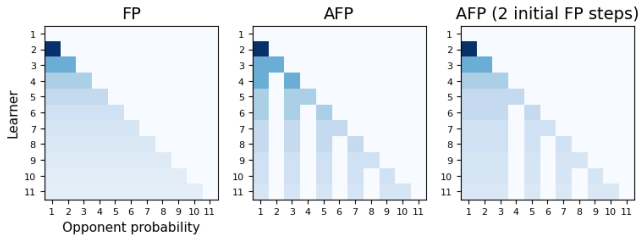


Figure 6: A visual depiction of the distributions of opponents ("meta-strategies" in PSRO or NeuPL) each learner faces in a population learning implementation of FP or AFP. The (i, j) entry is the probability that, given that agent i is training, it will face agent j in a particular episode. Dark blue indicates probability 1, white indicates probability 0.

indexed policy plus the most recent policy. The neural population net $\Pi_\theta(t) : \mathcal{H} \to \Delta(A)$ defines a different policy for each agent index t, and can equivalently be represented as $\Pi_\theta(a|s, t)$.

5.3 Experimental Setup

For the neural population net, we use an actor-critic [Sutton and Barto, 2018] architecture similar to that used for RWS in [Liu *et al.*, 2022]: first, a module is used to process environment observations into a dense representation that is shared between an actor head and a critic head. The actor takes this vector, concatenates it with a vector representing the distribution of opponents faced by the currently training agent t (e.g., $[0.5, 0.5, 0, \ldots, 0]$ for agent $t = 3$ in FP or AFP), then processes it with a dense MLP with ReLu activations, with action masking applied prior to a final Softmax layer. The critic is similar, except an additional input is used: the index of the opponent sampled at the beginning of this episode, matching the original implementation.[3] For the exact architectures used, see the supplementary material, Section E. We

[3]Note that the actor (policy network) does not observe which opponent it faces, only the *distribution* over agents it faces; this is important because otherwise our agent would not learn a best response to an average policy as intended in FP and AFP. The reason for including this information for the critic (value network) is that it may reduce the variance of the value function estimator.

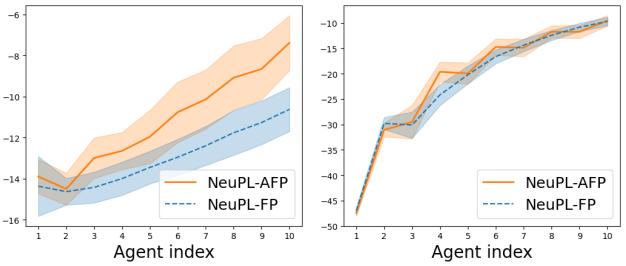
use a neural population size of $n = 11$. Based on matrix game simulations and preliminary RL experiments, we determined that AFP performs slightly better in a short time horizon when initialized with two steps of FP, so we do this, as shown in the final panel of Figure 6. See the supplementary material, Section D for a figure comparing performance in the matrix setting.

We implemented NeuPL within a basic self-play reinforcement learning loop by wrapping the base environment (TinyFighter or RWS) within a lightweight environment that handles NeuPL logic, such as opponent sampling. For reinforcement learning, we use the Asynchronous Proximal Policy Optimization (APPO) algorithm [Schulman *et al.*, 2017], a distributed actor-critic RL algorithm, as implemented in RLLib [Moritz *et al.*, 2018] with a single GPU learner. Hyperparameter settings are given in the supplementary material, Section E. We train the entire neural population net (agents 1-11) for 12,500 steps, where a step is roughly 450 minibatch updates of stochastic gradient descent. This corresponds to about five days of training. For each of the two environments, we repeat this procedure independently 10 times for FP and 10 times for AFP.

5.4 Results

To evaluate exploitability, we made use of the fact that each FP and AFP neural population are made up of agents trained to "exploit" the ones that came before them. Specifically, each agent is trained to approximate a best response to the average policy returned by the algorithm at the previous timestep. So, to estimate the exploitability of NeuPL-FP or NeuPL-AFP at step $t \in \{1, \ldots, n-1\}$, we simply use the average return earned by agent $t+1$ against agents $\{1, \ldots, t\}$ to obtain the *within-population exploitability* at t. This is a convenient metric, but insufficient on its own. In order for it to be meaningful, the agents in the population must have learned approximate best responses that are close to actual best responses; if they have not, it could be that within-population exploitability is low, not because the average policy approximates a Nash policy but because nothing had been learned at all. To account for this, we also evaluate the populations learned using *relative population performance* [Balduzzi *et al.*, 2019], which measures the strength of one population of agents against the other. The purpose of using relative population performance is simply to verify that one algorithm did not produce generally more competent agents than the other.

We paired each of the 10 replicates of FP and AFP and computed the relative population performance for each. On TinyFighter, the average was -0.73, with a Z-test-based 90% confidence interval width of 1.37. On RWS, the average was 4.60, with a Z-test-based 90% confidence interval width of 3.50. We conclude that the agents learned by FP and AFP are not statistically significantly different in terms of performance for TinyFighter, but the agents learned by FP have a slight, statistically significant advantage in RWS. However, these differences are small relative to the total obtainable reward in either environment (20 for TinyFighter, roughly 60 for RWS), so we conclude it is reasonable to use within-population exploitability to compare FP and AFP, as shown in Figure 7. For consistency with the matrix game simulation results, we plot worst case payoff, which is simply the negative of exploitability in this case.



(a) TinyFighter (b) Running With Scissors

Figure 7: Estimated worst-case payoffs for FP and AFP on two stochastic games. Highlighting indicates a pointwise 90% confidence region.

We find that AFP has a significantly better worst-case payoff of -7.4 versus 10.6 for FP at the final timestep in TinyFighter. This corresponds to a noteworthy 16% reduction in exploitability relative to the total possible reward of 20 that can be earned in TinyFighter. In RWS, the algorithms have essentially identical performance. The fact that AFP's advantage varies widely by environment is not surprising. The matrix game simulations in Figure 3 showed that until over 100 steps of each algorithm, there is some proportion of games for which FP performs better. Correspondingly, we would expect that there is a nontrivial proportion of stochastic games where NeuPL-FP outperforms NeuPL-AFP for small population sizes. Although we expect NeuPL will not be able to support over 100 policies (the original paper used population size 8), it would be possible to do so within the PSRO framework. This remains a topic for further investigation.

6 Conclusion

We proposed a variant of fictitious play for faster estimation of Nash equilibria in two-player, zero-sum games and extended it to the reinforcement learning setting. AFP is intuitive, easy to implement, and supported by theory and numerical simulations which suggest that it is virtually always preferable to fictitious play. It is a natural choice for a first implementation of a multiagent RL algorithm for competitive games, as it requires access only to a standard RL algorithm and no special tuning, configuration, or auxiliary computation. Consequently, we shed new light on two motivating problems for fictitious play: primarily, large-scale multiagent reinforcement learning for complicated real-world games; also, modeling strategic decision making in humans. Further work is needed to understand the conditions under which AFP outperforms FP in the RL setting.

Acknowledgements

The authors sincerely thank Ryan Martin for continued mentorship, especially around theory and the presentation of the paper; Philip Wardlaw for orchestrating preliminary reinforcement learning experiments; Adam Venis and Angelo Olcese for contributions to the proofs of FP and AFP applied to

C^n; Jesse Clifton and Eric Laber for helpful comments on a draft; Jesse Clifton and Marc Lanctot for suggesting related works that had been overlooked; and Siqi Liu for helpful explanations and implementation details for NeuPL.

References

[Adler, 2013] Ilan Adler. The equivalence of linear programs and zero-sum games. *International Journal of Game Theory*, 42(1):165–177, 2013.

[Agresti and Coull, 1998] Alan Agresti and Brent A Coull. Approximate is better than "exact" for interval estimation of binomial proportions. *The American Statistician*, 52(2):119–126, 1998.

[Balduzzi *et al.*, 2019] David Balduzzi, Marta Garnelo, Yoram Bachrach, Wojciech Czarnecki, Julien Perolat, Max Jaderberg, and Thore Graepel. Open-ended learning in symmetric zero-sum games. In *International Conference on Machine Learning*, pages 434–443. PMLR, 2019.

[Brown, 1951] George W Brown. Iterative solution of games by fictitious play. *Activity Analysis of Production and Allocation*, 13(1):374–376, 1951.

[Conlisk, 1993a] John Conlisk. Adaptation in games: Two solutions to the crawford puzzle. *Journal of Economic Behavior & Organization*, 22(1):25–50, 1993.

[Conlisk, 1993b] John Conlisk. Adaptive tactics in games: Further solutions to the crawford puzzle. *Journal of Economic Behavior & Organization*, 22(1):51–68, 1993.

[Daskalakis and Pan, 2014] Constantinos Daskalakis and Qinxuan Pan. A counter-example to karlin's strong conjecture for fictitious play. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 11–20, Philadelphia, PA, USA, 2014. IEEE, IEEE.

[Foerster *et al.*, 2018] Jakob Foerster, Richard Y Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. Learning with opponent-learning awareness. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 122–130, ', 2018. AAMAS.

[Heinrich *et al.*, 2015] Johannes Heinrich, Marc Lanctot, and David Silver. Fictitious self-play in extensive-form games. In *International Conference on Machine Learning*, pages 805–813, 2015.

[Kuhn, 1953] Harold Kuhn. Extensive games and the problem of information. In *Contributions to the Theory of Games*, pages 193–216. Princeton University Press, 1953.

[Lanctot *et al.*, 2017] Marc Lanctot, Vinicius Zambaldi, Audrunas Gruslys, Angeliki Lazaridou, Karl Tuyls, Julien Perolat, David Silver, and Thore Graepel. A unified game-theoretic approach to multiagent reinforcement learning. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, -, 2017. Curran Associates, Inc.

[Letcher *et al.*, 2018] Alistair Letcher, Jakob Foerster, David Balduzzi, Tim Rocktäschel, and Shimon Whiteson. Stable opponent shaping in differentiable games. In *International Conference on Learning Representations*, page ', Vancouver Convention Center, Vancouver, BC, Canada, 2018. ICLR.

[Liu *et al.*, 2022] Siqi Liu, Luke Marris, Daniel Hennes, Josh Merel, Nicolas Heess, and Thore Graepel. Neupl: Neural population learning. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[Luce and Raiffa, 1989] R Duncan Luce and Howard Raiffa. *Games and Decisions: Introduction and Critical Survey*. Courier Corporation, ', 1989.

[Moritz *et al.*, 2018] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I. Jordan, and Ion Stoica. Ray: A distributed framework for emerging {AI} applications. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 561–577, 2018.

[Nash Jr, 1950] John F Nash Jr. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, 36(1):48–49, 1950.

[Robinson, 1951] Julia Robinson. An iterative method of solving a game. *Annals of Mathematics*, 54(2):296–301, 1951.

[Schulman *et al.*, 2017] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[Shamma and Arslan, 2005] Jeff S Shamma and Gürdal Arslan. Dynamic fictitious play, dynamic gradient play, and distributed convergence to nash equilibria. *IEEE Transactions on Automatic Control*, 50(3):312–327, 2005.

[Shapley, 1953] Lloyd S Shapley. Stochastic games. *Proceedings of the National Academy of Sciences*, 39(10):1095–1100, 1953.

[Shoham and Leyton-Brown, 2008] Yoav Shoham and Kevin Leyton-Brown. *Multiagent Systems: Algorithmic, Game-theoretic, and Logical Foundations*. Cambridge University Press, 2008.

[Sutton and Barto, 2018] Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT press, Cambridge, Massachusetts, 2018.

[Vezhnevets *et al.*, 2020] Alexander Vezhnevets, Yuhuai Wu, Maria Eckstein, Rémi Leblond, and Joel Z Leibo. Options as responses: Grounding behavioural hierarchies in multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 9733–9742, ', 2020. PMLR, PMLR.

[Vinyals *et al.*, 2019] Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre,

Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander S. Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom L. Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

[Zhang and Lesser, 2010] Chongjie Zhang and Victor Lesser. Multi-agent learning with policy prediction. In *Twenty-fourth AAAI conference on artificial intelligence*, pages 927–937, ', 2010. AAAI.