Policy Optimization for Markov Games: Unified Framework and Faster Convergence

Anonymous Author(s)
Affiliation
Address
email

Abstract

This paper studies policy optimization algorithms for multi-agent reinforcement learning. We begin by proposing an algorithm framework for two-player zero-sum Markov Games in the full-information setting, where each iteration consists of a policy update step at each state using a certain matrix game algorithm, and a value update step with a certain learning rate. This framework unifies many existing and new policy optimization algorithms. We show that the *state-wise average policy* of this algorithm converges to an approximate Nash equilibrium (NE) of the game, as long as the matrix game algorithms achieve low weighted regret at each state, with respect to weights determined by the speed of the value updates. Next, we show that this framework instantiated with the Optimistic Follow-The-Regularized-Leader (OFTRL) algorithm at each state (and smooth value updates) can find an $\widetilde{\mathcal{O}}(T^{-5/6})$ approximate NE in T iterations, which improves over the current best $\widetilde{\mathcal{O}}(T^{-1/2})$ rate of symmetric policy optimization type algorithms. We also extend this algorithm to multi-player general-sum Markov Games and show an $\widetilde{\mathcal{O}}(T^{-3/4})$ convergence rate to Coarse Correlated Equilibria (CCE). Finally, we provide a numerical example to verify our theory and investigate the importance of smooth value updates, and find that using "eager" value updates instead (equivalent to the independent natural policy gradient algorithm) may significantly slow down the convergence, even on a simple game with $H = 2$ layers.

1 Introduction

Policy optimization, i.e. algorithms that learn to make sequential decisions by local search on the agent's policy directly, is a widely used class of algorithms in reinforcement learning [38, 42, 43]. Policy optimization algorithms are particularly advantageous in the multi-agent reinforcement learning (MARL) setting (e.g. compared with value-based counterparts), due to their typically lower representational cost and better scalability in both training and execution. A variety of policy optimization algorithms such as Independent PPO [13], MAPPO [53], QMix [40] have been proposed to solve real-world MARL problems [4, 37, 41]. These algorithms share a same high-level structure with iterative *value updates* (for certain value estimates) and *policy updates* (often independently with each agent) using information from the value estimates and/or true rewards.

While policy optimization for MARL has been studied theoretically in a growing body of work, there are still gaps between algorithms used in practice and provably-efficient algorithms studied in theory—Algorithms in practice generally follow two natural design principles: *symmetric updates* among all agents, and *simultaneous learning* of values and policies [56, 53]. By contrast, policy optimization algorithms studied in theory often diverge from these principles and incorporate some tweaks, such as (i) asymmetric updates, where one agent takes a much smaller learning rate than the others (two time-scale) [11] or waits until the other agents learn an approximate best response [59];

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

and (ii) batch-like learning, where policies are optimized to sufficient precision with respect to the current value estimate before the next value update [8]. There is so far a lacking of systematic studies on the performance of the more vanilla policy optimization algorithms following the above two principles, even under the setting where full-information feedback from the game is available.

Towards bridging these gaps, this paper studies policy optimization algorithms for Markov games, with a focus on algorithms with symmetric updates and simultaneous learning of values and policies. Our contributions can be summarized as follows:

- We propose an algorithm framework for two-player zero-sum Markov games in the full-information setting (Section 3). This framework unifies many existing and new policy optimization algorithms such as Nash V-Learning, Gradient Descent/Ascent, as well as seemingly disparate algorithms such as Nash Q-Learning (Section 3.2). We prove that the *state-wise average policy* outputted by the above algorithm is an approximate Nash Equilibrium (NE), so long as suitable per-state weighted regrets are bounded (Section 3.1). This generic result can be instantiated in a modular fashion to derive convergence guarantees for the many examples above.
- We instantiate our framework to show that a new algorithm based on Optimistic Follow-The-Regularized-Leader (OFTRL) and smooth value updates finds an $\widetilde{\mathcal{O}}(T^{-5/6})$ approximate NE in T iterations (Section 4). This improves over the current best rate of $\widetilde{\mathcal{O}}(T^{-1/2})$ achieved by symmetric policy optimization type algorithms. We additionally extend the above OFTRL algorithm to multi-player general-sum Markov games and show an $\widetilde{\mathcal{O}}(T^{-3/4})$ convergence rate to Coarse Correlated Equilibria (CCE), which is also the first rate faster than $\widetilde{\mathcal{O}}(T^{-1/2})$ for policy optimization in general-sum Markov games (Section 4.1).
- We perform simulations on a carefully constructed zero-sum Markov game with $H = 2$ layers to verify our convergence guarantees. The numerical tests further suggest the importance of *smooth value updates*: the Independent Natural Policy Gradient algorithm (as one instantiation of our algorithm framework with "eager" value updates) appears to converge much slower (Section 5).

1.1 Related work

Two-player zero-sum MGs Markov games (MGs) [28] (also known as Stochastic Games [45]) is a widely studied model for multi-agent reinforcement learning. In the most basic setting of two-player zero-sum MGs, algorithms for computing the NE have been extensively studied in both the full-information setting [29, 19, 18] and the sample-based/online setting [6, 49, 21, 46, 55, 2, 52, 3, 31, 24, 20, 54, 10, 32]. Our algorithm framework incorporates (the full-information version of) several algorithms in this line of work.

Policy optimization for zero-sum MGs Policy optimization for single-agent Markov Decision Processes has been extensively in a recent line of work, e.g. [1, 5, 30, 44, 35, 33, 7, 14, 51] and the many references therein. For two-player zero-sum MGs, the Nash V-Learning algorithm of Bai et al. [3] (originally proposed for the sample-based online setting) can be viewed as an independent policy optimization algorithm, and can be adapted to the full-information setting with $\widetilde{\mathcal{O}}(T^{-1/2})$ convergence rate. Daskalakis et al. [11] prove that the independent policy gradient algorithm with an asymmetric two time-scale learning rate can learn the NE (for one player only) with polynomial iteration/sample complexity. Zhao et al. [59] show that another asymmetric algorithm that simulates a policy gradient/best response dynamics converges to NE with $\widetilde{\mathcal{O}}(1/T)$ rate. Cen et al. [8] use a symmetric optimistic (extragradient) subroutine for matrix games to learn zero-sum MGs in a *layer-wise* fashion (more like a Value Iteration type algorithm), and also derive an $\widetilde{\mathcal{O}}(1/T)$ convergence rate. The closest to our work is Wei et al. [50] which proves that Optimistic Gradient Descent/Ascent (OGDA), combined with smooth value updates at all layers simultaneously, converges to an NE with $\widetilde{\mathcal{O}}(T^{-1/2})$ rate for both the average duality gap and the last iterate. The fast $\widetilde{\mathcal{O}}(T^{-5/6})$ rate of our OFTRL algorithm improves over [50] and is the first such faster rate for symmetric, policy optimization type algorithms.

Multi-player general-sum MGs A recent line of work shows that a generalization of the V-learning algorithm to the multi-player general-sum setting can learn Coarse Correlated Equilibria (CCE) [47, 23, 34] and Correlated Equilibria (CE) [47, 23]. The algorithmic designs in these works

are specially tailored to the sample-based setting, where the best possible rate is $\widetilde{\mathcal{O}}(T^{-1/2})$.[1] In contrast, this paper considers the full-information setting and proposes new algorithms achieving the faster $\widetilde{\mathcal{O}}(T^{-3/4})$ for learning CCE in general-sum MGs. Another recent line of work considers learning NE in Markov Potential Games [57, 27, 47, 15, 58], which can be seen as a cooperative-type subclass of general-sum MGs.

Optimistic algorithms in normal-form games Technically, our accelerated rates build on the recent line of work on faster rates for optimistic no-regret algorithms in normal-form games [48, 39, 9, 12]. Specifically, our $\tilde{\mathcal{O}}(T^{-5/6})$ rate for two-player zero-sum MGs builds upon a first-order smoothness analysis of [9], and our $\tilde{\mathcal{O}}(T^{-3/4})$ rate for multi-player general-sum MGs follows from the RVU-property [48, Definition 3]. Our incorporation of these techniques involves non-trivial new components such as *weighted* first-order smoothness bounds and handling changing game rewards.

2 Preliminaries

We consider the tabular episodic (finite-horizon) two-player-zero-sum Markov games (MGs), which can be denoted as $\mathcal{M}(H, \mathcal{S}, \mathcal{A}, \mathcal{B}, \mathbb{P}, r)$, where H is the horizon length; $\mathcal{S}$ is the state space with $|\mathcal{S}| = S$; $\mathcal{A}, \mathcal{B}$ are the action space of the *max-player* and *min-player* respectively, with $|\mathcal{A}| = A, |\mathcal{B}| = B$; $\mathbb{P} = \{\mathbb{P}_h\}_{h=1}^H$ is the transition probabilities, where each $\mathbb{P}_h(s'|s,a,b)$ gives the probability of transition to state s' from state-action (s,a,b); $r = \{r_h\}_{h=1}^H$ are the reward functions, such that $r_h(s,a,b)$ is reward[2] of the max-player and $-r_h(s,a,b)$ is the reward of the min-player at time step h and state-action (s,a,b). In each episode, the MG starts with a deterministic initial state s_1. Then at each time step $1 \le h \le H$, both players observes the state s_h, the max-player takes an action $a_h \in \mathcal{A}$, and the min-player takes an action $b_h \in \mathcal{B}$. Then, both players receive their rewards $r_h(s_h, a_h, b_h)$ and $-r_h(s_h, a_h, b_h)$, respectively, and the system transits to the next state $s_{t+1} \sim \mathbb{P}_h(\cdot|s_h, a_h, b_h)$.

Policies & value functions A (Markov) policy μ of the max-player is a collection of policies $\mu = \{\mu_h : \mathcal{S} \to \Delta_{\mathcal{A}}\}_{h=1}^H$, where each $\mu_h(\cdot|s_h) \in \Delta_{\mathcal{A}}$ specifies the probability of taking action a_h at (h, s_h). Similarly, a (Markov) policy ν of the min-player is defined as $\nu = \{\nu_h : \mathcal{S} \to \Delta_{\mathcal{B}}\}$. For any policy (μ, ν) (not necessarily Markov), we use $V_h^{\mu,\nu} : \mathcal{S} \to \mathbb{R}$ and $Q_h^{\mu,\nu} : \mathcal{S} \times \mathcal{A} \times \mathcal{B} \to \mathbb{R}$ to denote the value function and Q-function at time step h, respectively, i.e.

$$V_h^{\mu,\nu}(s) := \mathbb{E}_{\mu,\nu}\left[\textstyle\sum_{h=h'}^H r_{h'}(s_{h'}, a_{h'}, b_{h'}) \mid s_h = s\right], \tag{1}$$

$$Q_h^{\mu,\nu}(s,a,b) := \mathbb{E}_{\mu,\nu}\left[\textstyle\sum_{h=h'}^H r_{h'}(s_{h'}, a_{h'}, b_{h'}) \mid s_h = s, a_h = a, b_h = b\right]. \tag{2}$$

For notational simplicity, we use the following abbreviation: $[\mathbb{P}_h V](s,a,b) := \mathbb{E}_{s' \sim \mathbb{P}_h(\cdot|s,a,b)} V(s')$ for any value function V. By definition of the value functions and Q-functions, we have the following Bellman equations

$$\begin{aligned} Q_h^{\mu,\nu}(s,a,b) &= \left(r_h + \mathbb{P}_h V_{h+1}^{\mu,\nu}\right)(s,a,b), \\ V_h^{\mu,\nu}(s,a,b) &= \mathbb{E}_{a \sim \mu_h(\cdot|s), b \sim \nu_h(\cdot|s)}[Q_h^{\mu,\nu}(s,a,b)] = \langle Q_h^{\mu,\nu}(s,\cdot,\cdot), \mu(\cdot|s) \times \nu(\cdot|s) \rangle . \end{aligned}$$

The goal for the max-player is to maximize the value function, whereas the goal for the min-player is to minimize the value function.

Best response & Nash equilibrium For any Markov policy μ of a max-player, there exists a best response for the min-player, which can be taken as a Markov policy $\nu^\dagger(\mu)$ such that $V_h^{\mu,\nu^\dagger(\mu)}(s) = \inf_\nu V_h^{\mu,\nu}(s)$ for all $(s,h) \in \mathcal{S} \times [H]$. For simplicity we define $V_h^{\mu,\dagger} := V_h^{\mu,\nu^\dagger(\mu)}$. By symmetry, we can also define $\mu^\dagger(\nu)$ and $V_h^{\dagger,\nu}$. It is known (e.g. [16]) that there exist Markov policies $(\mu^\star, \nu^\star)$ that perform optimally against best responses. These policies are also equivalent to Nash Equilibria (NEs) of the game, where no player can gain by switching to a different policy unilaterally. It can also be shown that any NE $(\mu^\star, \nu^\star)$ satisfies the following minimax equation

$$\sup_\mu \inf_\nu V_h^{\mu,\nu}(s) = V_h^{\mu^\star,\nu^\star}(s) = \inf_\nu \sup_\mu V_h^{\mu,\nu}(s).$$

[1] Even if we specialize their algorithms to the full-information setting, the attained rates are no better than $\widetilde{\mathcal{O}}(T^{-1/2})$ because the bandit subroutines they deployed in V-learning converge no faster than $\Omega(T^{-1/2})$ even in the simplest setting of full-information matrix games.

[2] This assumes deterministic rewards; our results can be generalized directly to the case of stochastic rewards.

Algorithm 1 Algorithm framework for two-player zero-sum Markov Games

1: **Require:** Learning rate $\{\beta_t\}_{t\geq 1} \subset [0,1]$ (with $\beta_1 = 1$); Algorithm MatrixGameAlg.
2: **Initialize:** $Q_h^0(s,a,b) \leftarrow H - h + 1$ for all $(h,s,a,b) \in [H] \times \mathcal{S} \times \mathcal{A} \times \mathcal{B}$.
3: **for** $t = 1, \ldots, T$ **do**
4: **for** $h = H, \ldots, 1$ **do**
5: **Policy update:** Update policies for all $s \in \mathcal{S}$:

$$(\mu_h^t(\cdot|s), \nu_h^t(\cdot|s)) \leftarrow \mathsf{MatrixGameAlg}\Big(\big\{Q_h^i(s,\cdot,\cdot)\big\}_{i=1}^{t-1}, \big\{\mu_h^i(\cdot|s)\big\}_{i=1}^{t-1}, \big\{\nu_h^i(\cdot|s)\big\}_{i=1}^{t-1}\Big).$$

6: **Value update:** Update Q-value for all $(s,a,b) \in \mathcal{S} \times \mathcal{A} \times \mathcal{B}$:

$$Q_h^t(s,a,b) \leftarrow (1-\beta_t) Q_h^{t-1}(s,a,b) + \beta_t\big(r_h + \mathbb{P}_h[(\mu_{h+1}^t)^\top Q_{h+1}^t \nu_{h+1}^t]\big)(s,a,b). \quad (3)$$

7: **Output**: State-wise average policy $(\widehat{\mu}^T, \widehat{\nu}^T)$, with β_T^t defined in (6):

$$\widehat{\mu}_h^T(\cdot|s) \leftarrow \textstyle\sum_{t=1}^T \beta_T^t \mu_h^t(\cdot|s), \quad \widehat{\nu}_h^T(\cdot|s) \leftarrow \sum_{t=1}^T \beta_T^t \nu_h^t(\cdot|s) \quad \text{for all } (h,s) \in [H] \times \mathcal{S}. \quad (4)$$

Thus, while the NE policy $(\mu^\star, \nu^\star)$ may not be unique, all of them share the same value functions, which we denote as $V_h^\star := V_h^{\mu^\star,\nu^\star}$. The Q-function $Q_h^\star$ can be defined similarly. In this paper, our main goal is to find an approximate NE, which is formally defined below.

Definition 1 (ε-approximate Nash Equilibrium)**.** *For any $\varepsilon \geq 0$, a policy (μ,ν) is an ε-approximate Nash Equilibrium (ε-NE) if* $\mathrm{NEGap}(\mu,\nu) := V_1^{\dagger,\nu}(s_1) - V_1^{\mu,\dagger}(s_1) \leq \varepsilon$.

Additional notation For any $(h,s) \in [H] \times \mathcal{S}$ and Q function $Q_h : \mathcal{S} \times \mathcal{A} \times \mathcal{B} \to \mathbb{R}$, we define shorthand $[(\mu_h)^\top Q_h \nu_h](s) := \langle Q_h(s,\cdot,\cdot), \mu_h(\cdot|s) \times \nu_h(\cdot|s)\rangle$ for any policy (μ,ν). Similarly, we let $[Q_h \nu_h](s,\cdot) := \mathbb{E}_{b\sim\nu_h(\cdot|s)}[Q_h(s,\cdot,b)] \in \mathbb{R}^A$, and $[Q_h^\top \mu_h](s,\cdot) := \mathbb{E}_{a\sim\mu_h(\cdot|s)}[Q_h(s,a,\cdot)] \in \mathbb{R}^B$. We use $A \vee B := \max\{A,B\}$.

3 An algorithm framework for zero-sum Markov games

We begin by presenting an algorithm framework that unifies many existing and new algorithms for two-player zero-sum Markov Games, and its performance guarantee that could be specialized to yield concrete convergence results for many specific algorithms.

Our algorithm framework, described in Algorithm 1, consists of two main components: the *policy update* step computing policies (μ^t, ν^t), and the *value update* step computing the Q estimate Q_h^t's.

Policy update via matrix game algorithms In the policy update step (Line 5), for each (h,s), the two players update policies $(\mu_h^t(\cdot|s), \nu_h^t(\cdot|s))$ at (h,s) using some matrix game algorithm MatrixGameAlg which takes as input all past Q matrices and all past policies of both players. The MatrixGameAlg offers a flexible interface that allows many choices such as the matrix NE subroutine over the most recent Q matrix $\mathsf{MatrixNE}(Q_h^{t-1}(s,\cdot,\cdot))$, or any independent no-regret algorithm (for both players), such as Follow-The-Regularized-Leader (FTRL) (9) or projected Gradient Descent-Ascent (10) considered in the examples later.

Value update with learning rate $\{\beta_t\}$ For any (h,s,a,b), the value update step (Line 6) updates $Q_h^t(s,a,b)$ by the newest value function $r_h + \mathbb{P}_h[(\mu_{h+1}^t)^\top Q_{h+1}^t \nu_{h+1}^t]$ propagated from layer $h+1$, using a sequence of learning rates $\{\beta_t\}_{t\geq 1}$ which we assume to be within $[0,1]$ (with $\beta_1 := 1$). $\{\beta_t\}$ controls the speed of the value update, with two important special cases:

(1) *Eager* value updates, where we set $\beta_t = 1$ so that Q_h^t performs *policy evaluation* of the current policy (μ^t, ν^t), that is, $Q_h^t = Q_h^{\mu^t,\nu^t}$.

(2) *Smooth* (incremental) value updates, where we choose $\beta_t \to 0$ as $t \to \infty$. In this case, the Q_h^t moves slower (resembling a *critic* in Actor-Critic like algorithms), and becomes a weighted average of all past updates. A standard choice that is frequently used is from [22] (and many subsequent work),

$$\beta_t = \alpha_t := (H+1)/(H+t). \quad (5)$$

For any $\{\beta_t\}$, the update (3) implies that

$$Q_h^t(s,a,b) = \sum_{i=1}^t \beta_t^i \Big(\Big[r_h + \mathbb{P}_h[(\mu_{h+1}^i)^\top Q_{h+1}^i \nu_{h+1}^i] \Big](s,a,b) \Big),$$

where β_t^i's are a group of weights summing to one ($\sum_{i=1}^t \beta_t^i = 1$) defined as

$$\beta_t^t = \beta_t; \qquad \beta_t^i = \prod_{j=i+1}^t (1-\beta_j)\beta_i, \ \text{ for } \ i \in [t-1]. \tag{6}$$

Note that with smooth value updates ($\beta_t < 1$), Q_h^t is not necessarily the Q-function of any policy. Upon finishing, the algorithm outputs the *state-wise average policy* $(\widehat{\mu}^T, \widehat{\nu}^T)$ defined in (4), where each $\widehat{\mu}_h^T(\cdot|s)$ is the weighted average of $\mu_h^t(\cdot|s)$ using weights $\{\beta_T^t\}_{t=1}^T$ (and similarly for $\widehat{\nu}^T$).

Symmetric & simultaneous learning, (de)centralization We remark that Algorithm 1 by definition performs simultaneous learning (of policies and values) at all layers, and also yields symmetric (policy) updates if MatrixGameAlg is a symmetric algorithm with respect to μ and ν. Also, although Algorithm 1 appears to be a *centralized* algorithm as it maintains Q values in (3), this does not preclude possibilities that the algorithm can be executed in a *decentralized* fashion. This can happen e.g. when the Q-update (3) can be rewritten as an equivalent V-update (cf. Example 1 & 2).

3.1 Theoretical guarantee

We are now ready to state the main theoretical guarantee of Algorithm 1, which states that the $(\widehat{\mu}^T, \widehat{\nu}^T)$ is an approximate NE, as long as the algorithm achieves low *per-state weighted regrets* w.r.t. weights $\{\beta_t^i\}_{i=1}^t$, defined as

$$\begin{aligned}
&\mathrm{reg}_{h,\mu}^t(s) := \max_{\mu^\dagger \in \Delta_{\mathcal{A}}} \textstyle\sum_{i=1}^t \beta_t^i \left\langle \mu^\dagger - \mu_h^i(\cdot|s), [Q_h^i \nu_h^i](s,\cdot) \right\rangle, \\
&\mathrm{reg}_{h,\nu}^t(s) := \max_{\nu^\dagger \in \Delta_{\mathcal{B}}} \textstyle\sum_{i=1}^t \beta_t^i \left\langle \nu_h^i(\cdot|s) - \nu^\dagger, [(Q_h^i)^\top \mu_h^i](s,\cdot) \right\rangle, \\
&\mathrm{reg}_h^t := \max_{s \in \mathcal{S}} \max\{\mathrm{reg}_{h,\mu}^t(s), \mathrm{reg}_{h,\nu}^t(s)\}.
\end{aligned}$$

Theorem 2 (Main guarantee of Algorithm 1)**.** *Suppose that the per-state regrets can be upper-bounded as* $\mathrm{reg}_h^t \le \overline{\mathrm{reg}}_h^t$ *for all* $(h,t) \in [H] \times [T]$*, where* $\overline{\mathrm{reg}}_h^t$ *is non-increasing in* t*:* $\overline{\mathrm{reg}}_h^t \ge \overline{\mathrm{reg}}_h^{t+1}$ *for all* $t \ge 1$*. Then, the output policy* $(\widehat{\mu}^T, \widehat{\nu}^T)$ *of Algorithm 1 satisfies*

$$\mathrm{NEGap}(\widehat{\mu}^T, \widehat{\nu}^T) \le C \left[H \max_{h \in [H]} \overline{\mathrm{reg}}_h^T + H^2 c_\beta^H \log T \cdot \frac{1}{T} \sum_{t=1}^T \max_{h \in [H]} \overline{\mathrm{reg}}_h^t \right] \tag{7}$$

for all $T \ge 2$ *and some absolute constant* $C > 0$*, where* c_β *is a constant depending on* $\{\beta_t\}_{t \ge 1}$*:*

$$c_\beta := \sup_{j \ge 1} \textstyle\sum_{t=j}^\infty \beta_t^j \ge 1. \tag{8}$$

Specifically, $c_\beta = \left(1 + \frac{1}{H}\right)$ *if* $\beta_t = \alpha_t = \frac{H+1}{H+t}$*, and* $c_\beta = 1$ *if* $\beta_t = 1$*.*

Bound (7) is typically dominated by the second term on the right hand side, suggesting that the NEGap can be bounded by the average weighted regret $\widetilde{\mathcal{O}}\Big(\frac{1}{T}\sum_{t=1}^T \max_h \overline{\mathrm{reg}}_h^t\Big)$, if $c_\beta^H = O(1)$. Theorem 2 serves as a modular tool for analyzing a broad class of algorithms: As long as this average regret is sublinear in T (including—but not limited to—choosing MatrixGameAlg as uncoupled no-regret algorithms), the output policy will be an approximate NE. We emphasize though that this result is not yet end-to-end, as each reg_h^t is a weighted regret w.r.t. *the particular set of weights* $\{\beta_t^i\}_{i=1}^t$, minimizing which may require careful algorithm designs and/or case-by-case analyses. We provide some concrete examples in Section 3.2 to demonstrate the usefulness of Theorem 2.

We remark that the state-wise average policy considered in Theorem 2 is an average policy that is also *Markovian* by definition, which is different from existing work which considers either the (Markovian) last iterate [50] or non-Markovian average policies (e.g. [3]). However, this guarantee relies on full-information feedback (so that per-state regret bounds are available), and it remains an open question how such guarantees could be generalized to sample-based settings.

Proof overview The proof of Theorem 2 follows by (1) bounding $\mathrm{NEGap}(\widehat{\mu}^T, \widehat{\nu}^T)$ in terms of per-state regrets w.r.t. the *Nash value functions* $Q_h^\star$'s by performance difference arguments (Lemma C.1); (2) recursively bounding the value estimation error $\delta_h^t := \|Q_h^t - Q_h^\star\|_\infty$ (Lemma C.2) which yields the constant c_β; and (3) combining the above to translate the regret from $Q_h^\star$'s to Q_h^t's (which we assume to be bounded by $\overline{\mathrm{reg}}_h^t$) and obtain the theorem. The full proof can be found in Appendix C.

3.2 Examples

We now demonstrate the generality of Algorithm 1 and Theorem 2 by showing that they subsume many existing algorithms (and yield new algorithms) for two-player-zero-sum Markov games, and provide new guarantees with the particular output policy (4).

Example 1 (Nash V-Learning [3], full-information version)**:** The full algorithm (Algorithm 5) can be found in Appendix D.1. The algorithm is a special case of Algorithm 1 with $\beta_t = \alpha_t = (H+1)/(H+t)$, and MatrixGameAlg chosen as the weighted FTRL algorithm

$$\mu_h^t(a|s) \propto_a \exp\left(\frac{\eta}{w_{t-1}}\sum_{i=1}^{t-1} w_i \left[Q_h^i \nu_h^i\right](s,a)\right),\ \nu_h^t(b|s) \propto_b \exp\left(-\frac{\eta}{w_{t-1}}\sum_{i=1}^{t-1} w_i \left[\left(Q_h^i\right)^\top \mu_h^i\right](s,b)\right), \quad (9)$$

where $w_t := \alpha_t^t/\alpha_t^1$. Combining Theorem 2 with the standard regret bound of weighted FTRL, this algorithms achieves $\mathrm{NEGap}(\widehat{\mu}^T, \widehat{\nu}^T) \le \widetilde{\mathcal{O}}(H^{7/2}/\sqrt{T})$ choosing $\eta \asymp 1/\sqrt{T}$ (Proposition D.2).

Additionally, although the original Nash V-learning algorithm [3] updates the V values (which makes the algorithm implementable in a decentralized fashion) instead of the Q values used in Algorithm 1, these two forms are actually equivalent in the full-information setting (Proposition D.1). $\diamondsuit$

Example 2 (GDA-Critic)**:** This algorithm is a special case of Algorithm 1 with $\beta_t = \alpha_t = (H+1)/(H+t)$, and MatrixGameAlg as projected gradient descent/ascent (GDA), i.e.,

$$\mu_h^t(\cdot|s) \leftarrow \mathcal{P}_{\Delta_\mathcal{A}}\big(\mu_h^{t-1}(\cdot|s) + \eta\big[Q_h^{t-1}\nu_h^{t-1}\big](s)\big),\ \ \nu_h^t(\cdot|s) \leftarrow \mathcal{P}_{\Delta_\mathcal{B}}\big(\nu_h^{t-1}(\cdot|s) - \eta\big(\big[Q_h^{t-1}\big)^\top \mu_h^{t-1}\big](s)\big). \quad (10)$$

Similar as Nash V-Learning, GDA-Critic also admits an equivalent form with V value updates (full description in Algorithm 6). As GDA achieves weighted regret bounds with any monotone weights including $\{\alpha_t^i\}_{i=1}^t$ (Lemma B.1), we can invoke Theorem 2 to show that this algorithm achieves $\mathrm{NEGap}(\widehat{\mu}^T, \widehat{\nu}^T) \le \widetilde{\mathcal{O}}(H^{7/2}(A \vee B)^{1/2}/\sqrt{T})$ if we choose $\eta \asymp 1/\sqrt{T}$ (Proposition D.4).

The GDA-critic algorithm is also similar to the OGDA-MG algorithm of Wei et al. [50], except that we use the (non-optimistic) vanilla version of GDA. To our best knowledge, the above algorithm and guarantee are not known. We remark that even ignoring difference between GDA and OGDA, the above guarantee cannot be obtained by direct adaptation of the results of [50] which focus on either the average duality gap and/or last-iterate convergence. $\diamondsuit$

Besides the above examples, Algorithm 1 also incorporates the following algorithms which are typically not categorized as policy optimization algorithms.

Example 3 (Nash Q-Learning [19, 3], full-information version)**:** This algorithm is a special case of Algorithm 1 with $\beta_t = \alpha_t = (H+1)/(H+t)$ and MatrixGameAlg as the matrix Nash subroutine

$$(\mu_h^t(\cdot|s), \nu_h^t(\cdot|s)) \leftarrow \mathsf{MatrixNE}(Q_h^{t-1}(s,\cdot,\cdot)) := \arg\left(\min_{\mu\in\Delta_\mathcal{A}} \max_{\nu\in\Delta_\mathcal{B}} \mu^\top Q_h^{t-1}(s,\cdot,\cdot)\nu\right).$$

(Full description in Algorithm 7.) Although $\mathsf{MatrixNE}(Q_h^{t-1}(s,\cdot,\cdot))$ is not by default a no-regret algorithm, using the fact that $\left\|Q_h^t - Q_h^{t-1}\right\|_\infty$ is small (due to the small α_t) we can show that it is close to a (hypothetical) "Be-The-Leader" style algorithm that computes the matrix NE of the current Q matrix Q_h^t which achieves ≤ 0 regret (Lemma D.3). Combining this with Theorem 2 shows that this algorithm achieves $\mathrm{NEGap}(\widehat{\mu}^T, \widehat{\nu}^T) \le \widetilde{\mathcal{O}}(H^4/T)$ (Proposition D.5). $\diamondsuit$

Example 4 (Nash Policy Iteration (Nash-PI))**:** This classical algorithm (Algorithm 8) performs iterative policy evaluation and policy improvement (also similar to Nash Value Iteration [45, 2, 31]):

$$(\mu_h^{t+1}(\cdot|s), \nu_h^{t+1}(\cdot|s)) \leftarrow \mathsf{MatrixNE}(Q_h^{\mu^t,\nu^t}(s,\cdot,\cdot)). \quad (11)$$

This is also a special case of Algorithm 1 with $\beta_t = 1$ and MatrixGameAlg set as MatrixNE. It is a standard result that this algorithm converges exactly (achieving zero NE gap) in H steps, and this fact can be obtained using our framework as well (Proposition D.6). $\diamondsuit$

4 Fast convergence of optimistic FTRL

In this section, we instantiate Algorithm 1 by choosing MatrixGameAlg as the Optimistic Follow-The-Regularized-Leader (OFTRL) algorithm. OFTRL is also an uncoupled no-regret algorithm that is known to enjoy faster convergence than standard FTRL under additional loss smoothness assumptions [39, 48, 9, 12]. We show that, using OFTRL, Algorithm 1 enjoys faster convergence than the $\widetilde{\mathcal{O}}(1/\sqrt{T})$ rate of using FTRL or GDA (cf. Example 1 & 2).

Concretely, we use the following weighted OFTRL algorithm at each (h, s, t):

$$\begin{aligned} \mu_h^t(a|s) &\propto_a \exp\Big((\eta/w_t)\cdot\Big[\textstyle\sum_{i=1}^{t-1} w_i(Q_h^i\nu_h^i)(s,a) + w_{t-1}(Q_h^{t-1}\nu_h^{t-1})(s,a)\Big]\Big), \\ \nu_h^t(b|s) &\propto_b \exp\Big(-(\eta/w_t)\cdot\Big[\textstyle\sum_{i=1}^{t-1} w_i((Q_h^i)^\top\mu_h^i)(s,b) + w_{t-1}((Q_h^{t-1})^\top\mu_h^{t-1})(s,b)\Big]\Big), \end{aligned} \tag{12}$$

where w_t is the same weights as defined in Example 1, and we choose $\beta_t = \alpha_t = (H+1)/(H+t)$.

Theorem 3 (Fast convergence of OFTRL in zero-sum Markov Games)**.** *Suppose Algorithm 1 is instantiated with $\beta_t = \alpha_t = (H+1)/(H+t)$ and* MatrixGameAlg *to be the OFTRL algorithm* (12) *with any $\eta \le 1/H$ (full description in Algorithm 9). Then the per-state regret can be bounded as follows for some absolute constant $C > 0$:*

$$\mathrm{reg}_h^t \le \overline{\mathrm{reg}}_h^t := C\left[\frac{H^2\log(A\vee B)}{\eta t} + \eta^5 H^6\right] \quad \textit{for all } (h,t)\in[H]\times[T]. \tag{13}$$

Further, choosing $\eta = \mathrm{poly}(H, \log(A\vee B), \log T)\cdot T^{-1/6}$, the output (state-wise average) policy $(\widehat{\mu}^T, \widehat{\nu}^T)$ achieves approximate NE guarantee

$$\mathrm{NEGap}(\widehat{\mu}^T, \widehat{\nu}^T) \le \mathcal{O}\big(\mathrm{poly}(H, \log(A\vee B), \log T)\cdot T^{-5/6}\big). \tag{14}$$

To our best knowledge, the $\widetilde{\mathcal{O}}(T^{-5/6})$ rate asserted in Theorem 3 is the first rate faster than the standard $\widetilde{\mathcal{O}}(1/\sqrt{T})$ for symmetric, policy optimization type algorithms in two-player zero-sum Markov games. The closest existing result to this is of Wei et al. [50], who analyze the OGDA algorithm with smooth value updates and show a $\widetilde{\mathcal{O}}(1/\sqrt{T})$ convergence of both the average NEGap and the NEGap of the last-iterate. However, these only imply at most a $\widetilde{\mathcal{O}}(1/\sqrt{T})$ rate for the average policies, and not our faster rate[3]. Cen et al. [8], Zhao et al. [59] show $\widetilde{\mathcal{O}}(1/T)$ convergence of policy optimization-*like* algorithms with optimistic subroutines, which are however very different styles of algorithms that either performs *layer-wise learning* similar as Value Iteration (the matrix games at each state are learned to sufficient precision before the backup) [8], or uses strongly *asymmetric* updates that simulate a policy gradient-best response dynamics [59]. By contrast, our Algorithm 9 runs symmetric no-regret dynamics for both players, simultaneously at all layers.

Proof overview The proof of Theorem 3 (deferred to Appendix E) builds upon the recent line of work on fast convergence of optimistic algorithms [39, 48, 9], in particular the work of Chen and Peng [9] which shows an $\widetilde{\mathcal{O}}(T^{-5/6})$ convergence rate of OFTRL for two-player normal-form games. Our regret bound (13) generalizes this result non-trivially by additionally handling (1) The *weighted* regret, which requires bounding the weighted stability of the OFTRL iterates by a new analysis of the potential functions (Lemma B.4), and (2) The errors induced by *changing game matrices*, as $Q_h^t(s,\cdot,\cdot)$ changes over t. Plugging (13) into Theorem 2 yields the policy guarantee (14).

We also remark that, while OFTRL for zero-sum *matrix games* admits an improved $\widetilde{\mathcal{O}}(T^{-1})$ rate [39, 48] by an $\widetilde{\mathcal{O}}(T^{-1})$ bound on the (average) *two-sided regret*, that result is not applicable in our Markov game setting, as our analysis requires bounded *one-sided regrets* that is not implied by its corresponding two-sided version.

4.1 Extension to multi-player general-sum Markov games

Our fast convergence result can be extended to the more general setting of *multi-player general-sum* Markov games. Concretely, we consider general-sum Markov games with $m \ge 2$ players, S states,

[3] See also [17] for another example where last-iterates are *provably* slower than averages.

H steps, where the i-th player has action space $\mathcal{A}_i$ with $A_{\max} := \max_{i\in[m]} |\mathcal{A}_i|$ and her own reward function. The goal is to find a correlated policy over all players that is an approximate Coarse Correlated Equilibrium (CCE) of the game (see Appendix F.1 for the detailed setup).

We show that the OFTRL algorithm works for general-sum Markov games as well, with a fast $\widetilde{\mathcal{O}}(T^{-3/4})$ convergence to CCE. The formal statement and proof is in Theorem F.1 & Appendix F.4.

Theorem 4 (Fast convergence of OFTRL in general-sum Markov Games; Informal version of Theorem F.1)**.** *For m-player general-sum Markov Games, running the OFTRL algorithm (Algorithm 10) for T rounds, the output (correlated) policy $\widehat{\pi}$ is an ε-approximate CCE, where*

$$\varepsilon \le \mathcal{O}\big(\mathrm{poly}(H, \log A_{\max}, \log T) \cdot (m-1)^{1/2} \cdot T^{-3/4}\big).$$

A baseline result for this problem would be $\widetilde{\mathcal{O}}(T^{-1/2})$, which may be obtained directly by adapting existing proofs of the V-Learning algorithm [47, 23] to the full-information setting. Our Theorem 4 shows that a faster $\widetilde{\mathcal{O}}(T^{-3/4})$ rate is available by using the OFTRL algorithm, which to our best knowledge is the first such result for policy optimization in general-sum Markov games. We also remark that the output policy $\widehat{\pi}$ above is not a state-wise average policy as in the zero-sum setting, but rather a mixture policy that is in general non-Markov (cf. Algorithm 11), which is similar as (and slightly simpler than) the "certified policies" used in existing work [3, 47, 23]. The proof of Theorem 4 builds upon the RVU property of OFTRL [48] and additionally handles changing game rewards, similar as in Theorem 3. A proof sketch and comparison with the $\widetilde{\mathcal{O}}(T^{-5/6})$ analysis of the zero-sum case can be found in Appendix F.2.

5 Simulations

We perform numerical studies on the various policy optimization algorithms. Our goal is two-fold: (1) Verify the convergence guarantees in our theorems and examples; (2) Test some other important special cases of Algorithm 1 that may not yet admit a provable guarantee.

To this end, we consider three algorithms covered by the framework in Algorithm 1:

1. **FTRL** (Nash V-Learning) with smooth value updates $\beta_t = \alpha_t$ (Example 1 & Algorithm 5). Here the output policy $(\widehat{\mu}^T, \widehat{\nu}^T)$ are the state-wise averages with weights $\{\alpha_T^i\}_{i=1}^T$, and achieves $\mathrm{NEGap}(\widehat{\mu}^T, \widehat{\nu}^T) \lesssim T^{-1/2}$ if we choose $\eta \asymp T^{-1/2}$ (Proposition D.2).
2. **OFTRL** with smooth value updates $\beta_t = \alpha_t$ (Algorithm 9). Here the output policy $(\widehat{\mu}^T, \widehat{\nu}^T)$ are the state-wise averages with weights $\{\alpha_T^i\}_{i=1}^T$, and achieves $\mathrm{NEGap}(\widehat{\mu}^T, \widehat{\nu}^T) \lesssim T^{-5/6}$ if we choose $\eta \asymp T^{-1/6}$ (Theorem 3). We also consider the more aggressive choice $\eta = 1$.
3. **INPG** (Independent Natural Policy Gradients). This algorithm is an instantiation of Algorithm 1 (cf. Appendix G.3 for formal justifications) with *eager* value updates ($\beta_t = 1$), and MatrixGameAlg chosen as standard unweighted FTRL (a.k.a. Hedge) for all (h, s, t):

$$\mu_h^t(a|s) \propto_a \mu_h^{t-1}(a|s) \exp\big(\eta\big[Q_h^{t-1}\nu_h^{t-1}\big](s)\big), \;\; \nu_h^t(b|s) \propto_b \nu_h^{t-1}(b|s) \exp\Big(-\eta\Big[\big(Q_h^{t-1}\big)^\top \mu_h^{t-1}\Big](s)\Big).$$

   For this algorithm, we choose two standard learning rates: $\eta = 1$, and $\eta = T^{-1/2}$, and use the *vanilla (state-wise) average* as the output policies (since the last-iterate is known to be cyclic):

$$\widehat{\mu}_h^T(\cdot|s) = \tfrac{1}{T}\textstyle\sum_{t=1}^T \mu_h^t(\cdot|s), \;\; \widehat{\nu}_h^T(\cdot|s) = \tfrac{1}{T}\textstyle\sum_{t=1}^T \nu_h^t(\cdot|s) \quad \text{for all } (h,s) \in [H] \times \mathcal{S}.$$

The main motivation for considering INPG is that it is a natural generalization of both the widely-studied NPG algorithm for single-agent RL, and the standard Hedge algorithm for zero-sum matrix games. In both cases the algorithm admits favorable convergence guarantees: NPG converges with rate $\mathcal{O}(T^{-1})$ [1, 25, 36, 7] (in both last iterate and averaging) using $\eta = O(1)$; Hedge converges with rate $\mathcal{O}(T^{-1/2})$ in zero-sum matrix games (e.g. [39]) using $\eta \asymp T^{-1/2}$. However, to our best knowledge, the convergence of INPG for zero-sum Markov games is unclear, and it is commented by Wei et al. [50, Section 5] that eager value updates ($\beta_t = 1$) could cause the value function of the $(h+1)$th layer to oscillate, which make learning unstable or even biased within the h-th layer.

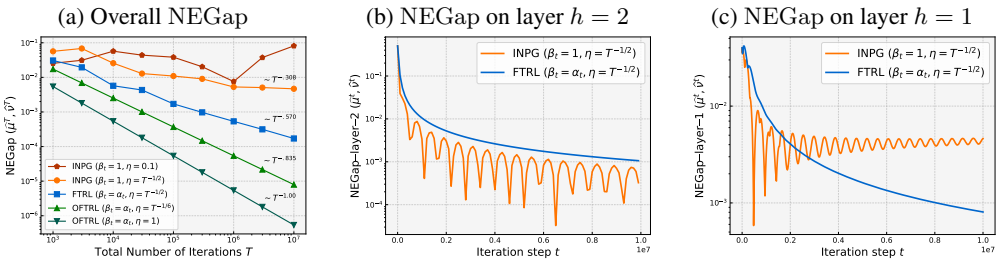

Figure 1: **(a)** NEGap of the final output policies (y-axis) against total # iterations T (x-axis) on the two-layer example (cf. Appendix G.1) in log-log scale. **Each dot represents a different run with its own** (T, η). The scalings of the form $\sim T^{-\alpha}$ are obtained via best linear fits in the log space. **(b,c)** Layer-wise NEGaps (y-axis, log-scale) against iteration count t (x-axis) for {INPG, FTRL} on a **single run** with $T = 10^7$ and $\eta = T^{-1/2}$.

A two-layer numerical example We design a simple zero-sum Markov game with two layers and small state/action spaces ($H = 2$, $S = 4$, $A = 2$; see Appendix G.1 for the detailed description). The main feature of this game is that the reward in the first layer is much lower magnitude than that of the second layer (the scale is roughly $|r_1(s, \cdot, \cdot)| \approx 0.1|r_2(s, \cdot, \cdot)|$), which may exaggerate the aforementioned unstable effect. We also choose a careful initialization (μ^1, ν^1) which is non-uniform (and modify the FTRL / OFTRL algorithms to start at this initialization, cf. Appendix G.1) but with all entries bounded in $[0.15, 0.85]$. We test all three algorithms above on this game, with this initialization, $T \in \{10^3, 3 \times 10^3, 10^4, \ldots, 10^7\}$, and η chosen correspondingly as described above.

Results Figure 1a plots the NEGap of the final output policies, one for each {algorithm, (T, η)}. Observe that FTRL converges with rate roughly $T^{-.570} \lesssim T^{-1/2}$, and OFTRL with $\eta = T^{-1/6}$ converges with rate $T^{-.835} \approx T^{-5/6}$, both corroborating our theory. Further, OFTRL with $\eta = 1$ appears to converge with rate T^{-1}; showing this may be an interesting open theoretical question.

On the other hand, the INPG algorithm appears to be much slower: The $\eta = 1$ version does not seem to converge, whereas the convergence of $\eta = T^{-1/2}$ version is not clear but at least substantially slower than $T^{-1/2}$ ($T^{-.308}$ given by the linear fit) .

To further understand the behavior of INPG, we visualize its *layer-wise* NEGap's for $h \in \{1, 2\}$ (on our example), defined as the NEGap of the h-th layer's policies with respect to $Q_h^\star$:

$$\text{NEGap-Layer-}h(\mu, \nu) := \max_s \left(\max_{\mu_h^\dagger} \left[(\mu_h^\dagger)^\top Q_h^\star \nu_h \right](s) - \min_{\nu_h^\dagger} \left[\mu_h^\top Q_h^\star \nu_h^\dagger \right](s) \right), \; h = 1, 2.$$

Note that NEGap-Layer-1 is a lower bound of $\text{NEGap}(\mu, \nu)$ (cf. Appendix G.3) and thus needs to be minimized by any convergent algorithm. By contrast, on our example, NEGap-Layer-2 is concerned with the last layer only, and can be minimized by any algorithm that works on matrix games.

Figure 1b & 1c plot the layer-wise NEGap's of INPG against FTRL, on the single run with $T = 10^7$ and $\eta = T^{-1/2}$. As expected, the NEGap-Layer-2 converges nicely for both algorithms with similar rates (Figure 1b) albeit the oscillation of INPG, whereas their behavior on NEGap-Layer-1 is drastically different: FTRL still converges, whereas INPG seems to be oscillating around a non-zero bias (Figure 1c). This suggests that INPG may indeed be suffer from a non-vanishing bias in the first layer caused by the second layer's learning dynamics. (See Appendix G.2 for additional illustrations.) It would be an interesting open question to investigate the convergence of INPG theoretically.

6 Conclusion

This paper provides a unified framework for analyzing a large class of policy optimization algorithms for two-player zero-sum Markov games. Using our framework, we prove new fast convergence rates for the OFTRL algorithm with smooth value updates: $\widetilde{\mathcal{O}}(T^{-5/6})$ for learning Nash Equilibria two-player zero-sum Markov games, and $\widetilde{\mathcal{O}}(T^{-3/4})$ for learning Coarse Correlated Equilibria in multi-player general-sum Markov games. We further demonstrate the importance of smooth value updates on a simple numerical example. We believe our work opens up many interesting directions, such as whether improved rates (e.g. $\widetilde{\mathcal{O}}(T^{-1})$) are available for OFTRL algorithms, or further investigation of policy optimization algorithms with eager value updates (such as Independent Natural Policy Gradients). Finally, a limitation of this work is its focus on the full-information setting, and it is an important open question how to generalize our anlayses to the sample-based setting.

References

[1] A. Agarwal, S. M. Kakade, J. D. Lee, and G. Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *Journal of Machine Learning Research*, 22 (98):1–76, 2021.

[2] Y. Bai and C. Jin. Provable self-play algorithms for competitive reinforcement learning. In *International conference on machine learning*, pages 551–560. PMLR, 2020.

[3] Y. Bai, C. Jin, and T. Yu. Near-optimal reinforcement learning with self-play. *Advances in neural information processing systems*, 33:2159–2170, 2020.

[4] N. Bard, J. N. Foerster, S. Chandar, N. Burch, M. Lanctot, H. F. Song, E. Parisotto, V. Dumoulin, S. Moitra, E. Hughes, et al. The hanabi challenge: A new frontier for ai research. *Artificial Intelligence*, 280:103216, 2020.

[5] J. Bhandari and D. Russo. Global optimality guarantees for policy gradient methods. *arXiv preprint arXiv:1906.01786*, 2019.

[6] R. I. Brafman and M. Tennenholtz. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3(Oct):213–231, 2002.

[7] S. Cen, C. Cheng, Y. Chen, Y. Wei, and Y. Chi. Fast global convergence of natural policy gradient methods with entropy regularization. *Operations Research*, 2021.

[8] S. Cen, Y. Wei, and Y. Chi. Fast policy extragradient methods for competitive games with entropy regularization. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 27952–27964. Curran Associates, Inc., 2021.

[9] X. Chen and B. Peng. Hedging in games: Faster convergence of external and swap regrets. *Advances in Neural Information Processing Systems*, 33:18990–18999, 2020.

[10] Z. Chen, D. Zhou, and Q. Gu. Almost optimal algorithms for two-player zero-sum linear mixture markov games. In *International Conference on Algorithmic Learning Theory*, pages 227–261. PMLR, 2022.

[11] C. Daskalakis, D. J. Foster, and N. Golowich. Independent policy gradient methods for competitive reinforcement learning. *Advances in neural information processing systems*, 33: 5527–5540, 2020.

[12] C. Daskalakis, M. Fishelson, and N. Golowich. Near-optimal no-regret learning in general games. *Advances in Neural Information Processing Systems*, 34, 2021.

[13] C. S. de Witt, T. Gupta, D. Makoviichuk, V. Makoviychuk, P. H. Torr, M. Sun, and S. Whiteson. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533*, 2020.

[14] D. Ding, K. Zhang, T. Basar, and M. Jovanovic. Natural policy gradient primal-dual method for constrained markov decision processes. *Advances in Neural Information Processing Systems*, 33:8378–8390, 2020.

[15] D. Ding, C.-Y. Wei, K. Zhang, and M. R. Jovanović. Independent policy gradient for large-scale markov potential games: Sharper rates, function approximation, and game-agnostic convergence. *arXiv preprint arXiv:2202.04129*, 2022.

[16] J. Filar and K. Vrieze. *Competitive Markov decision processes*. Springer Science & Business Media, 2012.

[17] N. Golowich, S. Pattathil, C. Daskalakis, and A. Ozdaglar. Last iterate is slower than averaged iterate in smooth convex-concave saddle point problems. In *Conference on Learning Theory*, pages 1758–1784. PMLR, 2020.

[18] T. D. Hansen, P. B. Miltersen, and U. Zwick. Strategy iteration is strongly polynomial for 2-player turn-based stochastic games with a constant discount factor. *Journal of the ACM (JACM)*, 60(1):1–16, 2013.

[19] J. Hu and M. P. Wellman. Nash q-learning for general-sum stochastic games. *Journal of machine learning research*, 4(Nov):1039–1069, 2003.

[20] B. Huang, J. D. Lee, Z. Wang, and Z. Yang. Towards general function approximation in zero-sum markov games. *arXiv preprint arXiv:2107.14702*, 2021.

[21] Z. Jia, L. F. Yang, and M. Wang. Feature-based q-learning for two-player stochastic games. *arXiv preprint arXiv:1906.00423*, 2019.

[22] C. Jin, Z. Allen-Zhu, S. Bubeck, and M. I. Jordan. Is q-learning provably efficient? *Advances in neural information processing systems*, 31, 2018.

[23] C. Jin, Q. Liu, Y. Wang, and T. Yu. V-learning–a simple, efficient, decentralized algorithm for multiagent rl. *arXiv preprint arXiv:2110.14555*, 2021.

[24] C. Jin, Q. Liu, and T. Yu. The power of exploiter: Provable multi-agent rl in large state spaces. *arXiv preprint arXiv:2106.03352*, 2021.

[25] S. Khodadadian, P. R. Jhunjhunwala, S. M. Varma, and S. T. Maguluri. On the linear convergence of natural policy gradient algorithm. *arXiv preprint arXiv:2105.01424*, 2021.

[26] T. Lattimore and C. Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

[27] S. Leonardos, W. Overman, I. Panageas, and G. Piliouras. Global convergence of multi-agent policy gradient in markov potential games. *arXiv preprint arXiv:2106.01969*, 2021.

[28] M. L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pages 157–163. Elsevier, 1994.

[29] M. L. Littman et al. Friend-or-foe q-learning in general-sum games. In *ICML*, volume 1, pages 322–328, 2001.

[30] B. Liu, Q. Cai, Z. Yang, and Z. Wang. Neural trust region/proximal policy optimization attains globally optimal policy. *Advances in neural information processing systems*, 32, 2019.

[31] Q. Liu, T. Yu, Y. Bai, and C. Jin. A sharp analysis of model-based reinforcement learning with self-play. In *International Conference on Machine Learning*, pages 7001–7010. PMLR, 2021.

[32] Q. Liu, Y. Wang, and C. Jin. Learning markov games with adversarial opponents: Efficient algorithms and fundamental limits. *arXiv preprint arXiv:2203.06803*, 2022.

[33] Y. Liu, K. Zhang, T. Basar, and W. Yin. An improved analysis of (variance-reduced) policy gradient and natural policy gradient methods. *Advances in Neural Information Processing Systems*, 33:7624–7636, 2020.

[34] W. Mao and T. Başar. Provably efficient reinforcement learning in decentralized general-sum markov games. *arXiv preprint arXiv:2110.05682*, 2021.

[35] J. Mei, C. Xiao, C. Szepesvari, and D. Schuurmans. On the global convergence rates of softmax policy gradient methods. In *International Conference on Machine Learning*, pages 6820–6829. PMLR, 2020.

[36] J. Mei, B. Dai, C. Xiao, C. Szepesvari, and D. Schuurmans. Understanding the effect of stochasticity in policy optimization. *Advances in Neural Information Processing Systems*, 34, 2021.

[37] I. Mordatch and P. Abbeel. Emergence of grounded compositional language in multi-agent populations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[38] J. Peters and S. Schaal. Policy gradient methods for robotics. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2219–2225. IEEE, 2006.

[39] S. Rakhlin and K. Sridharan. Optimization, learning, and games with predictable sequences. *Advances in Neural Information Processing Systems*, 26, 2013.

[40] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 4295–4304. PMLR, 2018.

[41] M. Samvelyan, T. Rashid, C. S. De Witt, G. Farquhar, N. Nardelli, T. G. Rudner, C.-M. Hung, P. H. Torr, J. Foerster, and S. Whiteson. The starcraft multi-agent challenge. *arXiv preprint arXiv:1902.04043*, 2019.

[42] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

[43] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[44] L. Shani, Y. Efroni, and S. Mannor. Adaptive trust region policy optimization: Global convergence and faster rates for regularized mdps. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5668–5675, 2020.

[45] L. S. Shapley. Stochastic games. *Proceedings of the national academy of sciences*, 39(10): 1095–1100, 1953.

[46] A. Sidford, M. Wang, L. Yang, and Y. Ye. Solving discounted stochastic two-player games with near-optimal time and sample complexity. In *International Conference on Artificial Intelligence and Statistics*, pages 2992–3002. PMLR, 2020.

[47] Z. Song, S. Mei, and Y. Bai. When can we learn general-sum markov games with a large number of players sample-efficiently? *arXiv preprint arXiv:2110.04184*, 2021.

[48] V. Syrgkanis, A. Agarwal, H. Luo, and R. E. Schapire. Fast convergence of regularized learning in games. *Advances in Neural Information Processing Systems*, 28, 2015.

[49] C.-Y. Wei, Y.-T. Hong, and C.-J. Lu. Online reinforcement learning in stochastic games. In *Advances in Neural Information Processing Systems*, pages 4987–4997, 2017.

[50] C.-Y. Wei, C.-W. Lee, M. Zhang, and H. Luo. Last-iterate convergence of decentralized optimistic gradient descent/ascent in infinite-horizon competitive markov games. In *Conference on Learning Theory*, pages 4259–4299. PMLR, 2021.

[51] L. Xiao. On the convergence rates of policy gradient methods. *arXiv preprint arXiv:2201.07443*, 2022.

[52] Q. Xie, Y. Chen, Z. Wang, and Z. Yang. Learning zero-sum simultaneous-move markov games using function approximation and correlated equilibrium. *arXiv preprint arXiv:2002.07066*, 2020.

[53] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. Bayen, and Y. Wu. The surprising effectiveness of ppo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955*, 2021.

[54] T. Yu, Y. Tian, J. Zhang, and S. Sra. Provably efficient algorithms for multi-objective competitive rl. *arXiv preprint arXiv:2102.03192*, 2021.

[55] K. Zhang, S. M. Kakade, T. Başar, and L. F. Yang. Model-based multi-agent rl in zero-sum markov games with near-optimal sample complexity. *arXiv preprint arXiv:2007.07461*, 2020.

[56] K. Zhang, Z. Yang, and T. Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of Reinforcement Learning and Control*, pages 321–384, 2021.

[57] R. Zhang, Z. Ren, and N. Li. Gradient play in multi-agent markov stochastic games: Stationary points and convergence. *CoRR*, abs/2106.00198, 2021.

[58] R. Zhang, J. Mei, B. Dai, D. Schuurmans, and N. Li. On the effect of log-barrier regularization in decentralized softmax gradient play in multiagent systems, 2022.

[59] Y. Zhao, Y. Tian, J. D. Lee, and S. S. Du. Provably efficient policy gradient methods for two-player zero-sum markov games. *arXiv preprint arXiv:2102.08903*, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] As this work is primarily theoretical and about understandings, we do not foresee any negative societal impacts of this work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See the supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A] Our simulations are deterministic.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]