Analysing the Sample Complexity of Opponent Shaping

Kitty Fung*
University of Oxford
Oxford, United Kingdom
kittyfung01@gmail.com

Qizhen Zhang*
University of Oxford
Oxford, United Kingdom
qizhen.zhang@eng.ox.ac.uk

Chris Lu
University of Oxford
Oxford, United Kingdom
christopher.lu@eng.ox.ac.uk

Jia Wan
Massachusetts Institute of Technology
Cambridge, United States
jiawan@mit.edu

Timon Willi
University of Oxford
Oxford, United Kingdom
timon.willi@eng.ox.ac.uk

Jakob Foerster
University of Oxford
Oxford, United Kingdom
jakob.foerster@eng.ox.ac.uk

ABSTRACT

Learning in general-sum games often yields collectively sub-optimal results. Addressing this, *opponent shaping* (OS) methods actively guide the learning processes of other agents, empirically leading to improved individual and group performances in many settings. Early OS methods use higher-order derivatives to shape the learning of co-players, making them unsuitable to shape multiple learning steps. Follow-up work, Model-free Opponent Shaping (M-FOS), addresses these by reframing the OS problem as a *meta-game*. In contrast to early OS methods, there is little theoretical understanding of the M-FOS framework. Providing theoretical guarantees for M-FOS is hard because A) there is little literature on theoretical sample complexity bounds for meta-reinforcement learning B) M-FOS operates in continuous state and action spaces, so theoretical analysis is challenging. In this work, we present R-FOS, a tabular version of M-FOS that is more suitable for theoretical analysis. R-FOS discretises the continuous meta-game MDP into a tabular MDP. Within this discretised MDP, we adapt the R_{max} algorithm, most prominently used to derive PAC-bounds for MDPs, as the meta-learner in the R-FOS algorithm. We derive a sample complexity bound that is exponential in the cardinality of the inner state and action space and the number of agents. Our bound guarantees that, with high probability, the final policy learned by an R-FOS agent is close to the optimal policy, apart from a constant factor. Finally, we investigate how R-FOS's sample complexity scales in the size of state-action space. Our theoretical results on scaling are supported empirically in the Matching Pennies environment.

KEYWORDS

Opponent Shaping; Multi-Agent; Reinforcement Learning; Meta Reinforcement Learning; Sample Complexity

ACM Reference Format:
Kitty Fung[1], Qizhen Zhang[1], Chris Lu, Jia Wan, Timon Willi, and Jakob Foerster. 2024. Analysing the Sample Complexity of Opponent Shaping. In *Proc.of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 9 pages.

*Equal Contribution
Corresponding Authors: kittyfung01@gmail.com, qizhen.zhang@eng.ox.ac.uk

Proc.of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand. © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

1 INTRODUCTION

Learning in general-sum games commonly leads to collectively worst-case outcomes [6]. To address this, *opponent shaping* (OS) methods account for opponents' learning steps and influence other agents' learning processes. Empirically, this can improve individual and group performances.

Early OS methods [6, 11, 13] rely on higher-order derivatives, which are high-variance and result in unstable learning. They are also myopic, focusing only on the opponent's immediate future learning steps rather than their long-term development [15]. Recent work, Model-free Opponent Shaping (M-FOS) [15], solves the above challenges. M-FOS introduces a *meta-game* structure, each *meta-step* representing an episode of the embedded "inner" game. The *meta-state* consists of "inner" policies, and the *meta-policy* generates an inner policy at each *meta-step*. M-FOS uses model-free optimisation techniques to train the meta-policy, eliminating the need for higher-order derivatives to accomplish long-horizon opponent shaping. The M-FOS framework has shown promising long-term shaping results in social-dilemma games [10, 15].

The original M-FOS paper presents two cases of the M-FOS algorithm. For simpler, low-dimensional games, M-FOS learns policy updates directly by taking policies as input and outputting the next policy as an action. Inputting and outputting entire policies does not extend well to more complex, higher-dimensional games, e.g. when policies are represented as neural networks. The original M-FOS paper also proposes a variant which uses trajectories as inputs instead of the exact policy representations. In this work we derive the sample complexity for both cases.

Whereas some previous OS algorithms enjoy strong theoretical foundations thanks to the Differentiable Games framework [1], the M-FOS framework has not been investigated theoretically. Understanding the sample complexity of an algorithm is helpful in many ways, such as evaluating its efficiency or predicting the learning time. However, providing theoretical guarantees for M-FOS is challenging because A) there is very little literature on theoretical sample complexity bounds for even single-agent meta-reinforcement learning (RL), let alone multi-agent and B) M-FOS operates in continuous state and action space (the meta-game).

In this work, we present R-FOS, a tabular algorithm approximation of M-FOS. Unlike M-FOS, which operates in a continuous meta-MDP, R-FOS operates in a discrete approximation of the original meta-MDP. The resulting *discrete* MDP allows us to perform rigorous theoretical analysis. We adapt R-FOS from M-FOS such that it still maintains all the key properties of M-FOS. Within this discrete, approximate MDP, R-FOS applies the R_{MAX} algorithm [9, 21] to the M-FOS meta-game. R_{MAX} is a model-based reinforcement learning (MBRL) algorithm typically used for the sample complexity analysis of tabular MDPs. Using existing results developed for R_{MAX} [3], we derive an exponential sample complexity PAC-bound, which guarantees with high probability (1-δ) that the optimal policy in the discretised meta-MDP is very close ($< \epsilon$ away) to the policy learned by R-FOS. We then derive several bounds which guarantee policies between the original meta-MDP and the discretised meta-MDP are close to each other up to a constant distance. Lastly, combining all of the previous bounds we derived, we obtain the final exponential sample complexity result.

For notational simplicity, we mostly omit the "meta" prefix in the rest of the paper. For example, the terms "MDP", "transition function", and "policy", refer to the meta-MDP, meta-transition function, and meta-policy respectively. We use the prefix "inner" whenever we refer to the inner game. Furthermore, our analysis of M-FOS is limited to the asymmetric *shaping* case (i.e. the meta-game of shaping a naive inner-learner*) and we leave the extension to meta-selfplay for future work.

Our contributions are three-fold:

(1) We present R-FOS (see Algorithm 1), a tabular approximation of M-FOS. Instead of learning a meta-policy inside the continuous meta-MDP M, R-FOS learns a meta-policy inside a discretised meta-MDP which approximates M. Inside this discretised Meta-MDP, R-FOS uses R_{MAX} as the meta-agent. Note that R-FOS still maintains key properties of the original M-FOS algorithm, such as being able to exploit naive learners.
(2) We present an exponential sample complexity bound for both cases described in M-FOS (See Theorems 4.12 and 4.13). Specifically, we prove that, with high probability, the final R-FOS policy is close to the optimal policy in the original meta-MDP up to a constant distance.
(3) We implement R-FOS † and analyse the empirical sample complexity in the *Matching Pennies* environment. We establish links between theory and experiments by demonstrating that in both realms, sample complexity scales exponentially with the inner-game's state-action space size.

2 RELATED WORK

Theoretical Analysis of Differentiable Games: Much past work assumes that the game being optimised is differentiable [1]. This assumption enables far easier theoretical analysis because one can directly use end-to-end gradient-based methods rather than reinforcement learning in those settings. Several works in this area investigate the convergence properties of various algorithms Balduzzi et al. [1], Letcher [12], Schäfer and Anandkumar [18].

Opponent Shaping: More closely related to our work are methods that specifically analyse OS. SOS [13] and COLA [22] both analyse opponent-shaping methods that operate in the differentiable games framework. These works provide theoretical *convergence* analysis for opponent-shaping algorithms; however, neither work analyzes sample complexity. POLA [24] theoretically analyses an OS method that is invariant to policy parameterization. M-FOS does not operate in the differentiable games framework. While this enables M-FOS to scale to more challenging environments, such as Coin Game [15], it comes at the cost of convenient theoretical analysis. Khan et al. [10] empirically scales M-FOS to more challenging environments with larger state spaces, while Lu et al. [16] empirically investigates applying M-FOS to a state-based adversary. To the best of our knowledge, our work is the first to theoretically analyse OS outside of the differentiable games framework. Furthermore, our work is the first to analyse the sample complexity of an OS method.

Theoretical Analysis of Sample Complexity in RL: There are several works that use the R_{MAX} [3] framework to derive the sample complexity of RL algorithms across a variety of settings. Closely related to our work is Zhang et al. [23], which uses the R_{MAX} algorithm to derive sample complexity bounds for learning in fully-cooperative multi-agent RL.

Our work is also related to methods that analyse sample complexity on continuous-space RL. Analyzing the sample complexity of algorithms in continuous-space RL is particularly challenging because there are an infinite number of potential states. To address this, numerous techniques have been suggested that each make specific assumptions: Liu and Brunskill [14] assumes a **stationary asymptotic occupancy distribution** under a random walk in the MDP. Malik et al. [17] uses an effective planning window to handle MDPs with non-linear transitions. However, neither of these assumptions applies to M-FOS.

Instead, this work focuses on **discretising** the continuous space and expresses the complexity bounds in terms of the discretisation grid size. This is related to the concept of *state aggregation* [2, 20], which groups states into clusters and treats the clusters as the states of a new MDP. These previous works only formulated the aggregation setting in MDPs and did not provide theoretical or empirical sample complexity proofs.

Furthermore, prior studies on *PAC-MDP* did not empirically verify the connection between the sample complexity and size of the state-action space. In this work, we **empirically verify the relationship between the sample complexity and the cardinality of the inner-state-action-space** in the Matching Pennies game.

3 BACKGROUND

3.1 Stochastic Game

A stochastic game (SG)‡ is given by a tuple $G = \langle \mathcal{I}, S, \boldsymbol{A}, T, \boldsymbol{R}, \gamma \rangle$. $\mathcal{I} = \{1, \cdots, n\}$ is the set of agents, S is the state space, $\boldsymbol{A}$ is the cross-product of the action space for each agent such that the joint action space $\boldsymbol{A} = A^1 \times \cdots \times A^n$, $T : S \times \boldsymbol{A} \mapsto S$ is the transition function, $\boldsymbol{R}$ is the cross-product of reward functions for all agents

*Naive learners are players who update their policy assuming other learning agents are simply a part of the environment.
†The project code is available on https://github.com/FLAIROx/rfos

‡We use the **bold** notation to indicate vectors over $\boldsymbol{n}$ agents.

Algorithm 1 The R-FOS Algorithm

Meta-game Inputs: Discretised meta-MDP $\langle \hat{S}_d, \hat{A}_d, T_d, R_d, \gamma \rangle$ m-known, meta-game horizon h_{meta}
Inner-game Inputs: Inner game G = $\langle S, A, T_{\text{inner}}, R_{\text{inner}} \rangle$ inner-game horizon h_{inner}
Initialisation: $\forall \hat{s}_d \in \hat{S}_d, \hat{a}_d \in \hat{S}_d, \hat{s}'_d \in \hat{S}_d$
$\hat{Q}(\hat{s}_d, \hat{a}_d) \leftarrow 0,\ r(\hat{s}_d, \hat{a}_d) \leftarrow 0,\ n(\hat{s}_d, \hat{a}_d) \leftarrow 0,\ n(\hat{s}_d, \hat{a}, \hat{s}') \leftarrow 0$

for meta-episode = 0, 1, .. **do**
  Reset environment
  for meta-time step = 1, 2, ..., h_{meta} **do**
    Choose $\hat{a}_d := \text{argmax}_{\hat{a}_d \in \hat{A}_d} \hat{Q}(\hat{s}_d, \hat{a}'_d)$
    Roll-out K inner games of length h_{inner} using $\hat{a}_d = \phi_t$
    Inner-game opponents each update their own inner-policies naively
    Let R be our agent's K inner-games' discounted return
    Let $\hat{s}'_d$ be the next meta-state after executing meta-action $\hat{a}_d$ from meta-state $\hat{s}_d$
    if $n(\hat{s}_d, \hat{a}_d) < m$ **then**
      $r(\hat{s}_d, \hat{a}_d) \leftarrow r(\hat{s}_d, \hat{a}_t) + R$
      $n(\hat{s}_d, \hat{a}_d) \leftarrow n(\hat{s}_d, \hat{a}_t) + 1$
      $n(\hat{s}_d, \hat{a}_d, \hat{s'}_d) \leftarrow n(\hat{s}_d, \hat{a}_d, \hat{s}'_d) + 1$
      if $n(\hat{s}_d, \hat{a}_d) = m$ **then**
        for $i = 1, 2, 3, \cdots, \lceil \frac{ln(\frac{1}{\varepsilon(1-\gamma)})}{1-\gamma} \rceil$ **do**
          for all $(\hat{s}, \hat{a})$ **do**
            if $n(\hat{s}_d, \hat{a}_d) \geq m$ **then**
              $\hat{Q}(\hat{s}_d, \hat{a}_d) \leftarrow \hat{R}_d(\hat{s}_d, \hat{a}_d) + \gamma \sum_{\hat{s}'_d} \hat{T}_d(\hat{s}' | \hat{s}_d, \hat{a}_d) \max_{\hat{a}'_d} \hat{Q}(\hat{s}'_d, \hat{a}'_d)$
    $\hat{s} \leftarrow \hat{s}'$

such that the joint reward space $\boldsymbol{R} = R^1 \times \cdots \times R^n$, and $\gamma \in [0, 1)$ is the discount factor.

In an SG, agents simultaneously choose an action according to their stochastic policy at each timestep t, $a^i_t \sim \pi^i_{\phi^i}(\cdot|s^i_t)$. The joint action at timestep t is $\boldsymbol{a_t} = \{a^i_t, \boldsymbol{a^{-i}_t}\}$, where the superscript $\boldsymbol{-i}$ indicates all agents except agent i and ϕ^i is the policy parameter of agent i. The agents then receive reward $r^i_t = R^i(s_t, \boldsymbol{a_t})$ and observe the next state $s_{t+1} \sim T(\cdot|s_t, \boldsymbol{a_t})$, resulting in a trajectory $\tau^i = (s_0, \boldsymbol{a_0}, r^i_0, ..., s_T, \boldsymbol{a_T}, r^i_T)$, where T is the episode length.

3.2 Markov Decision Process

A Markov decision process (MDP) is a special case of stochastic game and can be described as $\mathcal{M} = \langle S, A, T, R, \gamma \rangle$, where S is the state space, A is the action space, $T\left(s_{t+1} \mid s_t, a_t\right)$ is the transition function, $R\left(s_t, a_t\right)$ is the reward function, and γ is the discount factor. At each timestep t, the agent takes an action $a_t \in A$ from a state $s_t \in S$ and moves to a next state $s_{t+1} \sim T\left(\cdot \mid s_t, a_t\right)$. Then, the agent receives a reward $r_t = R\left(s_t, a_t\right)$.

3.3 Model-Free Opponent Shaping

Model-free Opponent Shaping (M-FOS) [15] frames the OS problem as a meta-reinforcement-learning problem, in which the opponent *shaper* plays a meta-game. The meta-game is an MDP (sometimes we also refer to it as meta-MDP), in which the meta-agent controls one of the inner agents in the inner game.

The inner game is the actual environment that our agents are playing, which is an SG. The original M-FOS describes two cases for the meta-state:

(1) In the meta-game at timestep t, the M-FOS agent is at the meta-state $\hat{s}_t = [\phi^i_{t-1}, \boldsymbol{\phi^{-i}_{t-1}}]$, which contains all inner-agents' policy parameters for the underlying SG. In this work, we assu,e all inner-agents are parameterised by their Q-value table.
(2) Alternatively, $\hat{s}_t = \boldsymbol{\tau}$ in cases where past trajectories of the inner-game represent the policies sufficiently.

We provide theoretical sample complexity results for both of these two cases

The meta-agent takes a meta-action $\hat{a}_t = \phi^i_t \sim \pi_\theta(\cdot|\hat{s}_t)$, which is the M-FOS' inner agent's policy parameters. The action is chosen from the meta-policy π parameterized by parameter θ. In this work, we only look at the case where the meta-policy is a Q-value function table, and is denoted as $\hat{Q}$ instead. The M-FOS agent receives reward $r_t = \sum_{k=0}^{K} r^i_k(\phi^i_t, \boldsymbol{\phi^{-i}_t})$, where K is the number of inner episodes. A new meta-state is sampled from a stochastic transition function $\hat{s}_{t+1} \sim T(\cdot|\hat{s}_t, \hat{a}_t)$.

Note that the original paper introduces two different algorithms: The first meta-trains M-FOS against *naive learners* commonly resulting in exploiting them. The second instead considers *meta-self-play*, whereby two M-FOS agents are trained to shape each other, resulting in reciprocity. In this work we only consider the first, asymmetric case.

3.4 The R_{MAX} Algorithm

R_{MAX} [3] is an MBRL algorithm proposed for analysing the sample complexity for tabular MDPs. Given any MDP M, R_{MAX} constructs an *empirical MDP* $\hat{M}$ that approximates M. The approximation is done by estimating the reward function R and transition T using *empirical* samples. The resulting approximate reward and transition models are denoted by $\hat{R}$ and $\hat{T}$ respectively.

R_{MAX} encourages exploration by dividing the state-action pairs into two groups - those that have been visited at least m-times, and those that haven't. The set of state-action pairs that have been visited at least m-times is called the "m-known set". Using the empirical MDP $\hat{M}$, the R_{MAX} algorithm constructs an *m-known empirical MDP*. This m-known empirical MDP behaves almost exactly as the empirical MDP, except when the agent is at a state-action pair outside the m-known set. When the agent is outside the m-known set, the transition function is self-absorbing (i.e. the transition function only transitions back to the current state) and the reward function is the maximum (See Table **??** in the appendix). The consequence of the m-known setup is the agent is encouraged to explore state-action pairs that have high uncertainty (i.e. that has been visited under m times). Specifically, the value function for the under-visited states is the maximum possible expected return, which gives the algorithm its name. This is in line with *optimism in the face of uncertainty*.

3.5 ε-Nets

Definition 3.1. (ε-Net [5, 8]) For $\varepsilon > 0$, $\mathcal{N}_\varepsilon$ is an ε-net over the set $\Theta \subseteq \mathbb{R}^D$ if for all $\theta \in \Theta$, there exists $\theta' \in \mathcal{N}_\varepsilon$ such that $\left\|\theta - \theta'\right\|_2 \leq \varepsilon$.

To discretise a D-dimensional sphere of radius R, we can use a ε-net containing D-dimensional cubes of sides λ. This results in $\left(\frac{2R}{\lambda} + 1\right)^D$ points. Within each D-dimensional cube, the largest distance between the vertices and the interior points comes from the center of the cube, which is $\frac{\lambda\sqrt{d}}{2}$. Therefore, to guarantee a full cover of all the points in the sphere, the largest cube size that we can have should satisfy $\varepsilon = \frac{\lambda\sqrt{d}}{2}$.From here on, we will replace the ε in ε-net with α to avoid notation overloading.

4 SAMPLE COMPLEXITY ANALYSIS WITH R_{MAX} AS META-AGENT

As introduced in Section 3, R_{MAX} [3] is a MBRL algorithm for learning in tabular MDPs. We adapt the original M-FOS algorithm to use R_{MAX} as the meta-agent (see Algorithm 1) and refer to this adapted algorithm as *R-FOS* from here on. We use a *tabular Q-learner* as the naive learner for all inner-game opponents. While the original M-FOS paper uses PPO [19], we choose the Q-learner for the ease of sample complexity analysis.

We provide theoretical results for the two cases of M-FOS' meta-agent proposed by the original paper [15]. *Case I* uses all agents' inner policy parameters from the previous timestep as the meta-state. *Case II* instead uses the most recent inner-game trajectories as the meta-state. In both cases, the meta-action determines the inner agent's policy parameters for the next inner episode.

At a high level, we first discretise the meta-MDP, which allows us to use the theoretical bounds from R_{MAX} (only suitable for tabular MDPs), then we develop theory for bounding the discrepancy between the continuous and discrete meta-MDP, and lastly, we use all of this to bound the final discrepancy. Specifically, the sample complexity analysis consists of six steps [§]:

(1) To use R_{MAX} as the M-FOS meta-agent, we first discretise the continuous meta-MDP $M = \langle \hat{S}, \hat{A}, T, R, \gamma \rangle$ into a discretised meta-MDP $M_d = \langle \hat{S}_d, \hat{A}_d, T_d, R_d, \gamma \rangle$. We first discretise the continuous meta-state space and meta-action space using epsilon-nets [5, 8]. Based on this discretised meta-state and meta-action space, we define the discretised transition and reward function. See Section 4.2 for details.
(2) We then construct a m-known discretised MDP M_m, d, as described by the R-MAX algorithm [21]. See Section 4.3 for details.
(3) Then, we deploy the R_{MAX} algorithm in M_m, d. R_{MAX} both estimates the empirical m-known discretised MDP, $\hat{M}_m, d$, using a maximum likelihood estimate from empirical samples and learns an optimal policy in $\hat{M}_m, d$. For example, to estimate the meta-reward, our algorithm, R-FOS evaluates the inner-game policy outputted by the meta-policy using episodic rollouts. The estimates are then used to update the meta-policy according to the R-FOS algorithm. Our R-FOS algorithm *optimistically* assigns rewards for all under-visited discretised (meta-state, meta-action) pairs to encourage exploration (like R_{MAX}). See Section 4.4 for details.
(4) We next prove a PAC-bound which guarantees with large probability, that the optimal policy learnt in $\hat{M}_m, d$ is similar to the optimal policy in M, d. This step uses results from [21]. See Section 4.5 for details.
(5) We also prove a strict bound that guarantees the optimal policies learnt in M, d and M are similar up to a constant. This step uses results from [4]. See Section 4.8 for details.
(6) Using the two bounds from above, we prove the final sample complexity guarantee which quantifies that, with large probability, the optimal policy learnt in $\hat{M}_m, d$ is similar to the optimal policy in M up to a constant. See Section 4.9 for details.

[§] see detailed proof in the appendix on arXiv [7]

4.1 Assumptions

We first outline all assumptions made in deriving the sample complexity of the R-FOS algorithm.

Assumption 4.1. Both meta-game and inner-game are finite horizon. We use h_{meta} to denote the meta-game horizon, and h_{inner} to denote the inner-game horizon.

Assumption 4.2. We assume the inner-game reward is bounded. For simplicity of the proof and without loss of generality, we set this bound as $\frac{1}{h_{\text{inner}}}$, where h_{inner} is the horizon of the inner game. Formally, for all (s, a), $0 \leq R_{\text{inner}}(s, a) \leq \frac{1}{h_{\text{inner}}}$. This allows us to introduce the notion of *maximum inner reward* and *maximum inner value function* as $R_{\text{max,inner}} = \frac{1}{h_{\text{inner}}}$ and $V_{\text{max,inner}} = 1$ respectively. This implies that the reward and value function in the meta-game are also bounded, i.e., $R_{\text{max}} = 1$ and $V_{\text{max}} = \frac{1}{1-\gamma}$ (the latter being an upper bound).

Assumption 4.3. The meta-game uses a discount factor of γ. For simplicity of the proof, the inner-game uses a discount factor of 1. This assumption can be easily deleted by adapting $R_{\text{max, inner}}$ (see above) in the original proof in [21].

Assumption 4.4. For simplicity, the inner game is assumed to be discrete.

Assumption 4.5. The meta-reward function is Lipschitz-continuous: For all $\hat{s}_1, \hat{s}_2 \in \hat{S}$ and $\hat{a}_1, \hat{a}_2 \in \hat{A}$,

$$\left|R(\hat{s}_1, \hat{a}_1) - R(\hat{s}_2, \hat{a}_2)\right| \leq \mathcal{L}_R \left\|(\hat{s}_1, \hat{a}_1) - (\hat{s}_2, \hat{a}_2)\right\|_\infty$$

where $\mathcal{L}_R$ is the meta-reward function's Lipschitz-constant.

Assumption 4.6. The meta-transition function is Lipschitz-continuous: For all $\hat{s}_1, \hat{s}'_1, \hat{s}_2 \in \hat{S}$ and $\hat{a}_1, \hat{a}'_1, \hat{a}_2 \in \hat{A}$,

$$\left|T(\hat{s}'_1 \mid \hat{s}_1, \hat{a}_1) - T(\hat{s}'_2 \mid \hat{s}_2, \hat{a}_2)\right| \leq \mathcal{L}_T \left\|(\hat{s}'_1, \hat{s}_1, \hat{a}_1) - (\hat{s}'_2, \hat{s}_2, \hat{a}_2)\right\|_\infty$$

where $\mathcal{L}_T$ is the meta-transition function's Lipschitz-constant.

Assumption 4.7. There's a Lipschitz-continuous point-to-set mapping between meta-state space and meta-action space such that for any $\hat{s}, \hat{s}' \in \hat{S}$ and $\hat{a}, \hat{a}' \in \hat{A}$, there exists some $\hat{a} \in U(\hat{s})$ such that $\left\|\hat{a} - \hat{a}'\right\|_\infty < \mathcal{L}\left\|\hat{s} - \hat{s}'\right\|_\infty$

Assumption 4.8. The meta-game transition function $T(\cdot \mid \hat{s}, \hat{a})$ is a probability density function such that $0 \leq T(\hat{s}' \mid \hat{s}, \hat{a}) \leq \mathcal{L}$ and $\int_{\hat{S}} T(\hat{s}' \mid \hat{s}, \hat{a})\, d\hat{s}' = 1, \quad \forall \hat{s}, \hat{s}' \in \hat{S}$ and $\hat{a} \in \hat{A}$

The first four assumptions are required to be able to use the R-MAX algorithm, while the latter assumptions are needed for bounding the discrepancy between the continuous meta-MDP and the discretised meta-MDP.

4.2 Step 1: Discretising the Meta-MDP

To use R_{MAX} as the M-FOS meta-agent, we discretise the continuous meta-MDP $M = \langle \hat{S}, \hat{A}, T, R, \gamma \rangle$ into a discretised meta-MDP $M_d = \langle \hat{S}_d, \hat{A}_d, T_d, R_d, \gamma \rangle$. We discretise the continuous state and action space using ε-nets with spacing α.

4.2.1 Discretising the State and Action Space: Case I

In *Case I*, the meta-state $\hat{s}_t$ is all inner agents' policies parameters from the previous timestep. Each of the inner agent i's policy is a Q-table, denoted as $\phi_i \in \mathbb{R}^{|S|\times|A|}$. Formally, $\hat{s}_t := \boldsymbol{\phi}_{t-1} = [\phi^i_{t-1}, \boldsymbol{\phi^{-i}_{t-1}}]$. The meta-action $\hat{a}_t$ is the inner agent's current policy parameters ϕ^i_t.

For the meta-action space $\hat{A}$ and a chosen discretisation error $\alpha > 0$, we obtain the ε-net $\hat{A}_d \subset \hat{A}$ such that for all $\hat{a} \in \hat{A}$, there exist $\hat{a}_d \in \hat{A}_d$ where

$$\left\| \hat{a} - \hat{a}_d \right\| \leq \alpha. \tag{1}$$

Dividing the space with grid size λ results in the size of discretised meta-action space upper bounded by

$$|\hat{A}_d| \leq \left(\frac{2\sqrt{|S||A|}}{\lambda} + 1 \right)^{|S||A|}. \tag{2}$$

Similarly, the size of the discretised meta-state space is upper bounded by

$$|\hat{S}_d| \leq \left(\frac{2\sqrt{n|S||A|}}{\lambda} + 1 \right)^{n|S||A|}. \tag{3}$$

4.2.2 Discretising the State and Action Space: Case II

In Case II, the meta-state $\hat{s}_t$ is all inner agents' past trajectories. Formally, $\hat{s}_t := \boldsymbol{\tau}_t.$, where $\tau^i_t = \{s_0, a_0, s_1, a_1, ..., s_t, a_t\}$. Because we assume the Inner-Game is discrete (i.e. the state and action space are both discrete), the meta-state in this case does not need discretisation. Let h be the maximum length of the past trajectories combined, i.e. $h = h_{\text{inner}} \cdot h_{\text{meta}}$. The size of the meta-state space is

$$|\hat{S}_t| = (|S||A|)^{nh}. \tag{4}$$

The meta-action remains the same as Case I.

4.2.3 Discretising the Transition and Reward Function

Under the above discretisation procedure, we define the discretised MDP $M_d = (\hat{S}, \hat{A}_d, T_d, R_d, \gamma)$, where the state space remains continuous and the action space is restricted to *discretised* actions. We define the transition function and reward function for M_d as:

$$T_d(\hat{s}' \mid \hat{s}, \hat{a}_d) = \frac{T(\hat{s}'_d \mid \hat{s}_d, \hat{a}_d)}{\int_{\hat{S}} T(\hat{z}_d \mid \hat{s}_d, \hat{a}_d) d\hat{z}} \tag{5}$$

Intuitively, $T_d(\hat{s}' \mid \hat{s}_d, \hat{a}_d)$ is a normalized sample of $T_d(\cdot \mid \cdot, \hat{a}_d)$ at $\hat{s}', \hat{s}_d$, and the transition probability $T_d(\cdot \mid \hat{s}, \hat{a}_d)$ takes a constant value within each grid in the state space. This means that instead of treating the transition function as a discretised distribution of all possible values of $\hat{S}_d$, we treat it as a continuous distribution over the original continuous state space, but normalize each grid from the ε-net into a step function.

$$R_d(\hat{s}, \hat{a}_d) = R(\hat{s}_d, \hat{a}_d) \tag{6}$$

Similarly, the reward function is continuous over the state space, but normalized each grid from the ε-net into a step function.

4.3 Step 2: The m-known Discretised MDP

In the previous step, we converted the meta-MDP M into a *discretised* meta-MDP M_d. From M_d, R-FOS builds an m-known discretised MDP $M_{m,d}$ (see Table 1).

Definition 4.9 (m-Known MDP)**.** Let $M_d = \langle \hat{S}_d, \hat{A}_d, T_d, R_d, \gamma \rangle$ be an MDP. We define M_m, d to be the m-known MDP. As is standard practice, m-known refers to the set of state-action pairs that have been visited at least m times. For all state-action pairs in m-known, the induced MDP $M_{m,d}$ behaves identical to M_d. For state-action pairs outside of m-known, the state-action pairs are self-absorbing (i.e. only self-transitions) and maximally rewarding with *RMAX*.

4.4 Step 3: The Empirical Discretised MDP

From the m-known discretised MDP $M_{m,d}$, we then learn an *empirical* m-known discretised MDP $M_{m,d}$ by calculating the maximum likelihood from empirical samples (see Table 1). As shown in Algorithm 1, R-FOS learns an optimal policy within this empirical m-known discretised MDP.

Definition 4.10 (Empirical m-Known discretised MDP)**.** $M_{m,d}$ is the expected version of $\hat{M}_{m,d}$ where:

$$\begin{aligned}
T_{m,d}(\hat{s}'_d \mid \hat{s}_d, \hat{a}_d) &:= \begin{cases} T_d(\hat{s}'_d \mid \hat{s}_d, \hat{a}_d) & \text{if } (\hat{s}_d, \hat{a}_d) \in \text{m-known} \\ \mathbb{1}[\hat{s}'_d = \hat{s}_d], & \text{otherwise} \end{cases} \\
\hat{T}_{m,d}(\hat{s}'_d \mid \hat{s}_d, \hat{a}_d) &:= \begin{cases} \frac{n(\hat{s}_d, \hat{a}_d, \hat{s}'_d)}{n(\hat{s}_d, \hat{a}_d)}, & \text{if } (\hat{s}_d, \hat{a}_d) \in \text{m-known} \\ \mathbb{1}[\hat{s}'_d = \hat{s}_d], & \text{otherwise} \end{cases} \\
R_{m,d}(\hat{s}_d, \hat{a}_d) &:= \begin{cases} R_d(\hat{s}_d, \hat{a}_d), & \text{if } (\hat{s}_d, \hat{a}_d) \in \text{m-known} \\ R_{\max} & \text{otherwise} \end{cases} \\
\hat{R}_{m,d}(\hat{s}_d, \hat{a}_d) &= \begin{cases} \frac{\sum_i^{n(\hat{s}_d, \hat{a}_d)} r(\hat{s}_d, \hat{a}_d)}{n(\hat{s}_d, \hat{a}_d)}, & \text{if } (\hat{s}_d, \hat{a}_d) \in \text{m-known} \\ R_{\max}, & \text{otherwise} \end{cases}
\end{aligned} \tag{7}$$

4.5 Step 4: The Bound Between M_d and $\hat{M}_{m,d}$

We first prove the PAC bound which guarantees that, with high probability, the optimal policies learnt in the discretised MDP M_d and empirical $m-known$ discretised MDP are very close. We prove the bound using results from [21].

Theorem 4.11. *(R_{MAX} MDP Bound [21]) Suppose that $0 \leq \varepsilon < \frac{1}{1-\gamma}$ and $0 \leq \delta < 1$ are two real numbers and $M = \langle S, A, T, R, \gamma \rangle$ is any MDP. There exists inputs $m = m\left(\frac{1}{\varepsilon}, \frac{1}{\delta}\right)$ and ε_1, satisfying $m\left(\frac{1}{\varepsilon}, \frac{1}{\delta}\right) = O\left(\frac{(S+\ln(SA/\delta))V^2_{\max}}{\varepsilon^2(1-\gamma)^2}\right)$ and $\frac{1}{\varepsilon_1} = O\left(\frac{1}{\varepsilon}\right)$, such that if R_{MAX} is executed*

	Ground Truth MDP M	Discretised MDP M_d	m-known Discretised MDP $M_{m,d}$	Empirical m-known Discretised MDP $\hat{M}_{m,d}$
m-known	$= M$	$= M_d$	$= M_d$	$\approx M_d$
Outside m-known	$= M$	$= M_d$	self-loop with maximum reward	

Table 1: Comparison between $M, M_d, M_{m,d}, \hat{M}_{m,d}$

on M with inputs m and ε_1, the following holds. Let A_t denote R_{MAX}'s policy at time t and s_t denote the state at time t. With probability at least $1-\delta$, $V_M^{A_t}(s_t) \geq V_M^*(s_t) - \varepsilon$ is true for all but

$$\tilde{O}\left(S^2A/\left(\varepsilon^3(1-\gamma)^6\right)\right)$$

timesteps (final sample complexity bound).

4.6 Case I

Directly plugging in Equations 3 and 2 into Theorem 4.11, we obtain the following PAC-bound.

Theorem 4.12. *Suppose that $0 \leq \varepsilon < \frac{1}{1-\gamma}$ and $0 \leq \delta < 1$ are two real numbers. Let M be any continuous meta-MDP with inner stochastic game $G = \langle S, A, T_{inner}, R_{inner}\rangle$. Let us denote $M_d = \langle \hat{S}_d, \hat{A}_d, T_d, R_d, \gamma\rangle$ as discretised version of M (as described in Case I) using grid size of λ. There exists inputs $m = m\left(\frac{1}{\varepsilon}, \frac{1}{\delta}\right)$ and ε_1, satisfying*

$$m\left(\frac{1}{\varepsilon}, \frac{1}{\delta}\right) = \tilde{O}\left(\frac{\left(\frac{2\sqrt{n|S||A|}}{\lambda}+1\right)^{n|S||A|}}{\varepsilon^2(1-\gamma)^4}\right)$$

and $\frac{1}{\varepsilon_1} = O\left(\frac{1}{\varepsilon}\right)$, such that if R-MAX is executed on M with inputs m and ε_1, then the following holds. Let $\mathcal{A}_t$ denote R-MAX's policy at time t and $\hat{s}_t$ denote the state at time t. With probability at least $1-\delta$, $V_{M_d}^*(\hat{s}_t) - V_{M_d}^{\mathcal{A}_t}(\hat{s}_t) \leq \varepsilon$ is true for all but

$$\tilde{O}\left(\frac{\left(\frac{2\sqrt{n|S||A|}}{\lambda}+1\right)^{2n|S||A|}\left(\frac{2\sqrt{|S||A|}}{\lambda}+1\right)^{|S||A|}}{\varepsilon^3(1-\gamma)^6}\right)$$

timesteps.

4.7 Case II

Directly plugging in Equations 4 and 2 into Theorem 4.11, we obtain the folling PAC-bound.

Theorem 4.13. *Suppose that $0 \leq \varepsilon < \frac{1}{1-\gamma}$ and $0 \leq \delta < 1$ are two real numbers. Let M be any continuous meta-MDP with inner stochastic game $G = \langle S, A, T_{inner}, R_{inner}\rangle$. Let us denote $M_d = \langle \hat{S}_d, \hat{A}_d, T_d, R_d, \gamma\rangle$ as discretised version of M (as described in Case I) using grid size of λ. There exists inputs $m = m\left(\frac{1}{\varepsilon}, \frac{1}{\delta}\right)$ and ε_1, satisfying*

$$m\left(\frac{1}{\varepsilon}, \frac{1}{\delta}\right) = \tilde{O}\left(\frac{(|S||A|)^{nh}}{\varepsilon^2(1-\gamma)^4}\right)$$

and $\frac{1}{\varepsilon_1} = O\left(\frac{1}{\varepsilon}\right)$, such that if R-MAX is executed on M with inputs m and ε_1, then the following holds. Let $\mathcal{A}_t$ denote R-MAX's policy at time t and $\hat{s}_t$ denote the state at time t. With probability at least $1-\delta$, $V_{M_d}^*(\hat{s}_t) - V_{M_d}^{\mathcal{A}_t}(\hat{s}_t) \leq \varepsilon$ is true for all but

$$\tilde{O}\left(\frac{(|S||A|)^{nh}\left(\frac{2\sqrt{|S||A|}}{\lambda}+1\right)^{|S||A|}}{\varepsilon^3(1-\gamma)^6}\right)$$

timesteps.

4.8 Step 5: The Bound between M and M_d

Next, we give a guarantee that the optimal policies learnt in the original meta-MDP M and the discretised MDP M_d are similar enough with a distance up to a constant factor. Using the results from [4], we obtain the following property.

Theorem 4.14. *(MDP Discretization Bound [8]) There exists a constant $\mathcal{K}$ (thats depends only on the Lipschitz constant $\mathcal{L}$) such that for some discretisation coarseness $\lambda \in (0, \frac{\mathcal{L}}{2}]$*

$$\left\|V_M^* - V_{M_d}^*\right\|_\infty \leq \frac{\mathcal{K}\lambda}{(1-\hat{\gamma})^2}.$$

4.9 Step 6: Adding it together

To combine the bounds we obtained in Step 4 and 5, we need an additional bound that bounds the policy value between the continuous and discretised MDP.

Lemma 4.15 (Simulation Lemma for Continuous MDPs)**.** *Let M and $\hat{M}$ be two MDPs that only differ in (T, R) and $(\hat{T}, \hat{R})$.*

Let $\epsilon_R \geq \max_{s,a} |\hat{R}(s,a) - R(s,a)|$ and $\varepsilon_p \geq \max_{s,a} \|\hat{T}(\cdot \mid s,a) - T(\cdot \mid s,a)\|_1$. Then $\forall \pi : \hat{S} \to a$,

$$\left\|V_M^\pi - V_{\hat{M}}^\pi\right\|_\infty \leq \frac{\varepsilon_R}{1-\gamma} + \frac{\gamma\epsilon_P V_{\max}}{2(1-\gamma)}.$$

Under discretisation, [4] showed that, with a small enough grid size, and restricting to the discretised action space, the difference in transition probability of the continuous MDP M and discretised MDP M_d is upper bounded by a constant.

Lemma 4.16. *[4] There exists a constant K_P (depending only on constant $\mathcal{L}$) such that*

$$|T_d(\hat{s}'|\hat{s},\hat{a}_d) - T(\hat{s}'|\hat{s},\hat{a}_d)| \leq K_p\alpha$$

for all $\hat{s}', \hat{s} \in \hat{S}, \hat{a}_d \in \hat{A}_d$ *and all* $\alpha \leq (0, \frac{1}{2}\mathcal{L}]$

We now apply the Lemma 4.15 to bound the difference in value for any discretised policy (i.e. restricting action space to $\hat{A}_d$) in the continuous MDP M and discretised MDP M_d.

Lemma 4.17. *Let* $M_{\hat{A}_d} = (\hat{S}, \hat{A}_d, T, R, \gamma)$ *be the continuous MDP M restricted to the discretised action space. Recall the discretised MDP* $M_d = (\hat{S}, \hat{A}_d, T_d, R_d, \gamma)$*. Then for any discretised policy* $\pi : \hat{S} \to \hat{A}_d$*,*

$$\|V^{\pi}_{M_{\hat{A}_d}} - V^{\pi}_{M_d}\|_\infty = \|V^{\pi}_{M} - V^{\pi}_{M_d}\|_\infty \leq \frac{\mathcal{L}_{\mathcal{R}}\alpha}{1-\gamma} + \frac{\gamma K_p \alpha}{(1-\gamma)^2}$$

Note that, restricted to discretised policies π which only picks actions in $\hat{A}_d$, the value of π in the original MDP M, V^{π}_{M}, equals to its value the same MDP restricted to discretised action space, $V^{\pi}_{M_{\hat{A}_d}}$.

4.10 Case I

Summing up the bounds in Lemma 4.17, Theorems 4.12 and 4.14, we obtain the final bound for Case I. The final bound guarantees, with high probability, that the policy we obtain from R-FOS is close to the optimal policy in M apart from a constant factor.

Theorem 4.18. *Suppose that* $0 \leq \varepsilon < \frac{1}{1-\gamma}$ *and* $0 \leq \delta < 1$ *are two real numbers. Let M be any continuous meta-MDP with inner stochastic game* $G = \langle S, A, T_{inner}, R_{inner} \rangle$*. Let us denote* $M_d = \langle \hat{S}_d, \hat{A}_d, T_d, R_d, \gamma \rangle$ *as discretised version of M (as described in Case I) using grid size of* λ*. There exists inputs* $m = m\left(\frac{1}{\varepsilon}, \frac{1}{\delta}\right)$ *and* ε_1*, satisfying*

$$m\left(\frac{1}{\varepsilon}, \frac{1}{\delta}\right) = \tilde{O}\left(\frac{\left(\frac{2\sqrt{n|S||A|}}{\lambda} + 1\right)^{n|S||A|}}{\varepsilon^2(1-\gamma)^4}\right)$$

and $\frac{1}{\varepsilon_1} = O\left(\frac{1}{\varepsilon}\right)$*, such that if R-MAX is executed on M with inputs m and* ε_1*, then the following holds. Let* $\mathcal{A}_t$ *denote R-MAX's policy at time t and* $\hat{s}_t$ *denote the state at time t. With probability at least* $1-\delta$*,*

$$V^{*}_{M}(\hat{s}_t) - V^{\mathcal{A}_t}_{M}(\hat{s}_t) \leq \varepsilon + \frac{\mathcal{K}\lambda}{(1-\hat{\gamma})^2} + \frac{\mathcal{L}_{\mathcal{R}}\alpha}{1-\gamma} + \frac{\gamma K_p \alpha}{(1-\gamma)^2}$$

is true for all but

$$\tilde{O}\left(\frac{\left(\frac{2\sqrt{n|S||A|}}{\lambda} + 1\right)^{2n|S||A|}\left(\frac{2\sqrt{|S||A|}}{\lambda} + 1\right)^{|S||A|}}{\varepsilon^3(1-\gamma)^6}\right)$$

timesteps. I.e. the above is the final sample complexity.

4.11 Case II

Similarly, summing up the bounds in Lemma 4.17, Theorems 4.13 and 4.14, we obtain the final bound for Case II. In Section 5, we also show empirically that the number of samples needed indeed scales by a factor of $|S||A|^{2nh}$, as seen in Theorem 4.19.

Theorem 4.19. *Suppose that* $0 \leq \varepsilon < \frac{1}{1-\gamma}$ *and* $0 \leq \delta < 1$ *are two real numbers. Let M be any continuous meta-MDP with inner stochastic game* $G = \langle S, A, T_{inner}, R_{inner} \rangle$*. Let us denote* $M_d = \langle \hat{S}_d, \hat{A}_d, T_d, R_d, \gamma \rangle$ *as discretised version of M (as described in Case I) using grid size of* λ*. There exists inputs* $m = m\left(\frac{1}{\varepsilon}, \frac{1}{\delta}\right)$ *and* ε_1*, satisfying*

$$m\left(\frac{1}{\varepsilon}, \frac{1}{\delta}\right) = \tilde{O}\left(\frac{(|S||A|)^{nh}}{\varepsilon^2(1-\gamma)^4}\right)$$

and $\frac{1}{\varepsilon_1} = O\left(\frac{1}{\varepsilon}\right)$*, such that if R-MAX is executed on M with inputs m and* ε_1*, then the following holds. Let* $\mathcal{A}_t$ *denote R-MAX's policy at time t and* $\hat{s}_t$ *denote the state at time t. With probability at least* $1-\delta$*,*

$$V^{*}_{M}(\hat{s}_t) - V^{\mathcal{A}_t}_{M}(\hat{s}_t) \leq \varepsilon + \frac{\mathcal{K}\lambda}{(1-\hat{\gamma})^2} + \frac{\mathcal{L}_{\mathcal{R}}\alpha}{1-\gamma} + \frac{\gamma K_p \alpha}{2(1-\gamma)^2}$$

is true for all but

$$\tilde{O}\left(\frac{(|S||A|)^{2nh}\left(\frac{2\sqrt{|S||A|}}{\lambda} + 1\right)^{|S||A|}}{\varepsilon^3(1-\gamma)^6}\right)$$ *timesteps*

5 EXPERIMENTS

We now validate our theoretical findings empirically.

5.1 The Matching Pennies Environment

Matching Pennies is a two-player, zero-sum game with a payoff matrix shown in Section 5.1. Each agent either pick Heads (H) or Tails (T), $a^i \in \{H, T\}$ and $a^i \sim \pi_{\phi^i}(\cdot \mid \{\})$, where ϕ^i correspond to the probability of player i picking H. Note that in this work, the game is not iterated. This means that an inner-episode has a length of 1 and the inner-episodic return corresponds to the payoff after one interaction $r = \text{Payoff Table}(a^1, a^2)$. For R-FOS, this means that a meta-step corresponds to one iteration of the Matching Pennies game. The meta-return corresponds to the discounted, cumulative meta-reward after playing the Matching Pennies K times. While the original M-FOS was evaluated on a more complex, iterated version of the Matching Pennies game, this simple setting with a binary action space is sufficient for our empirical validation. Our setting is also more practical for implementation because the R_{MAX} algorithm memory usage grows exponentially with the size of the state and action space. Thus, for any of the more complex environments from the M-FOS paper we were not able do any empirical analysis of R-FOS at all, due to the exponential sample and memory requirements.

Player 1\Player 2	Head	Tail
Head	(+1, -1)	(-1, +1)
Tail	(-1, +1)	(+1, -1)

Table 2: Payoff matrix for the Matching Pennies environment.

5.2 Experiment Setup

We implement an empirical version of our R-FOS algorithm. Because the R-FOS algorithm uses Q-value iteration to solve the meta-game, the algorithm needs to keep a copy of the meta-Q-value table. Therefore, memory usage grows exponentially with respect

to the inner-game's state-action space size. We found that Case I of the algorithm was intractable to implement even with a compact environment like MP. The meta Q-table of size $|\hat{S}| \times |\hat{A}|$ was simply too large to fit in memory. Therefore, we focus on empirically validating a simplified case of Case II. We make two simplifications,

(1) The meta-state uses a partial history of past actions. Only the most recent h actions are used, where h is a hyper-parameter we pick. The window size allows us to control the size of the meta-game state, i.e., $\hat{S} \in \mathbb{R}^{2^h}$. Because the MP game only has one state, it is not necessary to include the state.
(2) To further decrease the problem size for tractability, we define the meta-agent action to be the inner-agent's greedy action, instead of the Q-table. This results in a much small meta-action size of $|\hat{A}| = 2$

6 RESULTS AND DISCUSSION

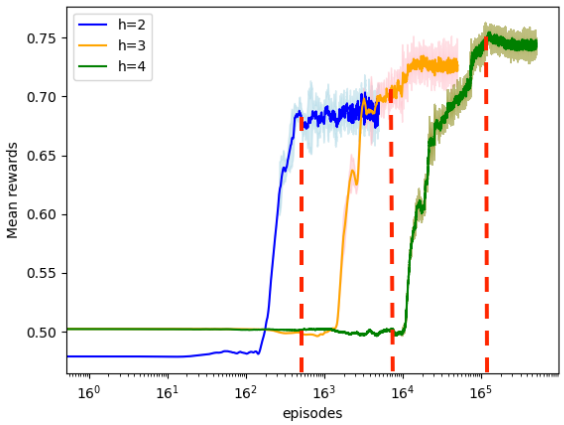

Figure 1: Empirical sample complexity while varying the trajectory window h. We plot the meta-reward per meta-episode. To better visualise the connection with the theory results, we plot the x-axis in $\log_{16}$ scale. The reported results are the mean over 3 seeds with standard error.

We draw connections between our sample complexity theory results and experimental results in the MP environment. Our goal is to analyse the scaling law of R-FOS. Specifically, we investigate how the sample complexity changes when we vary the window-size h. Under the MP environment settings, the inner-game state-action space size is $|S||A| = 2$ and the number of players is $n = 2$. Following the bound in Theorem 4.19, we see that the only term that depends on h is the 16^h term:

$$\tilde{O}\left(\frac{(|S||A|)^{2nh}\left(\frac{2\sqrt{|S||A|}}{\lambda}+1\right)^{|S||A|}}{\varepsilon^3(1-\gamma)^6}\right) \sim \tilde{O}\left(\frac{16^h\left(\frac{2\sqrt{2}}{\lambda}+1\right)^2}{\varepsilon^3(1-\gamma)^6}\right)$$

Hence, our theory results says that whenever the game horizon is increased by 1, we expect to see the sample complexity to increase by a factor of 16 in the MP environment. Figure 1 shows the reward across the meta episodes on a log 16 scale. The graph contains three reward curves for meta-trajectory length $h = 2, 3, 4$, which converges approximately at $16^3, 16^4, and 16^5$ episodes. Indeed, this is consistent with our theoretical results in Theorem 4.13.

7 CONCLUSION

We presented three main contributions in our work. First of all, we presented R-FOS, a tabular algorithm adapted from M-FOS. Unlike M-FOS, which learns a policy in a continuous meta-MDP, R-FOS instead learns a policy in a discrete approximation of the original meta-MDP which allows us to more easily perform theoretical analysis. Within this discretised meta-MDP, R-FOS uses the R_{MAX} algorithm as the meta-agent. We adapted R-FOS from M-FOS such that it still maintains all key attributes of M-FOS. Second of all, we derived an exponential sample complexity bound for both cases described in M-FOS (the two cases being either inner-game policies or inner-game trajectory history as meta-state). Specifically, we proved that with high probability, the policy learnt by R-FOS is close to the optimal policy from the original meta-MDP up to a constant distance. Finally, we implemented R-FOS and investigated the empirical sample complexity in the Matching Pennies environment. We draw connections between theory and experiments by showing both results scales exponentially according to the size of the inner-game's state-action-space.

ACKNOWLEDGMENTS

KF was supported by the D. H. Chen Foundation Scholarship. QZ is supported by Armasuisse and Cohere.

REFERENCES

[1] David Balduzzi, Sebastien Racaniere, James Martens, Jakob Foerster, Karl Tuyls, and Thore Graepel. 2018. The mechanics of n-player differentiable games. In *International Conference on Machine Learning*. PMLR, 354–363.

[2] Craig Boutilier, Thomas Dean, and Steve Hanks. 1999. Decision-Theoretic Planning: Structural Assumptions and Computational Leverage. *J. Artif. Int. Res.* 11, 1 (jul 1999), 1–94.

[3] Ronen Brafman and Moshe Tennenholtz. 2001. R-MAX - A General Polynomial Time Algorithm for Near-Optimal Reinforcement Learning. *The Journal of Machine Learning Research* 3, 953–958. https://doi.org/10.1162/153244303765208377

[4] CHEE-S Chow and John N Tsitsiklis. 1991. An optimal one-way multigrid algorithm for discrete-time stochastic control. *IEEE transactions on automatic control* 36, 8 (1991), 898–914.

[5] Murat A. Erdogdu. 2022. Covering with epsilon-nets. https://erdogdu.github.io/csc2532/lectures/lecture05.pdf

[6] Jakob N. Foerster, Richard Y. Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. 2018. Learning with Opponent-Learning Awareness. arXiv:1709.04326 [cs.AI]

[7] Kitty Fung, Qizhen Zhang, Chris Lu, Jia Wan, Timon Willi, and Jakob Foerster. 2024. Analysing the Sample Complexity of Opponent Shaping. *arXiv preprint arXiv:2402.05782* (2024).

[8] D Haussler and E Welzl. 1986. Epsilon-Nets and Simplex Range Queries. In *Proceedings of the Second Annual Symposium on Computational Geometry* (Yorktown Heights, New York, USA) *(SCG '86)*. Association for Computing Machinery, New York, NY, USA, 61–71. https://doi.org/10.1145/10515.10522

[9] Sham Kakade. 2003. *On the sample complexity of Reinforcement Learning*. Ph.D. Dissertation. University of London.

[10] Akbir Khan, Newton Kwan, Timon Willi, Chris Lu, Andrea Tacchetti, and Jakob Nicolaus Foerster. [n.d.]. Context and History Aware Other-Shaping. ([n. d.]).

[11] Dong-Ki Kim, Miao Liu, Matthew D Riemer, Chuangchuang Sun, Marwa Abdulhai, Golnaz Habibi, Sebastian Lopez-Cot, Gerald Tesauro, and JONATHAN P HOW. 2021. A Policy Gradient Algorithm for Learning to Learn in Multiagent Reinforcement Learning. https://openreview.net/forum?id=zdrls6LIX4W

[12] Alistair Letcher. 2020. On the impossibility of global convergence in multi-loss optimization. *arXiv preprint arXiv:2005.12649* (2020).

[13] Alistair Letcher, Jakob Foerster, David Balduzzi, Tim Rocktäschel, and Shimon Whiteson. 2018. Stable opponent shaping in differentiable games. *arXiv preprint arXiv:1811.08469* (2018).

[14] Yao Liu and Emma Brunskill. 2018. When Simple Exploration is Sample Efficient: Identifying Sufficient Conditions for Random Exploration to Yield PAC RL Algorithms.

[15] Chris Lu, Timon Willi, Christian A. Schroeder de Witt, and Jakob N. Foerster. 2022. Model-Free Opponent Shaping. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA (Proceedings of Machine Learning Research, Vol. 162)*. PMLR, 14398–14411.

[16] Chris Lu, Timon Willi, Alistair Letcher, and Jakob Foerster. 2022. Adversarial Cheap Talk. *arXiv preprint arXiv:2211.11030* (2022).

[17] Dhruv Malik, Aldo Pacchiano, Vishwak Srinivasan, and Yuanzhi Li. 2021. Sample Efficient Reinforcement Learning In Continuous State Spaces: A Perspective Beyond Linearity. In *International Conference on Machine Learning*.

[18] Florian Schäfer and Anima Anandkumar. 2019. Competitive gradient descent. *Advances in Neural Information Processing Systems* 32 (2019).

[19] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[20] Satinder P. Singh, Tommi Jaakkola, and Michael I. Jordan. 1994. Reinforcement Learning with Soft State Aggregation. In *Proceedings of the 7th International Conference on Neural Information Processing Systems* (Denver, Colorado) *(NIPS'94)*. MIT Press, Cambridge, MA, USA, 361–368.

[21] Alexander L. Strehl, Lihong Li, and Michael L. Littman. 2009. Reinforcement Learning in Finite MDPs: PAC Analysis. *J. Mach. Learn. Res.* 10 (dec 2009), 2413–2444.

[22] Timon Willi, Alistair Hp Letcher, Johannes Treutlein, and Jakob Foerster. 2022. COLA: consistent learning with opponent-learning awareness. In *International Conference on Machine Learning*. PMLR, 23804–23831.

[23] Qizhen Zhang, Chris Lu, Animesh Garg, and Jakob Foerster. 2022. Centralized Model and Exploration Policy for Multi-Agent RL. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*. https://arxiv.org/abs/2107.06434v2

[24] Stephen Zhao, Chris Lu, Roger B Grosse, and Jakob Foerster. 2022. Proximal Learning With Opponent-Learning Awareness. *Advances in Neural Information Processing Systems* 35 (2022), 26324–26336.