2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Performance Bounds for Policy-Based Reinforcement Learning Methods in Zero-Sum Markov Games With Linear Function Approximation

Anna Winnicki and R. Srikant

Abstract— **Until recently, efficient policy iteration algorithms for zero-sum Markov games that converge were unknown. Therefore, model-based RL algorithms for such problems could not use policy iteration in the planning modules of the algorithms. In an earlier paper, we showed that a convergent policy iteration algorithm can be obtained by using a commonly used technique in RL called lookahead. However, the algorithm could be applied to the function approximation setting only in the special case of linear MDPs (Markov Decision Processes). In this paper, we obtain performance bounds for policy-based RL algorithms for general settings, including one where policy evaluation is performed using noisy samples of (state, action, reward) triplets from a single sample path of a given policy.**

I. Introduction

Many modern machine learning paradigms can be viewed as controlling multi-agent systems, including games [24], [25], [15], traffic control [34], [22], and more [16], [36], [35]. In many of these multi-agent systems, the problem of interest is finding a Nash equilibrium strategy. In the setting of Markov Games (also known as Stochastic Games [23]), the problem of finding the Nash equilibrium is a generalization of finding an optimal policy for a Markov Decision process [21], [13], however many algorithms that find optimal policies for MDPs cannot efficiently be extended to the Markov Games setting, largely due to monotonicity issues that arise from the competing objectives of the two players as opposed to the single entity setting of MDPs.

This issue, coupled with the added layer of complexity from incorporating function approximation in large scale systems for which it is unrealistic to employ tabular methods makes the problem of finding efficient Nash equilibrium strategies for Markov Games with function approximation a largely unsolved, yet practical, problem. We study the problem of obtaining a Nash equilibrium for infinite horizon, discounted, two-player zero-sum stochastic games with linear value function approximation.

In the single-player setting, linear value function approximation has been successfully incorporated into several approximate dynamic programming (ADP) algorithms including approximate policy iteration, which has been studied in [2], [29]. However, even in the absence of function approximation, the policy iteration algorithm has not yet been efficiently extended to the setting of Markov Games.

Unlike the widely studied analogous value iteration (VI) algorithm for games in [23], the two main extensions of Howard's Policy Iteration (PI) [21] for games are not very computationally efficient or simply do not converge. The first algorithm, the Hoffman and Karp algorithm [10] converges, however, it requires solving an MDP at each iteration. It has been shown in [9] that one has to solve $\Omega(1/(1-\alpha))$ MDPs to implement the extension of PI for games.

The second algorithm, the algorithm of Pollatschek and Avi-Itzhak also known as naive policy iteration, while computationally efficient, has been shown in a counter-example not to converge [28]. The work of [8] analyzes a modification of the Pollatschek and Avi-Itzhak algorithm. However, the work of [18] shows that their proof hinges on the assumption that the $\mathcal{L}_2$-norm of the optimal bellman residual is smooth, which is generally not true. A notable recent contribution is the work of [1] which introduces the first modification of naive policy iteration that converges. The algorithm of [1] has been extended in the work of [3] which studies stochastic and optimistic settings. It is unclear whether the algorithm in [1] can be extended to incorporate function approximation, which is the focus of the current work. The work in [31] analyzes a variant of the algorithm of Pollatshek and Avi-Itzhak where instead of greedy policies in the policy improvement step, the algorithm uses lookahead polices in the policy improvement step. Lookahead policies have been used in many popular reinforcement learning algorithms including AlphaZero [26], [27]. It is important to note that while the lookahead can be computationally expensive, the work of [31] shows that computing the lookahead can be computationally expensive. However, in the setting of linear Markov games, which are the natural extension of linear MDPs to Markov games, the work of [31] shows that the lookahead requires low computational complexity. For more on lookahead see [6], [7], [29], [30], [32].

Beyond the algorithm of [1], the aforementioned algorithms that extend policy iteration to Markov games have all attempted to extend their work to the setting of function approximation with limited success. The work of [19] extends the original policy iteration algorithm of [10], however, it is limited in two respects. First, the algorithm in [19] propagates an error bound in the policy evaluation and policy improvement steps. However, their bounds do not explicitly take into account the implementation details of least-squares-based policy evaluation. Hence, analogously to the counter-example in [29], it is unclear whether the bounds in [19] can be accurately applied in the least squares policy evaluation for games algorithm. Second, the algorithm in [19] is inefficient for the same reasons that the Hoffman-Karp algorithm is inefficient, i.e., it requires that each policy

[1]Anna Winnicki and R. Srikant are with Coordinated Science Laboratory and the Department of Electrical and Computer Engineering at University of Illinois at Urbana-Champaign. Srikant is also with the C3.ai Digital Transformation Institute. annaw5@illinois.edu, rsrikant@illinois.edu.

corresponding to the minimizer be evaluated approximately at each iteration. The work of [13] attempts to extend the algorithm of Pollatschek and Avi-Itzhak to the linear value function approximation setting, which, while efficient, does not necessarily converge even in the exact case as shown by the counter-example in [28]. Finally, the modification to the algorithm of Pollatschek and Avi-Itzhak, the algorithm in the work of [8], has been extended to the function approximation setting in [18], however, for the same reasons that algorithm also does not converge.

In this paper, we consider extensions of the algorithm in [31] to the function approximation setting. Our contributions are as follows:

- We provide performance bounds for a variant of policy iteration for Markov games based on the algorithm of Pollatschek and Avi-Itzhak with linear value function approximation. Our bounds are interpretable and in the special case where there is no function approximation, the error of the algorithm is zero. In the algorithm it is assumed that returns of the policy evaluation with the lookahead for only several states are generated exactly, and the returns for the rest of states are determined by finding a best fitting parameter.
- We then consider the case where policy evaluation is performed via simulations resulting in an unbiased error. We employ a stochastic approximation algorithm that ensures convergence guarantees that are independent of the noise arising from simulations. These results are consistent with the results in the case where stochastic approximation is not used and error bounds are dependent on a bound on the noise arising from the simulations.

II. Other Related Works

Value iteration based multi-agent planning algorithms Markov games were first formulated in [23] and since then, other works have studied efficient methods for computing the Nash equilibrium using value function based methods including the works of [14], [17], [11].

Function approximation methods in multi-agent RL Function approximation methods have been studied in the setting of two-player zero-sum Markov games including in the work of [13] which studies linear value function approximation in games where knowledge of the model is assumed. Other prior works include the works of [33] which considered linear MDPs as well as the work of [12], which studies a general function approximation setting.

Policy iteration for games Another class of algorithms that has been proposed for computing the Nash equilibrium includes extensions of policy iteration algorithms for MDPs to Markov games. However, these algorithms have been far less successfully studied than the value iteration based methods, despite studies including [28] that have shown that Shapley's value iteration in games algorithm works slower in practice than the naive policy iteration algorithm in [20]. The works of [17], [19] have provided convergence guarantees of a policy iteration algorithm for Markov games. However, this algorithm has been shown to be rather computationally inefficient; in fact, the work of [9] shows that one has to solve $\Omega(1/(1-\alpha))$ MDPs to implement the algorithm. Furthermore, the work of [20] following the work of [10] proposes an algorithm that is far more computationally efficient but they only provide convergence guarantees for specific problem instances. The works of [8], [4] analyze variants of the algorithm in [20], however, these algorithms too only converge in specific settings. Notably, the works of [1], [3] provide convergence guarantees of a variant of the algorithm in [20], yet is unclear whether the algorithm in [1] can be extended to incorporate function approximation.

Policy gradient methods for two-player games A decentralized algorithm for policy gradient methods that converges to a min-max equilibrium when both players independently perform policy gradient has been studied in [5]. In addition to the work of [5], the work of [37] studies natural policy gradient and obtains performance bounds of the natural policy gradient algorithm extended to the two-player zero-sum games setting.

III. Model and Preliminaries

We consider a two-player simultaneous-action zero-sum discounted Markov game, characterized by $(\mathcal{S}, \mathcal{A}_1, \mathcal{A}_2, P, g, \alpha)$ where the state space is $\mathcal{S}$, the action space for the first player (maximizer) is $\mathcal{A}_1$ where $\mathcal{A}_1(s')$ is the action space at state s', the action space for the second player (the minimizer) is $\mathcal{A}_2$ where $\mathcal{A}_2(s')$ is the action space at state s' for the second player, P is the probability transition kernel, g is the reward function, and $\alpha \in (0,1)$ is the discount factor. At each instance i, the state of the game is x_i and the maximizer takes action u_i while the minimizer takes action v_i. The maximizer incurs a reward of $g(x_i, u_i, v_i)$ where $g(x_i, u_i, v_i) \in [0,1]$ and the minimizer incurs a cost of $g(x_i, u_i, v_i)$ (and, hence a reward of $-g(x_i, u_i, v_i)$). The maximizer seeks to take actions u_i to maximize $\sum_{i=0}^{\infty} \alpha^i g(x_i, u_i, v_i)$, the discounted sum of rewards, while the minimizer correspondingly takes actions v_i to minimize the discounted sum of rewards. We assume that at each instance i, the joint action taken is (u_i, v_i) following (non-deterministic) policy (μ, ν) where where $\mu(s) \in \Delta(\mathcal{A}_1(s))$ and $\nu(s) \in \Delta(\mathcal{A}_2(s))$. We say that (μ, ν) is the *policy* of the game.

We denote the *value function* corresponding to policy (μ, ν) component-wise as:

$$J^{\mu,\nu}(x) = E_{u_i \sim \mu(\cdot|x_i), v_i \sim \nu(\cdot|x_i)} \Big[\sum_{i=0}^{\infty} \alpha^i g(x_i, u_i, v_i) | x_0 = x \Big].$$

We call the Nash-equilibrium policy (μ^*, ν^*) or *optimal policy* the policy satisfying:

$$J^{\mu,\nu^*} \leq J^{\mu^*,\nu^*} \leq J^{\mu^*,\nu}$$

for all policies (μ, ν). We have that

$$J^* := J^{\mu^*,\nu^*}.$$

It is well known that a Nash equilibrium policy exists for two-player discounted zero-sum Markov games. We can also write the Nash equilibrium policy as follows:

$$(\mu^*, \nu^*) \in \arg\max_{\mu} \arg\min_{\nu} J^{\mu,\nu}.$$

The probability transition matrix for any policy (μ, ν) is $P(\mu, \nu) \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}$ where

$$P_{ij}(\mu, \nu) = \sum_x \sum_y \mu(x)\nu(y)P(j|i, \mu(x), \nu(y)).$$

The corresponding reward function is $r(\mu, \nu)$ where

$$r_i(\mu, \nu) = \sum_x \sum_y \mu(x)\nu(y)g(i, x, y).$$

The Bellman operator for policy (μ, ν), $T_{\mu,\nu} : \mathcal{S} \to \mathcal{S}$ is:

$$T_{\mu,\nu}J := r(\mu, \nu) + \alpha P(\mu, \nu)J.$$

We have that

$$\|T_{\mu,\nu}J - J^{\mu,\nu}\|_\infty \leq \alpha \|J - J^{\mu,\nu}\|_\infty .$$

The Bellman optimality operator or Bellman operator is $T : \mathcal{S} \to \mathcal{S}$ as

$$TJ = \max_{\mu} \min_{\nu}(T_{\mu,\nu}J).$$

Note that T is a pseudo-contraction towards the optimal value function J^* where

$$\|TJ - J^*\|_\infty \leq \alpha \|J - J^*\|_\infty .$$

It is well known that $T_{\mu,\nu}$ is monotone, i.e.,

$$J \leq J' \implies T_{\mu,\nu}J \leq T_{\mu,\nu}J'$$

and that

$$T_{\mu,\nu}(J + ce) = T_{\mu,\nu}J + \alpha ce \forall c \in \mathbb{R}$$

where $e(i) = 1 \forall i \in 1, 2, \ldots, |\mathcal{S}|$.

IV. Linear Value Function Approximation

The policy iteration algorithm for Markov games can be easily extended to incorporate linear function approximation. However, in this algorithm, at at every iteration k, all value functions corresponding to policies ν must be evaluated, which is computationally inefficient.

The work of [19] provides error bounds for the function approximation setting of the policy iteration algorithm for Markov games. However, even with function approximation, all policies $J^{\mu,\nu_{k+1}}$ must be evaluated which is inefficient.

A more efficient policy iteration algorithm for Markov games, sometimes called the algorithm of Pollatschek and Avi-Itzhak, does not converge [28], and hence, its variants and their extensions to function approximation, such as [18] suffers from the same difficulty with convergence. The work of [31] makes a small modification of the algorithm of Pollatschek and Avi-Itzhak that ensures convergence of a variant of the policy iteration for games algorithm. In the improved algorithm, the policy improvement step is replaced with a lookahead version of policy improvement. Lookahead has been widely used in practice [27], [26].

We now wish to extend the variant of the algorithm of Pollatschek and Avi-Itzhak in [31] to the function approximation setting, specifically, the linear value function approximation setting. In order to extend the algorithm in [31] to the linear value function approximation setting, we associate with each state $i \in \mathcal{S}$ a feature vector $\phi(i) \in \mathbb{R}^d$ where typically $d << |\mathcal{S}|$. The matrix comprised of all the feature vectors as rows is denoted by Φ. We use those estimates to find the best fitting $\theta \in \mathbb{R}^d$ based on estimates of $J^{\mu_{k+1},\nu_{k+1}}(i)$ for $i \in \mathcal{D}$, where $\mathcal{D}$ is a set of states, at each iteration k where $|\mathcal{D}| << |\mathcal{S}|$, i.e.,

$$\min_{\theta} \sum_{i\in D} \Big((\Phi\theta)(i) - \hat{J}^{\mu_{k+1},\nu_{k+1}}(i)\Big)^2,$$

where

$$\hat{J}^{\mu_{k+1},\nu_{k+1}}(i) = T^m_{\mu_{k+1},\nu_{k+1}} T^{H-1}(\Phi\theta_k)(i)$$

for $i \in \mathcal{D}$ and $\Phi\theta_k$ is the estimate of the optimal value function at iteration k. We note that we can estimate $\hat{J}^{\mu_{k+1},\nu_{k+1}}(i)$ using simulations since

$$\begin{aligned} &T^m_{\mu_{k+1},\nu_{k+1}} T^{H-1}(\Phi\theta_k)(i) \\ &= E_{u_i\sim\mu(\cdot|x_i),v_i\sim\nu(\cdot|x_i)}\Big[\sum_{i=0}^{m-1} \alpha^i g(x_i, u_i, v_i) \\ &+ \alpha^m T^{H-1}\Phi\theta_k(x_m)|x_0 = i\Big]. \end{aligned}$$

This procedure is referred to as taking m-step returns starting at state i. The policies (μ_{k+1}, ν_{k+1}) are the greedy policies corresponding to $T^{H-1}\Phi\theta_k$, i.e.,

$$\mu_{k+1}, \nu_{k+1}, \nu_{k+1} \in \arg\max_{\mu} \arg\min_{\nu} T_{\mu,\nu} T^{H-1}\Phi\theta_k,$$

also known as lookahead policies. For more on lookahead policies, see [29], [2]. While the computation of the lookahead is difficult, techniques such as Monte Carlo Tree Search have been shown to be successful in practice. Additionally, under certain structures, such as linear MDPs, the lookahead can be computed efficiently [31]. Our algorithm is outlined in Algorithm 1.

We denote the matrix of feature vectors in $\mathcal{D}$ as rows $\Phi_{\mathcal{D}}$ and we assume that the rank of $\Phi_{\mathcal{D}}$ is d, i.e., $\Phi_{\mathcal{D}}$ is full rank. Thus, we can alternatively rewrite our $\Phi\theta_k$ as follows:

$$\Phi\theta_k = \underbrace{\Phi(\Phi_{\mathcal{D}}^\top\Phi_{\mathcal{D}})^{-1}\Phi_{\mathcal{D}}^\top\mathcal{P}_k}_{=:\mathcal{M}} \hat{J}^{\mu_{k+1},\nu_{k+1}},$$

where $\mathcal{P}_k$ is a projection matrix of ones and zeros such that $\mathcal{P}_k\hat{J}^{\mu_{k+1},\nu_{k+1}}$ is a vector whose elements are a subset of the elements in $\hat{J}^{\mu_{k+1},\nu_{k+1}}$ corresponding to $\mathcal{D}$.

We now present our first main result on the convergence of Algorithm 1. To do so, we give our first assumption:

Assumption 1: We make the following assumption about the amount of lookahead and the parameter m:

$$\alpha^{H-1} + 2(1 + \alpha^m)\frac{\alpha^{H-1}}{1-\alpha} < 1.$$

Algorithm 1 Least-Squares Function Approximation Policy Iteration For Markov Games With Lookahead

Input: θ_0, m, H, feature vectors $\{\phi(i)\}_{i\in\mathcal{S}}, \phi(i) \in \mathbb{R}^d$ and subset $\mathcal{D} \subseteq \mathcal{S}$. Here $\mathcal{D}$ is the set of states at which we evaluate the current policy at iteration k.

1: Let $k = 0$.
2: Let $\mu_{k+1}, \nu_{k+1}, \nu_{k+1}$ be such that

$$\mu_{k+1}, \nu_{k+1}, \nu_{k+1} \in \arg\max_{\mu} \arg\min_{\nu} T_{\mu,\nu} T^{H-1} \Phi\theta_k,$$

where the T operator is the Bellman operator.
3: Compute $\hat{J}^{\mu_{k+1},\nu_{k+1}}(i) = T^m_{\mu_{k+1},\nu_{k+1}} T^{H-1}(\Phi\theta_k)(i) + w_{k+1}(i)$ for $i \in \mathcal{D}$.
4: Choose θ_{k+1} to solve

$$\min_{\theta} \sum_{i\in D} \left((\Phi\theta)(i) - \hat{J}^{\mu_{k+1},\nu_{k+1}}(i)\right)^2, \tag{1}$$

where Φ is a matrix whose rows are the feature vectors.
5: Set $k \leftarrow k + 1$. Go to 2.

Theorem 1: Almost surely, under Assumption 1 the following bound holds for iterates θ_k of Algorithm 1:

$$\limsup_{k\to\infty} \|\Phi\theta_k - J^*\|_\infty \leq \frac{\delta_{app}}{1 - 2\alpha^{H-1} - \delta_{FV}\alpha^{m+H-1}}$$

where δ_{app} is ability of the feature vectors to approximate the policies:

$$\delta_{app} := \sup_k \|J^{\mu_k} - \mathcal{M}J^{\mu_k}\|_\infty$$

and

$$\delta_{FV} := \|\mathcal{M}\|_\infty$$

is a parameter that depends on the feature vectors. ◇
See the extended version of the paper [31] for the proof of Theorem 1.

Interpretation of Theorem 1

Theorem 1 shows that the performance bounds are mostly determined by the ability of the feature vectors to represent value functions corresponding to policies. However, the performance bounds can also be improved with judicious choice of feature vectors, since the bounds are also dependent on δ_{FV}. In the special case where there is no function approximation error with the feature vectors, the algorithm has no error.

V. Stochastic Approximation

Note that in the previous section, we assume that exact estimates of $\hat{J}^{\mu_{k+1},\nu_{k+1}}(i) = T^m_{\mu_{k+1},\nu_{k+1}} T^{H-1}(\Phi\theta_k)(i) + w_{k+1}(i)$ for $i \in \mathcal{D}$ are available at each iteration. However, in general, $T^m_{\mu_{k+1},\nu_{k+1}} T^{H-1}(\Phi\theta_k)(i) + w_{k+1}(i)$ can be determined using a single trajectory, which incorporates sources of error. However, a single trajectory results in an unbiased estimate of $T^m_{\mu_{k+1},\nu_{k+1}} T^{H-1}(\Phi\theta_k)(i) + w_{k+1}(i)$, hence we wish to incorporate stochastic approximation techniques to limit the effect of noise on the convergence of the algorithm. We outline the stochastic approximation algorithm in Algorithm 2. Defining

Algorithm 2 Least Squares Function Approximation For Policy Iteration In Markov Games With Unbiased Noise and Lookahead

Input: θ_0, m, H feature vectors $\{\phi(i)\}_{i\in\mathcal{S}}, \phi(i) \in \mathbb{R}^d$ and subsets $\mathcal{D}_k \subseteq \mathcal{S}, k = 0, 1, \ldots$. Here $\mathcal{D}_k$ is the set of states visited by a trajectory corresponding to the current policy at iteration k.

1: Let $k = 0$.
2: Let μ_{k+1}, ν_{k+1} be such that

$$\left\|T_{\mu_{k+1},\nu_{k+1}} T^{H-1}\Phi\theta_k - T^H\Phi\theta_k\right\|_\infty \leq \varepsilon_{LA}.$$

3: Compute

$$\hat{J}^{\mu_{k+1},\nu_{k+1}}(i) = T^m_{\mu_{k+1},\nu_{k+1}} T^{H-1}\Phi\theta_k(i) + w_k(i)$$

for $i \in \mathcal{D}_k$ and set $\hat{J}^{\mu_{k+1},\nu_{k+1}}(i) = 0$ for $i \notin \mathcal{D}_k$.
4: Choose θ_{k+1} to solve

$$\min_{\theta} \left\|(\mathcal{P}_{1,k}\Phi)\theta - \mathcal{P}_{2,k}\hat{J}^{\mu_{k+1},\nu_{k+1}}\right\|_2, \tag{2}$$

where Φ is a matrix whose rows are the feature vectors (see below for definitions of $\mathcal{P}_{1,k}$ and $\mathcal{P}_{2,k}$). To compute θ_{k+1}, use the Moore-Penrose inverse of $\mathcal{P}_{1,k}\Phi$.
5:

$$\theta_{k+1} = (1 - \gamma_k)\theta_k + \gamma_k(\theta_{k+1}). \tag{3}$$

6: Set $k \leftarrow k + 1$. Go to 2.

$$V_k := \Phi\theta_k,$$

the iterates in Algorithm 2 can be written as follows:

$$\begin{aligned} V_{k+1} &= (1 - \gamma_k)V_k + \gamma_k(\Phi\theta_{k+1}) \\ &= (1 - \gamma_k)V_k + \gamma_k(\Phi(\mathcal{P}_{1,k}\Phi)^+\hat{J}^{\mu_{k+1},\nu_{k+1}}) \\ &= (1 - \gamma_k)V_k \\ &+ \gamma_k(\underbrace{\Phi(\mathcal{P}_{1,k}\Phi)^+\mathcal{P}_{2,k}}_{=:\mathcal{M}_k}(T^m_{\mu_{k+1},\nu_{k+1}} T^{H-1}V_k + w_k)), \end{aligned} \tag{4}$$

where $(\mathcal{P}_{1,k}\Phi)^+$ is the Moore-Penrose inverse of $\mathcal{P}_{1,k}\Phi$ and $\mathcal{P}_{1,k}$ is a matrix of zeros and ones such that rows of $\mathcal{P}_{1,k}\Phi$ correspond to feature vectors associated with states in $\mathcal{D}_k$ and $\mathcal{P}_{2,k}(T^m_{\mu_{k+1},\nu_{k+1}} T^{H-1}V_k + w_k)$ is a vector whose elements are a subset of the elements of $\hat{J}^{\mu_{k+1},\nu_{k+1}}$ corresponding to $\mathcal{D}_k$. We define the term δ'_{FV} associated with our feature vectors $\phi(i) \forall i \in \mathcal{S}$ as follows:

$$\delta'_{FV} := \sup_k \|\mathcal{M}_k\|_\infty.$$

Using δ'_{FV}, we now give Assumption 2 which is similar to Assumption 1, adapted to Algorithm 2:

Assumption 2:

$$\delta'_{FV}\alpha^{m+H-1}\frac{1+\alpha}{1-\alpha}+\frac{2\alpha^{H-1}}{1-\alpha}<1.$$

$\diamond$

Using Assumption 2, we provide our main performance bounds for Algorithm 2:

Theorem 2: When

1) The starting state of the trajectory at each instance is drawn from a fixed distribution, p, where $p(i)>0 \forall i \in \mathcal{S}$.
2) $\sum_{i=0}^{\infty}\gamma_i=\infty$. Also, $\sum_{i=0}^{\infty}\gamma_i^2<\infty$.

under Assumption 2, the iterates obtained in (4) almost surely have the following property:

$$\limsup_{k\to\infty}\|V_k-J^*\|_\infty \leq \frac{\delta'_{app}}{1-2\alpha^{H-1}-(2+\delta'_{FV})\alpha^{m+H-1}}$$

almost surely, where δ'_{app} is ability of the feature vectors to approximate the policies:

$$\delta'_{app}:=\sup_k E[\|\mathcal{M}_k(J^{\mu_{k+1},\nu_{k+1}}+w_k)-(J^{\mu_{k+1},\nu_{k+1}}+w_k)\|_\infty|\mathcal{F}_k]. \quad (5)$$

$\diamond$

See the extended version of the paper [31] for the proof of Theorem 2.

Interpretation of Theorem 2

Analogously to Theorem 1, Theorem 2 shows that the performance bound is largely based on the ability of the feature vectors represent unbiased estimates of the value functions. Without feature vectors (i.e., when feature vectors are simply unit vectors) and trajectories from all states are obtained (i.e., in the special case where the Markov chains induced by all policies are irreducible and infinitely long trajectories are obtained, the error becomes zero.

VI. Future Work

Some interesting directions for further work involve extending the results to incorporate TD-learning instead of returns as well as investigating the the stochastic shortest path games problem setting where there is no discount factor. For more on stochastic shortest path games, see the [17].

VII. Acknowledgments

This work was supported by NSF grants CCF 22-07547, CNS 21-06801, CCF 1934986, and ONR grant N00014-19-1-2566.

References

[1] Dimitri Bertsekas. Distributed asynchronous policy iteration for sequential zero-sum games and minimax control. *arXiv preprint arXiv:2107.10406*, 2021.

[2] Dimitri P Bertsekas. *Reinforcement learning and optimal control*. Athena Scientific Belmont, MA, 2019.

[3] Sarnaduti Brahma, Yitao Bai, Duy Anh Do, and Thinh T Doan. Convergence rates of asynchronous policy iteration for zero-sum markov games under stochastic and optimistic settings. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 3493–3498. IEEE, 2022.

[4] Michèle Breton, Jerzy A Filar, Alain Haurle, and Todd A Schultz. *On the computation of equilibria in discounted stochastic dynamic games*. Springer, 1986.

[5] Constantinos Daskalakis, Dylan J Foster, and Noah Golowich. Independent policy gradient methods for competitive reinforcement learning. *Advances in neural information processing systems*, 33:5527–5540, 2020.

[6] Yonathan Efroni, Gal Dalal, Bruno Scherrer, and Shie Mannor. Multiple-step greedy policies in online and approximate reinforcement learning. *arXiv preprint arXiv:1805.07956*, 2018.

[7] Yonathan Efroni, Gal Dalal, Bruno Scherrer, and Shie Mannor. How to combine tree-search methods in reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3494–3501, 2019.

[8] Jerzy A Filar and Boleslaw Tolwinski. *On the Algorithm of Pollatschek and Avi-ltzhak*. Springer, 1991.

[9] Thomas Dueholm Hansen, Peter Bro Miltersen, and Uri 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.

[10] Alan J Hoffman and Richard M Karp. On nonterminating stochastic games. *Management Science*, 12(5):359–370, 1966.

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

[12] Chi Jin, Qinghua Liu, and Tiancheng Yu. The power of exploiter: Provable multi-agent rl in large state spaces. In *International Conference on Machine Learning*, pages 10251–10279. PMLR, 2022.

[13] Michail Lagoudakis and Ron Parr. Value function approximation in zero-sum markov games. *arXiv preprint arXiv:1301.0580*, 2012.

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

[15] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. *CoRR*, abs/1602.01783, 2016.

[16] Asuman Ozdaglar, Muhammed O Sayin, and Kaiqing Zhang. Independent learning in stochastic games. *arXiv preprint arXiv:2111.11743*, 2021.

[17] Stephen David Patek. *Stochastic and shortest path games: theory and algorithms*. PhD thesis, Massachusetts Institute of Technology, 1997.

[18] Julien Pérolat, Bilal Piot, Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. Softened approximate policy iteration for markov games. In *International Conference on Machine Learning*, pages 1860–1868. PMLR, 2016.

[19] Julien Perolat, Bruno Scherrer, Bilal Piot, and Olivier Pietquin. Approximate dynamic programming for two-player zero-sum markov games. In *International Conference on Machine Learning*, pages 1321–1329. PMLR, 2015.

[20] MA Pollatschek and B Avi-Itzhak. Algorithms for stochastic games with geometrical interpretation. *Management Science*, 15(7):399–415, 1969.

[21] M. Puterman and M. C. Shin. Modified policy iteration algorithms for discounted markov decision problems. *Management Science*, 24:1127–1137, 1978.

[22] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*, 2016.

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

[24] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[25] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy P. Lillicrap, Karen Simonyan, and Demis Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *CoRR*, abs/1712.01815, 2017.

[26] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017.

[27] Gerald Tesauro and Gregory Galperin. On-line policy improvement using monte-carlo search. *Advances in Neural Information Processing Systems*, 9, 1996.

[28] J Van Der Wal. Discounted markov games: Generalized policy iteration method. *Journal of Optimization Theory and Applications*, 25(1):125–138, 1978.

[29] Anna Winnicki, Joseph Lubars, Michael Livesay, and R. Srikant. The role of lookahead and approximate policy evaluation in policy iteration with linear value function approximation. *CoRR*, abs/2109.13419, 2021.

[30] Anna Winnicki and R. Srikant. Reinforcement learning with unbiased policy evaluation and linear function approximation. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 801–806, 2022.

[31] Anna Winnicki and R. Srikant. A new policy iteration algorithm for reinforcement learning in zero-sum markov games, 2023.

[32] Anna Winnicki and R Srikant. On the convergence of policy iteration-based reinforcement learning with monte carlo policy evaluation. *Artificial Intelligence and Statistics*, 2023.

[33] Qiaomin Xie, Yudong Chen, Zhaoran Wang, and Zhuoran Yang. Learning zero-sum simultaneous-move markov games using function approximation and correlated equilibrium. In *Conference on learning theory*, pages 3674–3682. PMLR, 2020.

[34] Yang Yang, Li Juntao, and Peng Lingling. Multi-robot path planning based on a deep reinforcement learning dqn algorithm. *CAAI Transactions on Intelligence Technology*, 5(3):177–183, 2020.

[35] Yaodong Yang and Jun Wang. An overview of multi-agent reinforcement learning from game theoretical perspective. *arXiv preprint arXiv:2011.00583*, 2020.

[36] Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of reinforcement learning and control*, pages 321–384, 2021.

[37] Yulai Zhao, Yuandong Tian, Jason Lee, and Simon Du. Provably efficient policy optimization for two-player zero-sum markov games. In *International Conference on Artificial Intelligence and Statistics*, pages 2736–2761. PMLR, 2022.