Computationally Efficient Learning in Large-Scale Games: Sampled Fictitious Play Revisited

BRIAN SWENSON†*, SOUMMYA KAR† AND JOÃO XAVIER*

Abstract—Fictitious Play (FP) is a popular algorithm known to achieve Nash equilibrium learning in certain large-scale games. However, for games with many players, the computational demands of the FP algorithm can be prohibitive. Sampled FP (SFP) is a variant of FP that mitigates computational demands via a Monte Carlo approach. While SFP does mitigate the complexity of FP, it can be shown that SFP still uses information in an inefficient manner. The paper generalizes the SFP convergence result and studies a stochastic-approximation-based variant that significantly reduces the complexity of SFP.

I. INTRODUCTION

Game-theoretic learning algorithms allow groups of interacting decision makers to cooperatively learn to act in an optimal fashion. Originally used to help economists understand how agents learn in market settings [1], [2], such algorithms also have practical use in distributed control [3], [4] and optimization [5].

In this paper we focus on the canonical Fictitious Play (FP) algorithm [2], [6], [7]. FP is known to achieve Nash equilibrium learning in many important games [8]–[10], but can be impractical to implement if the number of players is large [11], [12]. In particular, the per-iteration complexity of FP increases exponentially in the number of players.

One method that has been proposed to mitigate the complexity of FP is to use a Monte Carlo approach [11]—a method known as Sampled FP (SFP). In SFP players draw samples from an underlying distribution and use the sample-average utility as an estimate for the expected utility. Let $k_t$ denote the number of samples drawn in iteration $t \in \mathbb{N}$ of the SFP algorithm. A serious drawback to SFP is that, in order to ensure convergence of the algorithm, the sequence $(k_t)_{t \geq 1}$ must grow as $k_t = O(\sqrt{t})$ [11].

In this paper we revisit the SFP algorithm and show that this requirement can be relaxed. We show that convergence of the algorithm is ensured as long as $k_t$ is increasing $k_t \to \infty$, regardless of the rate. Subsequently, we consider an extension of SFP known as Single Sample FP (SSFP) [13]. Rather than using a direct Monte Carlo approach, the algorithm, SSFP uses stochastic approximation techniques to track the expected utility. Because the stochastic approximation technique uses information more efficiently than the Monte Carlo approach, SSFP is able to ensure convergence using only one sample per round (hence the name Single Sample FP).

We study both SFP and SSFP empirically. We find that, although SFP converges at low sample rates, the convergence rate of the algorithm is significantly affected by reducing the rate at which $k_t \to \infty$. On the other hand, due to the increased efficiency of information handling, SSFP tends to outperform SFP, even at high sample rates.

The remainder of the paper is organized as follows. Section II sets up notation and introduces FP. Section III presents the SFP algorithm and our accompanying convergence result. Section IV presents the SSFP algorithm and discusses convergence results. Section V discusses simulation results, and Section VI concludes the paper.

II. PRELIMINARIES

A. Notation

A game in normal form is given by the tuple $\Gamma = (\mathcal{N}, (Y_i, u_i)_{i \in \mathcal{N}})$ where $\mathcal{N} = \{1, \ldots, N\}$ denotes the set of players, $Y_i$ denotes the set of actions available to player $i$, and $u_i : Y_1 \times \cdots \times Y_N \to \mathbb{R}$ denotes the utility function of player $i$. We let $Y := Y_1 \times \cdots \times Y_N$ denote the joint action space.

Given a finite set $S$, let $\triangle(S)$ denote the set of probability distributions over $S$. Let $\Delta_i := \triangle(Y_i)$ denote the set of mixed strategies available to player $i$—note that a mixed strategy is a probability distribution over the action space of player $i$. Let $\Delta := \Delta_1 \times \cdots \times \Delta_N$ denote the set of joint mixed strategies, where it is implicitly assumed that players’ individual strategies are independent. When convenient, we represent a mixed strategy $p \in \Delta$ by $p = (p_i, \ldots, p_i)$, where $p_i$ denotes the marginal strategy of player $i$ and $p_{-i}$ is a $(n-1)$-tuple containing the marginal strategies of the other players.

In the context of mixed strategies, we often wish to retain the notion of playing a single deterministic action. For this purpose, let $\mathcal{A}_i := \{e_1, \ldots, e_m\} \subset \Delta_i$ denote the set of “pure strategies” of player $i$, where $e_j$ is the $j$-th canonical vector containing a 1 at position $j$ and zeros otherwise. Note that there is a one-to-one correspondence between a player’s action set $Y_i$ and the player’s set of pure strategies $\mathcal{A}_i \subset \Delta_i$.

Given a mixed strategy $p = (p_1, \ldots, p_N) \in \Delta$, the expected utility of player $i$ is given by

$$U_i(p) := \sum_{y \in Y} u_i(y) p_1(y_1) \cdots p_N(y_N)$$

where $U_i : \Delta \to \mathbb{R}$. Note that the mixed utility $U_i(p)$ may be interpreted as the expected utility of $u_i(y)$ given that players’ (marginal) mixed strategies $p_i$ are independent.

The set of Nash equilibria is given by $\text{NE} := \{ p \in \Delta : U_i(p, \ldots, p_{-i}) \geq U_i(p_i', \ldots, p_{-i}), \forall p_i' \in \mathcal{A}_i, \forall i \in \mathcal{N} \}$. The distance
of a distribution \( p \in \Delta \) from a set \( S \subset \Delta \) is given by \( d(p, S) = \inf \{ \| p - p' \| : p' \in S \} \). Throughout the paper \( \| \cdot \| \) denotes the standard \( L_2 \) Euclidean norm unless otherwise specified.

Throughout, we assume there exists a probability space rich enough to carry out the construction of the various random variables required in the paper.

B. Potential Games

A game is said to be an (exact) potential game [14] if there exists a function \( \phi : Y \to \mathbb{R} \) such that

\[
    u_i(y_i', y_{-i}) - u_i(y_i'', y_{-i}) = \phi(y_i', y_{-i}) - \phi(y_i'', y_{-i})
\]

for all \( y_i', y_i'' \in Y_i, y_{-i} \in Y_{-i} \), and all \( i \in \mathcal{N} \). The potential function captures the relevant information needed to determine the best response for each player. The existence of a potential function for a game \( \Gamma \) implies that \( \Gamma \) is in some sense cooperative, and that players share an underlying objective. In particular, a potential game is strategically equivalent to a game in which all players use the potential function as their utility function. The FP algorithm, described next, is proven to converge to the set of NE in potential games.

C. Fictitious Play

Assume that players repeatedly face off in a fixed game \( \Gamma \). Let \( a_i(t) \in \mathcal{A}_i \) denote the action taken by player \( i \) at time \( t \in \mathbb{N} \) and let \( a(t) = (a_1(t), \ldots, a_N(t)) \) denote the joint action. We refer to \( q(t) := \frac{1}{t} \sum_{s=1}^{t} a_i(s) \) as the empirical distribution of player \( i \) and \( q(t) := (q_1(t), \ldots, q_N(t)) \) simply as the empirical distribution.

A sequence of actions \((a(t))_{t \geq 1}\) such that for all \(^1\) \( t \geq 1 \),

\[
a_i(t + 1) = \arg \max_{a_i \in \mathcal{A}_i} U_i(a_i, q_{-i}(t)), \quad \forall i \in \mathcal{N}, \quad (1)
\]

is referred to as a fictitious play process. This describes a process where players choose their actions as a myopic best response to the current empirical distribution.

If \( \Gamma \) is a potential game, then FP is shown to learn NE strategies in the sense that \( d(q(t), \text{NE}) \rightarrow 0 \) as \( t \rightarrow \infty \) [10], [14].

D. Computational Complexity in Fictitious Play

At each stage of the FP process, each player must solve the optimization problem (1). For each player \( i \), this involves computing the expected utility of each of the actions \( a_i \in \mathcal{A}_i \) under the \( (\mathcal{N} - 1) \) dimensional probability distribution \( q_{-i}(t) := (q_j(t))_{j \in \mathcal{N}, j \neq i} \). The complexity of this computation scales exponentially in the number of players \( N \).

III. SAMPLED FP

The Sampled FP (SFP) algorithm aims to mitigate the complexity of FP by using a Monte Carlo approach. As noted in the previous section, the computationally strenuous aspect of FP is the computation of the expected utility. In SFP, the expected utility \( U_i(a_i, q_{-i}(t)), \alpha_i \in \mathcal{A}_i \) is approximated by drawing samples from the underlying probability distribution \( q_{-i}(t) \) and using the sample-average utility as a surrogate for the true expected utility in the FP algorithm (see (1)). The SFP algorithm is given below.

A. Sampled FP Algorithm

**Initialize**

(i) Each player \( i \) chooses an arbitrary initial action \( a_i(1) \in \mathcal{A}_i \).

The empirical distribution is initialized as \( q_i(1) = a_i(1), \quad \forall i \).

Fix a sample rate sequence \((k_t)_{t \geq 1}\).

**Iterate** \((t \geq 1)\)

(ii) \( k_t \) “test actions” are drawn as random samples from the joint empirical distribution \( q(t) \); let \( \hat{a}_i^*(t) \) denote the \( s \)-th random sample drawn in round \( t \). Player \( i \) estimates the utility of each of her actions \( a_i \in \mathcal{A}_i \) as

\[
    \hat{U}_i(\alpha_i, t) = \frac{1}{k_t} \sum_{s=1}^{k_t} U_i(\alpha_i, \hat{a}_i^*(t)).
\]

(iii) Each player \( i \) chooses a next-stage action that is a best response given her estimate of the mixed utility:

\[
    a_i(t + 1) = \arg \max_{a_i \in \mathcal{A}_i} \hat{U}_i(a_i, t).
\]

(iv) The empirical distribution for each player \( i \) is updated recursively to account for the action just taken: \( q_i(t + 1) = q_i(t) + \frac{1}{t + 1} (a_i(t + 1) - q_i(t)). \)

B. Discussion

In [11] it was shown that if \( k_t = O(\sqrt{t}) \) then the SFP algorithm will converge to NE in the sense that, almost surely, \( q(t) \rightarrow \text{NE} \) as \( t \rightarrow \infty \). In this section we study the question of whether SFP will still converge using a reduced sampling rate. We assume:

A. 1. \( \lim_{t \to \infty} k_t = \infty \).

Note that this assumption does not impose any conditions on the rate at which \( k_t \to \infty \). The following theorem shows that convergence of SFP is ensured under this assumption.

**Theorem 1.** Let \( \Gamma \) be a potential game. Assume A.1 holds. Then SFP converges to the set of NE with probability 1. That is, \( d(q(t), \text{NE}) \rightarrow 0 \) almost surely as \( t \to \infty \).

**Proof.** A learning process is a weakened FP process [15] if there exists a sequence \( (\epsilon_t)_{t \geq 1} \) such that \( \epsilon_t \to 0 \) and \( U_i(a_i(t + 1), q_{-i}(t)) \geq \max_{a_i \in \mathcal{A}_i} U_i(a_i, q_{-i}(t)) - \epsilon_t \) for all \( t \) in \( (1, 2, \ldots) \). In [16], [17] it is shown that in a potential game, a weakened FP process converges to the set of NE in the sense that \( d(q(t), \text{NE}) \rightarrow 0 \) as \( t \to \infty \).

By the strong law of large numbers, if \( k_t \to \infty \), then \( \left| \hat{U}_i(\alpha_i, t) - U_i(a_i, q_{-i}(t)) \right| \to 0 \) as \( t \to \infty \), almost surely. Since \( a_i(t + 1) \in \arg \max_{a_i \in \mathcal{A}_i} \hat{U}_i(\alpha_i, t) \), SFP is almost surely a weakened FP process, and convergence follows by [17] Theorem 7 and [16] Corollary 5.

This result shows that, at least theoretically, we may reduce the rate at which \( k_t \) increases and still obtain convergence of
Theorem 2. Let $\Gamma$ be a potential game and assume that A.2–A.3 hold. Then SSFP converges to the set of NE in the sense that $d(q(t), NE) \to 0$ as $t \to \infty$, almost surely.

A complete proof of this theorem is given in [13]. Let $k_t$ denote the number of samples drawn in iteration $t$ of either SSFP or SFP. In SFP we required $k_t \to \infty$ in order to ensure convergence. In contrast, in SSFP we obtain convergence using only $k_t = 1$.

Let $n_t$ denote the number of samples used to form the estimate $\hat{U}(\alpha_i, t)$ in either SFP or SSFP. In SFP the estimate $\hat{U}(\alpha_i, t)$ is formed using only the samples $(\tilde{a}_s(t))_{s=1}^t$ taken in round $t$ (see (2)). Hence, we have $n_t = k_t$.

In SSFP we have $k_t = 1$ for all $t \in \mathbb{N}$. However, in SSFP the estimate $\hat{U}(\alpha_i, t)$ is formed using a recursive algorithm that incorporates information from the current-round sample and the samples taken in all previous rounds. Hence, in SSFP we have $n_t = \sum_{s=1}^t k_s = t$.

Thus, if the SFP sample rate $k_t$ is sublinear, then the estimate of $\hat{U}(\alpha_i, t)$ in SSFP actually utilizes more sample information than the SFP estimate. In the simulations results (Section V) we see a corresponding trend—SSFP tends to outperform SFP when the SFP sample rate is sublinear.

V. Simulation Results

We simulated SFP and SSFP in a simple traffic routing scenario. We considered 30 vehicles sharing a common source and destination. Each vehicle must choose between one of 15 parallel routes.

We model this as a game $\Gamma = (\mathcal{N}, (Y_i, u_i)) \in \mathcal{N}$ in which the player set $\mathcal{N}$ is the set of vehicles and the action set $Y_i$ for each player $i$ is the set of routes. The cost for using route $r = 1, \ldots, 15$ when there are $k = 1, \ldots, 30$ vehicles on the route is given by

$$c_r(k) = \alpha_{r,2}k^2 + \alpha_{r,1}k + \alpha_{r,0},$$

where $\alpha_{r,\ell} \ell = 0, \ldots, 3$ is a randomly generated cost coefficient. For a routing choice $y \in Y = Y_1 \times \cdots \times Y_N$, let $\sigma_r(y)$ denote the number of vehicles on route $r$. The utility of player $i$ is given by $u_i(y) = -\sum_{r=1}^{15} c_r(\sigma_r(y))$, and is identical for all $i$. In other words, each player wishes to minimize the total travel time experienced by all vehicles.

Figure 1 shows the expected total travel time under $q(t)$ (i.e., $-U_i(q(t))$) for SSFP and for SFP (averaged over 5 instantiations with the same initial conditions) using a SFP sampling rate of $k_t = [t^\gamma]$ for various values of $\gamma$.

The per-iteration convergence rate of an algorithm refers to how quickly $d(q(t), NE) \to 0$ as $t \to 0$. Note that, despite drawing only one sample per round, the per-iteration convergence rate of SSFP tends to be faster than SFP for $\gamma < 1$. This is consistent with our observations in Section IV-C.

We note that a NE strategy does not necessarily minimize total travel time. However, the trend in the figure is consistent with convergence to NE and the curves are indicative of the convergence rate of each algorithm.
Let \( w(t) \) denote the wall-clock time of evaluating the first \( t \) iterations of an algorithm. Figure 2 plots \( w(t) \) for each of the respective algorithms. We note that the wall-clock evaluation time of each algorithm is proportional to the total number of samples drawn through iteration \( t \); that is, \( w(t) \propto \sum_{s=1}^{t} k_s \), where \( k_s \) is the number of samples drawn in round \( s \). In SSFP, we get \( w(t) \propto t \). In SFP we get \( w(t) \propto \sum_{s=1}^{t} s^7 \approx \frac{1}{2} t^{7+1} \). It is important to note that this means the SFP algorithm becomes progressively more difficult to evaluate as time goes on.

In addition to the per-iteration convergence rate, one may consider the per-sample convergence rate of an algorithm. For \( \gamma < 1 \), SSFP shows faster per-iteration convergence while simultaneously requiring few samples per iteration, suggesting a much quicker per-sample convergence rate.

VI. Conclusions

The FP algorithm can be impractical in large games. A central issue is the complexity of the best response computation. Sampled FP mitigates this issue using a Monte Carlo approach. Previous convergence results for Sampled FP required a per-iteration sampling rate of \( k_t = O(\sqrt{t}) \). We have shown that this condition can be relaxed and convergence of Sampled FP can be ensured using any increasing \( k_t \). We then considered a stochastic-approximation-based variant of Sampled FP, referred to as Single Sample FP, in which convergence is ensured using only one sample per round. The relative performance of the algorithms was studied empirically, and SSFP was found to outperform SFP for sublinear SFP sample rates.

REFERENCES