Journal of Machine Learning Research 25 (2024) 1-31 Submitted 4/23; Revised 2/24; Published 3/24

Off-Policy Action Anticipation in Multi-Agent Reinforcement Learning

Ariyan Bighashdel A.BIGHASHDEL@TUE.NL
Daan de Geus D.C.D.GEUS@TUE.NL
Pavol Jancura P.JANCURA@TUE.NL
Gijs Dubbelman G.DUBBELMAN@TUE.NL
Department of Electrical Engineering
Eindhoven University of Technology
Eindhoven, 5612 AZ, The Netherlands

Editor: Alessandro Lazaric

Abstract

Learning anticipation in Multi-Agent Reinforcement Learning (MARL) is a reasoning paradigm where agents anticipate the learning steps of other agents to improve cooperation among themselves. As MARL uses gradient-based optimization, learning anticipation requires using Higher-Order Gradients (HOG), with so-called HOG methods. Existing HOG methods are based on *policy parameter anticipation*, i.e., agents anticipate the changes in policy parameters of other agents. Currently, however, these existing HOG methods have only been developed for differentiable games or games with small state spaces. In this work, we demonstrate that in the case of non-differentiable games with large state spaces, existing HOG methods do not perform well and are inefficient due to their inherent limitations related to policy parameter anticipation and multiple sampling stages. To overcome these problems, we propose Off-Policy Action Anticipation (OffPA2), a novel framework that approaches learning anticipation through action anticipation, i.e., agents anticipate the changes in actions of other agents, via off-policy sampling. We theoretically analyze our proposed OffPA2 and employ it to develop multiple HOG methods that are applicable to non-differentiable games with large state spaces. We conduct a large set of experiments and illustrate that our proposed HOG methods outperform the existing ones regarding efficiency and performance.

Keywords: Multi-agent reinforcement learning, Reasoning, Learning anticipation, opponent shaping

1. Introduction

In multi-agent systems, the paradigm of *agents' reasoning about other agents* has been explored and researched extensively (Goodie et al., 2012; Liu and Lakemeyer, 2021). Recently, this paradigm is also being studied in the subfield of Multi-Agent Reinforcement Learning (MARL) (Wen et al., 2019, 2020; Konan et al., 2022). Generally speaking, MARL deals with several agents simultaneously learning and interacting in an environment. In the context of MARL, one reasoning strategy is anticipating the learning steps of other agents (Zhang and Lesser, 2010), i.e., learning anticipation. As MARL uses gradient-based optimization, learning anticipation naturally leads to the usage of Higher-Order Gradients (HOG), with

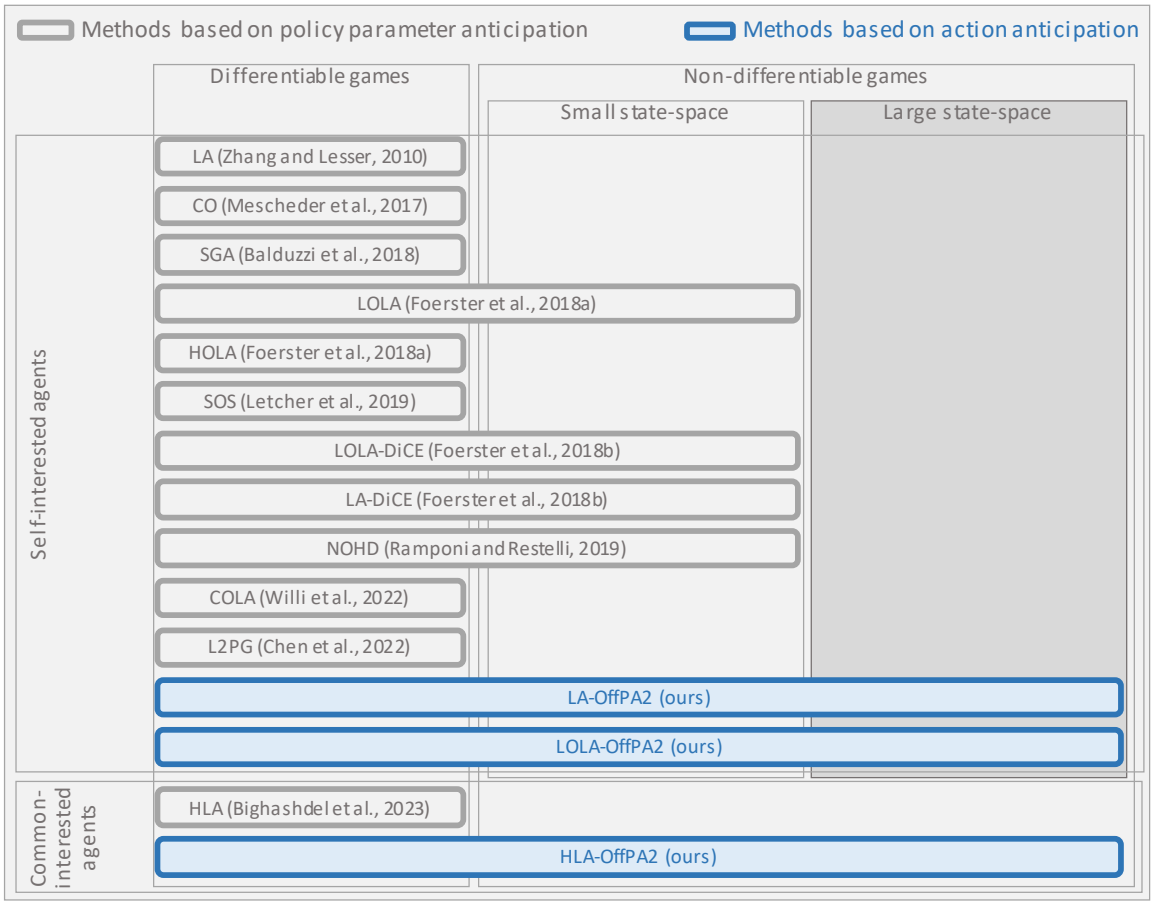


Figure 1: Overview of HOG methods and their applicability in various game settings. Existing HOG methods (gray rectangles) are based on the policy parameter approach. These methods have been developed for differentiable games or non-differentiable games with small state spaces. Our proposed HOG methods (blue rectangles) are developed in our novel framework of Off-Policy Action Anticipation (OffPA2) and can be applied to non-differentiable games with large state spaces.

so-called HOG methods (Letcher et al., 2019). The significance of learning anticipation in HOG methods has been frequently shown in the literature. For instance, Look-Ahead (LA) (Zhang and Lesser, 2010; Letcher et al., 2019) uses learning anticipation to guarantee convergence in cyclic games such as matching pennies, Learning with Opponent-Learning Awareness (LOLA) (Foerster et al., 2018a) employs learning anticipation to ensure cooperation in general-sum games such as Iterated Prisoner's Dilemma (IPD), and Hierarchical Learning anticipation (HLA) (Bighashdel et al., 2023) utilizes learning anticipation to improve coordination among common-interested agents in fully-cooperative games. In this study, we explore the limitations of current HOG methods and propose novel solutions so that learning anticipation can be applied to a broader range of MARL problems. In Fig-

ure 1, we provide an overview of the applicability of both existing and our proposed HOG methods.

Learning anticipation in the current HOG methods is developed based on the *policy parameter anticipation* approach, i.e., agents anticipate the changes in policy parameters of other agents (Zhang and Lesser, 2010; Foerster et al., 2018a,b) (see Figure 1). In this approach, first of all, agents should either have access to other agents' exact parameters or infer other agents' parameters from state-action trajectories (Foerster et al., 2018a). In many game settings, these parameters are obscured. This is problematic because when the size of the state space increases, the dimensionality of the parameter spaces increases as well, making the parameter inference problem computationally expensive. Furthermore, anticipating the changes in high-dimensional policy parameters is inefficient, whether the parameters are inferred or exact. Finally, policy parameter anticipation requires higher-order gradients with respect to policy parameters which is shown to be challenging in MARL (Foerster et al., 2018b; Lu et al., 2022). Current HOG methods mainly assume that the games are differentiable, i.e., agents have access to gradients and Hessians (Willi et al., 2022; Letcher et al., 2019) (Figure 1). When the games are non-differentiable, existing HOG methods employ the Stochastic Policy Gradient (SPG) theorem (Sutton and Barto, 2018) with on-policy sampling to compute the gradients with respect to the policy parameters (Foerster et al., 2018a,b). However, estimating higher-order gradients in SPG requires either analytical approximations – since the learning step for one agent in the standard SPG theorem is independent of other agents' parameters – or multi-stage sampling, which is inefficient and comes typically with high variance, making learning unstable (Foerster et al., 2018a,b). In this work, we aim to propose novel HOG methods that overcome the aforementioned limitations of existing HOG methods, making learning anticipation applicable to non-differentiable games with large state spaces.

To accomplish our goal, we propose Off-Policy Action Anticipation (OffPA2), a novel framework that approaches learning anticipation through action anticipation (see Figure 1). Specifically, the agents in OffPA2 anticipate the changes in actions of other agents during learning. Unlike policy parameter anticipation, action anticipation is performed in the action space whose dimensionality is generally lower than the policy parameter space in MARL games with large state spaces (Lowe et al., 2017; Peng et al., 2021). Furthermore, we employ the Deterministic Policy Gradient (DPG) theorem with off-policy sampling to estimate differentiable objective functions. Consequently, high-order gradients can be efficiently computed, while still following the standard Centralized Training and Decentralized Execution (CTDE) setting where agents can observe the other agents' actions during training (Lowe et al., 2017). We theoretically analyze our OffPA2 in terms of performance and time complexity. The proposed OffPA2 framework allows us to develop HOG methods that, unlike existing HOG methods, are applicable to non-differentiable games with large state spaces. To show this, we apply the principles of LA, LOLA, and HLA to our OffPA2 framework and develop the LA-OffPA2, LOLA-OffPA2, and HLA-OffPA2 methods, respectively. We compare our methods with existing HOG methods in well-controlled studies. By doing so, we demonstrate that the overall performance and efficiency of our proposed methods do not decrease with increasing the state-space size, unlike for existing HOG methods, where they get drastically worse. Finally, we compare our methods with the standard, DPG-based

MARL algorithms and highlight the importance of learning anticipation in MARL. Below, we summarize our contributions.

- We propose OffPA2, a novel framework that approaches learning anticipation through action anticipation, which makes HOG methods applicable to non-differentiable games with large state spaces. We provide theoretical analyses of the influence of our proposed action anticipation approach on performance and time complexity.
- Within our OffPA2 framework, we develop three novel methods, i.e., LA-OffPA2, LOLA-OffPA2, and HLA-OffPA2. We show that our methods outperform the existing HOG methods and state-of-the-art DPG-based approaches.

2. Related work

In many real-world MARL tasks, communication constraints during execution require the use of decentralized policies. In these cases, one reasoning tool is Agents Modeling Agents (AMA) (Albrecht and Stone, 2018), where agents explicitly model other agents to predict their behaviors. Although AMA traditionally assumes naïve opponents with no reasoning abilities (He et al., 2016; Hong et al., 2018), recent studies have extended AMA to further consider multiple levels of reasoning where each agent considers the reasoning process of other agents to make better decisions (Wen et al., 2019, 2020). For instance, Wen et al. (2019) proposed the probabilistic recursive reasoning (PR2) update rule for MARL agents to recursively reason about other agents' beliefs. However, in these approaches, agents do not take into account the learning steps of other agents, which has shown to be important in games where interaction among self-interested agents otherwise leads to worst-case outcomes (Foerster et al., 2018a). In Section 5, we conduct several experiments and compare our proposed methods with these approaches.

HOG methods, on the other hand, are a range of methods that employ higher-order gradients to anticipate the learning dynamics of other agents, playing a crucial role in tackling varied challenges within game-theoretical frameworks. Among these methods, LOLA and Higher-order LOLA (HOLA), as introduced by Foerster et al. (2018a), are designed to foster cooperation in the Iterated Prisoner's Dilemma (IPD) setting. Similarly, the Look-Ahead (LA) technique, proposed by Zhang and Lesser (2010), aims to guarantee convergence in cyclic games, while Stable Opponent Shaping (SOS), developed by Letcher et al. (2019), acts as a hybrid of LOLA and LA, integrating their benefits. Consistent LOLA (COLA), put forward by Willi et al. (2022), seeks to enhance consistency in opponent shaping. Hierarchical Learning Anticipation (HLA), proposed by Bighashdel et al. (2023), focuses on improving coordination among fully cooperative agents. Moreover, the advancement in training stability and convergence is furthered by methods such as Consensus Optimization (CO) by Mescheder et al. (2017), Symplectic Gradient Adjustment (SGA) by Balduzzi et al. (2018), Newton Optimization on Helmholtz Decomposition (NOHD) by Ramponi and Restelli (2021), and Learning to Play Games (L2PG) by Chen et al. (2023), each contributing uniquely to the domain.

As previously mentioned, these HOG methods are developed based on the policy parameter anticipation approach, necessitating access to or inference of other agents' exact parameters. However, policy parameter anticipation becomes inefficient when policy param-

eters are high-dimensional. Furthermore, most existing HOG methods have been primarily developed for differentiable games, wherein agents have access to the exact gradients or Hessians (see Figure 1). In the context of non-differentiable games, approximation methods are employed to estimate the higher-order gradients. Specifically, Foerster et al. (2018a) utilized the SPG framework to estimate gradients in LOLA. As the standard SPG operates independently of other agents' parameters, the authors combined Taylor expansions of the expected return with analytical derivations of the second-order gradients. A similar strategy was also adopted in NOHD (Ramponi and Restelli, 2021) for estimating the second-order gradients. Nevertheless, Foerster et al. (2018b) empirically demonstrated that this estimation is not stable in learning, as it requires much larger batch sizes. Moreover, this technique has been designed primarily for estimating up to second-order gradients, leaving its potential extension for higher-order gradient estimation – necessary for higher levels of learning anticipation – unclear. To address these challenges, Foerster et al. (2018b) proposed an infinitely differentiable Monte Carlo estimator, known as DiCE, to correctly optimize stochastic objectives with any order of gradients. Analogous to meta-learning, the agents in the DiCE framework reason about and predict the learning steps of their opponents using inner learning loops and update their parameters in outer learning loops. However, each learning loop for each agent necessitates a sampling stage, which is highly inefficient for high-order reasoning and games with large state spaces, i.e., beyond matrix games. In Section 5, we conduct a set of experiments to closely compare our proposed OffPA2 framework with DiCE.

3. Problem formulation and background

We formulate the MARL setup as a Markov Game (MG) (Littman, 1994). An MG is a tuple $(\mathcal{N}, \mathcal{S}, \{\mathcal{A}_i\}_{i\in\mathcal{N}}, \{\mathcal{R}_i\}_{i\in\mathcal{N}}, \mathcal{T}, \rho, \gamma)$, where $\mathcal{N}$ is the set of agents ($|\mathcal{N}| = n$), $\mathcal{S}$ is the set of states, and $\mathcal{A}_i$ is the set of possible actions for agent $i \in \mathcal{N}$. Agent i chooses its action $a_i \in \mathcal{A}_i$ through the stochastic policy network $\pi_{\theta_i} : \mathcal{S} \times \mathcal{A}_i \to [0, 1]$ parameterized by θ_i conditioning on the given state $s \in \mathcal{S}$. Given the actions of all agents, each agent i obtains a reward r_i according to its reward function $\mathcal{R}_i : \mathcal{S} \times \mathcal{A}_1 \times ... \times \mathcal{A}_n \to \mathbb{R}$. Given an initial state, the next state is produced according to the state transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A}_1 \times ... \times \mathcal{A}_n \times \mathcal{S} \to [0, 1]$. We denote an episode of horizon T as $\tau = (\{s^0, a_1^0, ..., a_n^0, r_1^0, ..., r_n^0\}, ..., \{s^T, a_1^T, ..., a_n^T, r_1^T, ..., r_n^T\})$, and the discounted return for each agent i at time step $t \leq T$ is defined by $G_i^t(\tau) = \sum_{l=t}^{T} \gamma^{l-t} r_i$ where γ is a predefined discount factor. The expected return given the agents' policy parameters approximates the state value function for each agent $V_i(s, \theta_1, ..., \theta_n) = \mathbb{E}[G_i^t(\tau|s^t = s)]$. The goal for each agent i is to find the policy parameters, θ_i, that maximize the expected return given the distribution of the initial state $\rho(s)$, denoted by the performance objective $J_i = \mathbb{E}_{\rho(s)} V_i(s, \theta_1, ..., \theta_n)$.

Naïve gradient ascend. In the naïve update rule, agents do not perform learning anticipation to update their policy parameters. More specifically, each naïve agent i maximizes its performance objective by updating its policy parameters in the direction of the objective's gradient

$$\nabla_{\theta_i} J_i = \mathbb{E}_{\rho(s)} \nabla_{\theta_i} V_i(s, \theta_1, ..., \theta_n). \tag{1}$$

Learning With Opponent-Learning Awareness (LOLA). Unlike naïve agents, LOLA agents modify their learning objectives by differentiating through the anticipated learning steps of the opponents (Foerster et al., 2018a). Given $n = 2$ for simplicity, a first-order LOLA agent (agent One) assumes a naïve opponent and uses policy parameter anticipation to optimize $V_1^{\text{LOLA}}(s, \theta_1, \theta_2 + \Delta\theta_2)$ where $\Delta\theta_2 = \mathbb{E}_{\rho(s)}\eta\nabla_{\theta_2}V_2(s, \theta_1, \theta_2)$ and $\eta \in \mathbb{R}^+$ is the prediction length. Using first-order Taylor expansion and by differentiating with respect to θ_1, the gradient adjustment for the first LOLA agent (Foerster et al., 2018a) is given by

$$\nabla_{\theta_1}V_1^{\text{LOLA}}(s, \theta_1, \theta_2 + \Delta\theta_2) \approx \nabla_{\theta_1}V_1 + (\nabla_{\theta_2\theta_1}V_1)^\intercal\Delta\theta_2 + \underbrace{(\nabla_{\theta_1}\Delta\theta_2)^\intercal\nabla_{\theta_2}V_1}_{\text{shaping}}, \tag{2}$$

where $V_1 = V_1(s, \theta_1, \theta_2)$. The rightmost term in the LOLA update allows for active shaping of the opponent's learning. This term has been proven effective in enforcing cooperation in various games, including IPD (Foerster et al., 2018a,b). The LOLA update can be further extended to non-naïve opponents, resulting in HOLA agents (Foerster et al., 2018a; Willi et al., 2022).

Look Ahead (LA). LA agents assume that the opponents' learning steps cannot be influenced, i.e., cannot be shaped (Zhang and Lesser, 2010; Letcher et al., 2019). In other words, agent One assumes that the prediction step, $\Delta\theta_2$, is independent of the current optimization, i.e., $\nabla_{\theta_1}\Delta\theta_2 = 0$. Therefore, the shaping term disappears, and the gradient adjustment for the first LA agent will be

$$\nabla_{\theta_1}V_1^{\text{LA}}(s, \theta_1, \theta_2 + \perp \Delta\theta_2) \approx \nabla_{\theta_1}V_1 + (\nabla_{\theta_2\theta_1}V_1)^\intercal\Delta\theta_2, \tag{3}$$

where $\perp$ prevents gradient flowing from $\Delta\theta_2$ upon differentiation.

Hierarchical Learning Anticipation (HLA). Unlike LOLA and LA, HLA is proposed to improve coordination in fully cooperative games with common interested agents (Bighashdel et al., 2023), i.e., $\mathcal{R}_i = \mathcal{R}_j = \mathcal{R}\ \forall i, j \in \mathcal{N}$ and, consequently, $V_i = V_j = V\ \forall i, j \in \mathcal{N}$. HLA randomly assigns the agents into hierarchy levels to specify their reasoning orders. In each hierarchy level, the assigned agent is a *leader* of the lower hierarchy levels and a *follower* of the higher ones, with two reasoning rules: 1) a leader knows the reasoning levels of the followers and is one level higher, and 2) a follower cannot shape the leaders and only follows their shaping plans. Concretely, if $n = 2$, and we assume that agent Two is the leader (HLA-L) and agent One is the follower (HLA-F), the gradient adjustment for the leader is:

$$\nabla_{\theta_2}V^{\text{HLA-L}}(s, \theta_1 + \Delta\theta_1, \theta_2) \approx \nabla_{\theta_2}V + (\nabla_{\theta_1\theta_2}V)^\intercal\Delta\theta_1 + (\nabla_{\theta_2}\Delta\theta_1)^\intercal\nabla_{\theta_1}V, \tag{4}$$

where $V = V(s, \theta_1, \theta_2)$ is the common value function, and $\Delta\theta_1 = \eta\nabla_{\theta_1}V$. The plan of the leader is to change its parameters $\bar{\theta}_2 = \theta_2 + \eta\nabla_{\theta_2}V^{\text{HLA-L}}(s, \theta_1 + \Delta\theta_1, \theta_2)$ in such a way that an optimal increase in the common value is achieved after its new parameters are taken into account by the follower. Therefore, the follower must follow the plan and adjust its parameters through

$$\nabla_{\theta_1}V^{\text{HLA-F}}(s, \theta_1, \bar{\theta}_2) \approx \nabla_{\theta_1}V + (\nabla_{\theta_2\theta_1}V)^\intercal\eta\nabla_{\theta_2}V^{\text{HLA-L}}(s, \theta_1 + \Delta\theta_1, \theta_2). \tag{5}$$

4. Approach

In this section, we propose OffPA2, a framework designed to enable the application of HOG methods to non-differentiable games with large state spaces. To solve the problems regarding policy parameter anticipation, we propose the novel approach of action anticipation, where agents anticipate the changes in actions of other agents during learning. Furthermore, we employ the DPG theorem with off-policy sampling to estimate differentiable objective functions. Consequently, high-order gradients can be efficiently computed. Our proposed OffPA2 complies with the standard Centralized Training and Decentralized Execution (CTDE) setting in DPG, where the agents during training have access to the actions of other agents (Lowe et al., 2017).

4.1 OffPA2: Off-policy action anticipation

We define a deterministic policy $\mu_{\theta_i} : \mathcal{S} \to \mathcal{A}_i$, parameterized by θ_i for each agent $i \in \mathcal{N}$. Let $Q_i(s, a_1, ..., a_n) = \mathbb{E}[G_i^t(\tau | s^t = s, a_i^t = a_i \forall i \in \mathcal{N})]$ denote the state-action value function, then we have $V_i(s, \theta_1, ..., \theta_n) = Q_i(s, \mu_{\theta_1}(s), ..., \mu_{\theta_n}(s))$. Furthermore, we define a stochastic behavior policy for each agent i as $\pi_{\theta_i}^b = \mu_{\theta_i} + SG$, where SG is a standard Gaussian distribution. Given the behavior policies, the policy parameters can be learned off-policy, from trajectories generated by the behavior policies, i.e., $\rho^B(s, \{a_i\}_{i\in\mathcal{N}}, \{r_i\}_{i\in\mathcal{N}}, s')$ where s and s' are consecutive states. Using the deterministic policy gradient theorem (Silver et al., 2014; Lowe et al., 2017), we can obtain the gradient of the performance objective for each naïve agent i as $\nabla_{\theta_i} J_i = \mathbb{E}_{\rho^\beta(s)} \nabla_{\theta_i} Q_i(s, \mu_{\theta_1}(s), ..., \mu_{\theta_n}(s))$.

Without the loss of generality, we consider two agents, i.e., $n = 2$, and we assume that agent One wants to anticipate the learning step of agent Two, who is a naïve learner. At each state $s \sim \rho^b(s)$, agent One anticipates the changes in the policy parameters of agent Two as $\Delta\theta_2(s) = \eta \nabla_{\theta_2} Q_2(s, \mu_{\theta_1}(s), \mu_{\theta_2}(s))$, i.e., policy parameter anticipation. Therefore, agent One updates its policy parameters in the direction of:

$$\nabla_{\theta_1} J_1 = \mathbb{E}_{\rho^\beta(s)} \nabla_{\theta_1} Q_1(s, \mu_{\theta_1}, \mu_{\theta_2 + \Delta\theta_2(s)}(s)), \tag{6}$$

Before presenting the main results, we introduce two key assumptions necessary for the development of our theoretical framework.

Assumption 1 (state-action value function independence) We assume that the state-action value function does not directly depend on the policy parameters. This assumption is standard in off-policy reinforcement learning, including deterministic and stochastic off-policy Actor-Critic algorithms (Silver et al., 2014; Degris et al., 2012), and is further justified in (Degris et al., 2012).

Assumption 2 (first-order Taylor expansion) We rely on a first-order Taylor expansion to approximate the changes in the state-action value function. The influence of this assumption on performance is addressed in Theorem 2.

Building upon these assumptions, we now present our main theorem which formalizes the concept of action anticipation within the established theoretical framework.

Theorem 1 (action anticipation) *Under Assumptions 1 and 2, the gradient of the performance objective for agent One, Eq. (6), can be approximated as:*

$$\nabla_{\theta_1} J_1 \approx \mathbb{E}_{\rho^{\beta}(s)} \nabla_{\theta_1} \mu_{\theta_1}(s) \nabla_{a_1} Q_1(s, a_1, a_2 + \Delta a_2)|_{a_1=\mu_{\theta_1}(s), a_2=\mu_{\theta_2}(s)}, \tag{7}$$

where

$$\Delta a_2 = \hat{\eta}_{1st} \nabla_{a_2} Q_2(s, a_1, a_2), \tag{8}$$

is the anticipated change of action, where $\hat{\eta}_{1st} = \eta \left\| \nabla_{\theta_2} \mu_{\theta_2(s)} \right\|^2 \in \mathbb{R}^+$ *is the projected prediction length.*

Proof. See Appendix A.1.

Theorem 1 indicates that agent One can anticipate the learning step of agent Two in the action space rather than the policy parameter space. This way of reasoning has two benefits. First, in the MARL games with large state spaces, the dimensionality of action space is significantly lower than that of the policy parameter space (Lowe et al., 2017; Peng et al., 2021). The justification is that large state spaces require more complex policy networks with more parameters to properly represent all possible states. Second, action anticipation, unlike policy parameter anticipation, complies with the standard centralized training and decentralized execution (CTDE) setting in DPG. In the standard CTDE setting, the agents during training have access to the centralized state-action value functions to train the decentralized policies. Consequently, the agents are informed of other agents' actions and can perform action anticipation during training. This is while in policy parameter anticipation, the agents need to additionally access the policy parameters of other agents.

4.1.1 Influence of action anticipation on performance

Our proposed action anticipation approach employs the first-order Taylor approximation to map the anticipated learning from the policy parameter space to the action space. In other words:

$$\mu_{\theta_2 + \Delta\theta_2(s)}(s) \approx a_2 + \Delta a_2, \tag{9}$$

where $a_2 = \mu_{\theta_2}(s)$ and $\Delta a_2 = \hat{\eta}_{1\text{st}} \nabla_{a_2} Q_2(s, a_1, a_2)$. In the theorem below, we show how this approximation influences the principles of HOG methods, which have been theoretically and experimentally researched throughout the literature (Zhang and Lesser, 2010; Foerster et al., 2018a; Letcher et al., 2019; Willi et al., 2022).

Theorem 2 *For a sufficiently small* $\hat{\eta}_{1st}$*, there exists an* $\eta' \in \mathbb{R}^+$ *such that*

$$\mu_{\theta_2 + \Delta\theta'_2(s)}(s) = a_2 + \Delta a_2, \tag{10}$$

where

$$\begin{aligned} &\Delta\theta'_2(s) = \eta' \nabla_{\theta_2} \mu_{\theta_2}(s) \nabla_{a_2} Q_2(s, a_1, a_2) \\ &a_2 = \mu_{\theta_2}(s) \\ &\Delta a_2 = \hat{\eta}_{1st} \nabla_{a_2} Q_2(s, a_1, a_2) \qquad \hat{\eta}_{1st} = \eta \left\| \nabla_{\theta_2} \mu_{\theta_2(s)} \right\|^2 \qquad \eta \in \mathbb{R}^+ \end{aligned} \tag{11}$$

Proof. See Appendix A.2.

Based on Theorem 2, action anticipation via first-order Taylor expansion, when employing a sufficiently small $\hat{\eta}_{\text{1st}}$, just scales the prediction length, as both η and η' are non-negative. This substantiates that, with an appropriately small $\hat{\eta}_{\text{1st}}$, the direction of action anticipation remains consistent with that of policy parameter anticipation, merely altering its scale.

4.1.2 Practical application of action anticipation

Following Theorem 1, it is crucial to clarify the mechanism of action anticipation and its implementation in our experiments. As indicated, action anticipation is based on the projected prediction length, $\hat{\eta}_{\text{1st}} = \eta \left\| \nabla_{\theta_2} \mu_{\theta_2(s)} \right\|^2$, which appears dependent on the policy parameters of agent Two, implying a potential need for parameter inference. Nevertheless, we can effectively circumvent this dependency by directly setting $\hat{\eta}_{\text{1st}}$ within our experimental framework, obviating the need to initially determine η and then compute $\hat{\eta}_{\text{1st}}$. The rationale and implications of directly setting the projected prediction length are justified in Theorem 2. Specifically, Theorem 2 demonstrates that a sufficiently small projected prediction length merely scales the prediction length without altering its direction. However, two considerations remain. The first involves determining the appropriate magnitude for $\hat{\eta}_{\text{1st}}$. In Appendix A.2, we establish that the upper bound for $\hat{\eta}_{\text{1st}}$, to ensure a positive prediction length in policy parameters, is ${}^{C_1(s)^2}/_{4|C_2(s)|}$, where

$$
\begin{aligned}
C_1(s) &= \left\| \nabla_{\theta_2} \mu_{\theta_2}(s) \right\|^2, \\
C_2(s) &= \frac{1}{2} \left(\nabla_{\theta_2} \mu_{\theta_2}(s) \right)^\intercal H_{\mu_{\theta_2}}(s) \nabla_{\theta_2} \mu_{\theta_2}(s) \left(\nabla_{a_2} Q_2(s, a_1, a_2) \right)^\intercal,
\end{aligned}
\tag{12}
$$

where $H_{\mu_{\theta_2}}(s)$ denotes the Hessian of μ_{θ_2} at state s. However, computing this bound precisely is impractical due to the potential unavailability of information about Agent Two's policy parameters. The second consideration arises from the fact that scaling the prediction length affects the convergence behavior of HOG methods, necessitating that the adjusted prediction length in policy parameters exhibits suitable convergence properties. Despite these considerations, it is pertinent to note that the selection of an appropriate prediction length is a common issue in all HOG methods, whether based on policy parameter anticipation or action anticipation (Zhang and Lesser, 2010; Foerster et al., 2018a). Similar to our findings in action anticipation, the choice of prediction length in HOG methods utilizing policy parameter anticipation significantly influences theoretical outcomes (Letcher et al., 2019), with its determination affected by game dynamics among other factors. Nonetheless, the standard practice in the empirical evaluation of these methods is to treat the prediction length as a tunable hyperparameter, determined empirically. Our study adopts this approach to address the aforementioned considerations by focusing on the empirical tuning of the projected prediction length rather than the conventional prediction length. In Section 5.2.3, we provide empirical insights into the impact of the projected prediction length on the convergence behavior of our proposed methods.

4.1.3 Computing higher-order gradients

The state-action value function in Eq (7) is generally unknown and non-differentiable. Similarly to the DPG-based algorithms (Silver et al., 2014; Lowe et al., 2017), we substitute a differentiable state-action value function $Q_i(s, a_1, ..., a_n; \omega_i)$, parameterized by ω_i, in place of the true state-action value function, i.e., $Q_i(s, a_1, ..., a_n; \omega_i) \approx Q_i(s, a_1, ..., a_n)$. The parameters of the state-action value function can be obtained by minimizing the Temporal Difference (TD) error, off-policy, from episodes generated by the behavior policies (Lowe et al., 2017):

$$\mathcal{L}(\omega_i) = \mathbb{E}_{\rho^b(s,\{a_i\}_{i\in\mathcal{N}},\{r_i\}_{i\in\mathcal{N}},s')}[(Q_i(s, a_1, ..., a_n; \omega_i) - y_i)^2], \tag{13}$$

where y_i is the TD target value:

$$y_i = r_i + \gamma Q'_i(s, a'_1, ..., a'_n)|_{a'_i=\mu'_i(s')\ \forall i\in\mathcal{N}}, \tag{14}$$

where Q'_i and μ'_i are the target state-action value and policy functions, respectively. The differentiability of objective functions in OffPA2 is particularly beneficial for HOG methods as they need to frequently compute the higher-order gradients to anticipate the agents' learning.

4.1.4 Influence of action anticipation on time complexity

Apart from the differentiability of objective functions, the action anticipation approach further reduces the gradient computation complexity as it requires the anticipated changes of actions, i.e., Δa_i, rather than the anticipated changes of policy parameters, i.e., $\Delta\theta_i$. To demonstrate this, assume that the policy and state-action value networks are multi-layer perceptrons (as is common in most experiments), with the same number of hidden layers, H, and neurons in each hidden layer, N. As both policy and state-action value functions are represented by neural networks, the time complexity of gradient anticipation follows the time complexity of backpropagation. The backpropagation time complexity for the aforementioned networks for an input state of size N_s and action of size N_a is (Lister and Stone, 1995):

- Backpropagation time complexity in the policy network: $O(N_sN + (H-1)N^2 + NN_a)$
- Backpropagation time complexity in the state-action value network: $O((N_s + N_a)N + (H-1)N^2 + N)$

Given that $N > N_s + N_a$, the time complexity of both networks can be upper bounded by $O(LN^2)$ where we defined $L = H + 1$. Now we assume that agent $i \in \mathcal{N}$ wants to anticipate the learning step of another agent $j \in \{\mathcal{N} - \{i\}\}$. In the case of policy parameter anticipation, agent i anticipates $\Delta\theta_j(s)$ as:

$$\Delta\theta_j(s) = \eta\nabla_{\theta_j}\mu_{\theta_j}(s)\nabla_{a_j}Q_j(s, a_1, ..., a_n)|_{a_j=\mu_{\theta_j}(s)}. \tag{15}$$

Therefore, the time complexity is $O(LN^2) \times O(LN^2)$, or in other words, $O(L^2N^4)$. In the case of action anticipation, on the other hand, agent i anticipates $\Delta a_j(s)$ as:

$$\Delta a_j = \hat{\eta}_{1st}\nabla_{a_j}Q_2(s, a_1, ..., a_n), \tag{16}$$

which has the complexity of $O(LN^2)$. Consequently, the time complexity is reduced by $O(LN^2)$.

4.2 OffPA2-based HOG methods

Having the OffPA2 framework, we can now develop HOG methods that are applicable to non-differentiable games with large state spaces. In the following sections, we develop LOLA-OffPA2, LA-OffPA2, and HLA-OffPA2 by applying the LOLA, LA, and HLA principles to our OffPA2 framework, respectively.

4.2.1 LOLA-OffPA2

As described in Section 3, LOLA agents predict and shape the learning steps of other agents to improve cooperation in non-team games. Given two agents ($n = 2$) for simplicity, the LOLA-OffPA2 agent (agent One) predicts and shapes the action of the opponent (agent Two) that is assumed by agent One to be naïve. Using first-order Taylor expansion, the gradient adjustment for the first LOLA-OffPA2 agent is given by

$$\begin{aligned}\nabla_{\theta_1} J_1^{\text{LOLA-OffPA2}} &= \mathbb{E}_{\rho^\beta(s)} \nabla_{\theta_1} \mu_{\theta_1}(s) \nabla_{a_1} Q_1(s, a_1, a_2 + \Delta a_2)|_{a_1=\mu_{\theta_1}(s), a_2=\mu_{\theta_2}(s)} \\ &\approx \mathbb{E}_{\rho^\beta(s)} \nabla_{\theta_1} \mu_{\theta_1}(s) \left(\nabla_{a_1} Q_1 + (\nabla_{a_2 a_1} Q_1)^\intercal \Delta a_2 + \underbrace{(\nabla_{a_1} \Delta a_2)^\intercal \nabla_{a_2} Q_1}_{\text{action shaping}}, \right)\end{aligned} \tag{17}$$

where

$$\begin{aligned}\Delta a_2 &= \hat{\eta}_{\text{1st}} \nabla_{a_2} Q_2(s, a_1, a_2)|_{a_1=\mu_{\theta_1}(s), a_2=\mu_{\theta_2}(s)} \\ Q_1 &= Q_1(s, a_1, a_2)|_{a_1=\mu_{\theta_1}(s), a_2=\mu_{\theta_2}(s)}.\end{aligned} \tag{18}$$

The rightmost term in the LOLA-OffPA2 update, i.e., Eq. (17), allows for active *action shaping* of the opponent.

In practice, we don't need to rely on Taylor expansion for the update rules in LOLA-OffPA2 as we can use an automatic differentiation engine, e.g., PyTorch autograd (Paszke et al., 2019), to directly compute the gradients. Algorithm 1 illustrates the LOLA-OffPA2 optimization framework for the case of n agents. At each state $s \sim \rho^b(s)$ the agent $i \in \mathcal{N}$ first anticipates the changes in actions of all agents $j \in \{\mathcal{N} - \{i\}\}$:

$$\Delta a_j = \hat{\eta}_{\text{1st}} \nabla_{a_2} Q_2(s, a_1, ..., a_n)|_{a_i=\mu_{\theta_i}(s)\ \forall i \in \mathcal{N}}. \tag{19}$$

Then, agent i updates its parameters θ_i by the following gradient adjustment:

$$\nabla_{\theta_i} J_i^{\text{LOLA-OffPA2}} = \mathbb{E}_{\rho^\beta(s)} \nabla_{\theta_i} \mu_{\theta_i}(s) \nabla_{a_i} Q_i(s, a_1 + \Delta a_1, ..., a_i, ..., a_n + \Delta a_n)|_{a_i=\mu_{\theta_i}(s)\ \forall i \in \mathcal{N}}. \tag{20}$$

Equation (20) denotes the update rule for *first-order* LOLA-OffPA2 agents that assume naïve opponents. However, we can also consider a *second-order* LOLA-OffPA2 agent that differentiates through the learning steps of first-order LOLA-OffPA2 opponents. Likewise, we can extend the update rules to include higher-order reasoning, as in HOLA (Foerster et al., 2016).

Algorithm 1: LOLA-OffPA2 for a set of n self-interested agents ($\mathcal{N}$).

Initialize μ_{θ_i}, Q_i, μ'_i, and Q'_i $\forall i \in \mathcal{N}$, and set $\hat{\eta}_{\text{1st}}$
for episode = 1 to max-num-episodes **do**
  Receive initial state s
  for $t = 1$ to max-episode-length **do**
    Select action a_i from $\pi^b_{\theta_i}(s)$ $\forall i \in \mathcal{N}$
    Execute actions $a = \{a_i\}_{\forall i \in \mathcal{N}}$ and observe rewards $r = \{r_i\}_{\forall i \in \mathcal{N}}$ and new state s'
    Store the tuple (s, a, r, s') in replay buffer $\mathcal{D}$
    Set $s = s'$

    Sample a random K tuples $\{(s^k, a^k, r^k, s'^k)\}_{k \in \{1,...,K\}}$ from $\mathcal{D}$
    for agent $i = 1$ to n **do**
      Set $y^k_i = r^k_i + \gamma Q'_i(s'^k, a'_1, ..., a'_n)|_{a'_h = \mu'_h(s'^k)}$, for $k \in \{1, ..., K\}$
      Update state-action value function Q_i by minimizing:

$$\mathcal{L}(\omega_i) = \frac{1}{K} \sum_{k \in \{1,...,K\}} [(Q_i(s^k, a^k_1, ..., a^k_n; \omega_i) - y^k_i)^2]$$

    end for
    Set $a^k_i = \mu_{\theta_i}(s^k)$, for $k \in \{1, ..., K\}$ and $i \in \mathcal{N}$
    for agent $i = 1$ to n **do**
      for agent $j = 1$ to n **do**
        if $j = i$ **then** continue
        Set $\Delta a^k_j = \hat{\eta}_{\text{1st}} \frac{\partial}{\partial a^k_j} Q_j(s^k, a^k_1, ..., a^k_n)$ for $k \in \{1, ..., K\}$
      end for
      Update policy parameters θ_i via: $\nabla_{\theta_i} J_i^{\text{LOLA-OffPA2}} =$

$$\frac{1}{K} \sum_{k \in \{1,...,K\}} \nabla_{\theta_i} \mu_{\theta_i}(s^k) \frac{\partial}{\partial a^k_i} Q_i(s^k, a^k_1 + \Delta a^k_1, ..., a^k_i, ..., a^k_n + \Delta a^k_n)$$

    end for
    Update Q'_i and μ'_i $\forall i \in \mathcal{N}$
  end for
end for

4.2.2 LA-OffPA2

Similarly to the LA principles (Zhang and Lesser, 2010; Letcher et al., 2019), LA-OffPA2 agents cannot shape the opponents' learning steps, i.e., they cannot shape the opponent's actions. Consequently, in the two-agent case, we have $\nabla_{a_1} \Delta a_2 = 0$. Using first-order Taylor expansion, the gradient adjustment for the first LA-OffPA2 agent (Foerster et al., 2018a) is given by

$$\begin{aligned} \nabla_{\theta_1} J_1^{\text{LA-OffPA2}} &= \mathbb{E}_{\rho^\beta(s)} \nabla_{\theta_1} \mu_{\theta_1}(s) \nabla_{a_1} Q_1(s, a_1, a_2 + \perp \Delta a_2)|_{a_1 = \mu_{\theta_1}(s), a_2 = \mu_{\theta_2}(s)} \\ &\approx \mathbb{E}_{\rho^\beta(s, \hat{a}_2)} \nabla_{\theta_1} \mu_{\theta_1}(s) \left(\nabla_{a_1} Q_1 + (\nabla_{a_2 a_1} Q_1)^\intercal \Delta a_2\right), \end{aligned} \tag{21}$$

where $\perp$ prevents gradient flowing from Δa_2 upon differentiation, and Δa_2 and Q_1 are defined in Eq. (18).

As in the case of LOLA-OffPA2, we can use an automatic differentiation engine to directly compute the gradients. Algorithm 2 in Appendix illustrates the LA-OffPA2 optimization framework for the case of n agents. At each state $s \sim \rho^b(s)$ the agent $i \in \mathcal{N}$ first anticipates the changes in actions of all agents $j \in \{\mathcal{N} - \{i\}\}$ using Eq. (19). Then, agent i updates its parameters θ_i by the following gradient adjustment:

$$\nabla_{\theta_i} J_i^{\text{LA-OffPA2}} = \mathbb{E}_{\rho^\beta(s)} \nabla_{\theta_i} \mu_{\theta_i}(s) \nabla_{a_i} Q_i(s, a_1 + \perp \Delta a_1, ..., a_i, ..., a_n + \perp \Delta a_n)|_{a_i = \mu_{\theta_i}(s)\ \forall i \in \mathcal{N}}. \tag{22}$$

4.2.3 HLA-OffPA2

As previously mentioned, HLA is proposed to improve coordination in fully cooperative games with common interested agents. To develop HLA-OffPA2, we first define $\mathcal{M} \subseteq \mathcal{N}$, as a set of size $m = |\mathcal{M}|$ common-interested agents with a common reward function $\mathcal{R} = \mathcal{R}_i = \mathcal{R}_j\ \forall i, j \in \mathcal{M}$. Without the loss of generality, we consider $\mathcal{M} = \mathcal{N}$, i.e., team games with a common state-action value function $Q(s, a_1, ..., a_m)$.

Hierarchy level assignment. Similarly to HLA, each agent in HLA-OffPA2 is first assigned to one of m levels, with level one as the lowest hierarchy level and level m as the highest. Although the hierarchy level assignment can be random as proposed by Bighashdel et al. (2023), we utilize the amount of influence that agents have on others, i.e., their shaping capacity, as the indicator for the agents' hierarchy levels (see Section 5.3.3 for experimental comparisons). We define the shaping capacity of the i^{th} agent, $\mathcal{SC}_i$, as the sum of the action shaping values with respect to all other agents j:

$$\mathcal{SC}_i = \sum_{j \in \{\mathcal{M} - \{i\}\}} \left\| (\nabla_{a_i} \Delta a_j)^\intercal \nabla_{a_j} Q(a_1, ..., a_m) \right\|, \tag{23}$$

where $\Delta a_j = \nabla_{a_j} Q(a_1, ..., a_m)$. The agent with the highest shaping capacity is assigned to the highest hierarchy level, and so on. As HLA-OffPA2 benefits from centralized learning, the only constraint for HLA reasoning rules remains the centralized state-action value function.

Update rules. After the hierarchy level assignment, the agents update their policy parameters in m update stages, i.e., one for each agent, and in a top-down fashion: the agent in the highest hierarchy level updates its policy parameters first. In each update stage, the corresponding agent 1) reasons about the actions of followers (if any) in a bottom-up fashion, i.e., it reasons about the agent in the lowest hierarchy level first, 2) updates its policy parameters, and 3) updates its action for the next update stage (if any).

If we set $m = 2$ and assume that agent Two is the leader (HLA-OffPA2-L) and agent One is the follower (HLA-OffPA2-F), the gradient adjustment for the leader is

$$\begin{aligned} \nabla_{\theta_2} J^{\text{HLA-OffPA2-L}} &= \mathbb{E}_{\rho^\beta(s)} \nabla_{\theta_2} \mu_{\theta_2}(s) \nabla_{a_2} Q(s, a_1 + \Delta a_1, a_2)|_{a_1 = \mu_{\theta_1}(s), a_2 = \mu_{\theta_2}(s)} \\ &\approx \mathbb{E}_{\rho^\beta(s)} \nabla_{\theta_2} \mu_{\theta_2}(s) \left(\nabla_{a_2} Q + (\nabla_{a_1 a_2} Q)^\intercal \Delta a_1 + (\nabla_{a_2} \Delta a_1)^\intercal \nabla_{a_1} Q \right), \end{aligned} \tag{24}$$

where

$$\begin{aligned} \Delta a_1 &= \hat{\eta}_{1\text{st}} \nabla_{a_1} Q(s, a_1, a_2)|_{a_1 = \mu_{\theta_1}(s), a_2 = \mu_{\theta_2}(s)} \\ Q &= Q(s, a_1, a_2)|_{a_1 = \mu_{\theta_1}(s), a_2 = \mu_{\theta_2}(s)}. \end{aligned} \tag{25}$$

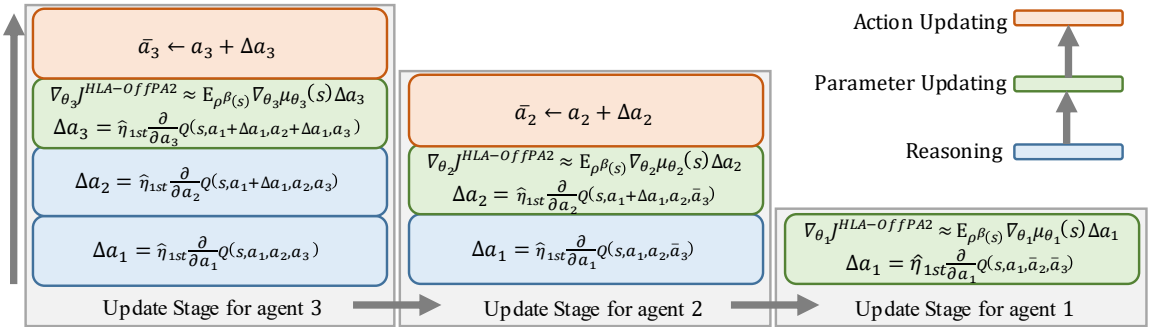


Figure 2: An example of the parameter update stages in HLA-OffPA2, for a game with three common-interested agents, where agent 1, agent 2, and agent 3 are assigned to hierarchy level 1, hierarchy level 2, and level 3, respectively.

The shaping plan of the leader is to change its actions as

$$\bar{a}_2 = a_2 + \hat{\eta}_{1\mathrm{st}} \nabla_{a_2} Q(s, a_1 + \Delta a_1, a_2)|_{a_1 = \mu_{\theta_1}(s), a_2 = \mu_{\theta_2}(s)}, \tag{26}$$

so that an optimal increase in the common state-action value is achieved after its new actions are taken into account by the follower. Therefore, the follower adjusts its parameters through

$$\begin{aligned} \nabla_{\theta_1} J^{\text{HLA-OffPA2-F}} &= \mathbb{E}_{\rho^{\beta}(s)} \nabla_{\theta_1} \mu_{\theta_1}(s) \nabla_{a_1} Q(s, a_1, \bar{a}_2)|_{a_1 = \mu_{\theta_1}(s)} \\ &\approx \mathbb{E}_{\rho^{\beta}(s)} \nabla_{\theta_1} \mu_{\theta_1}(s) \left(\nabla_{a_1} Q + (\nabla_{a_1 a_2} Q)^{\intercal} \hat{\eta}_{1\mathrm{st}} \nabla_{a_2} Q(s, a_1 + \Delta a_1, a_2) \right). \end{aligned} \tag{27}$$

As in the case of LOLA-OffPA2 and LA-OffPA2, we can use an automatic differentiation engine to directly compute the gradients. To clarify the update rules in HLA-OffPA2 for $m > 2$, we demonstrate an example of update stages for three common-interested agents in Figure 2, where agent One, agent Two, and agent Three are assigned to hierarchy level 1, hierarchy level 2, and hierarchy level 3, respectively. For the case of m agents, see the HLA-OffPA2 optimization framework in Algorithm 3 in Appendix.

5. Experiments

In this section, we conduct a set of experiments to accomplish two main goals: 1) to indicate the benefits of learning anticipation in a broader range of MARL problems, including non-differentiable games with large state spaces, and 2) to show the advantages of our proposed action anticipation approach with respect to policy parameter anticipation.

To accomplish the first goal, we compare our proposed methods with Multi-Agent Deep Deterministic Policy Gradient (MADDPG) (Lowe et al., 2017), configured with three state-of-the-art update rules: 1) standard update rule (Lowe et al., 2017), referred to as MADDPG, 2) Centralized Policy Gradient (CPG) update rule (Peng et al., 2021), referred to as CPG-MADDPG, and 3) Probabilistic Recursive Reasoning (PR2) update rule (Wen et al., 2019), referred to as PR2-MADDPG. To achieve our second goal, we compare our

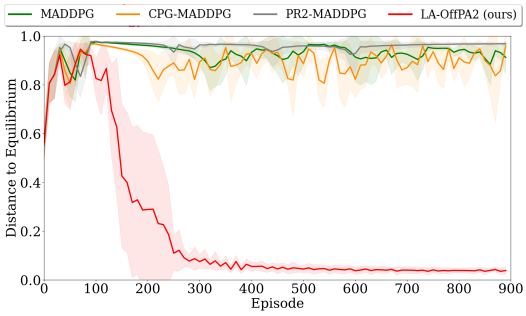


Figure 3: Learning curves in iterated rotational game in terms of the distance to the equilibrium point (↓).

Method	Distance to Equilibrium ↓
LA-DiCE	0.09±0.07
LA-OffPA2 (ours)	**0.03±0.02**

Method	Learning Anticipation Time Complexity ↓
LA-DiCE	1.06
LA-OffPA2 (ours)	**0.13**

Table 1: Comparison of LA principles in the frameworks of DiCE and our proposed OffPA2 in iterated rotational game.

proposed OffPA2-based methods with existing HOG methods that are capable of solving non-differentiable games (see Figure 1). Specifically, we compare LOLA-OffPA2 and LA-OffPA2 with LOLA-DiCE and LA-DiCE, respectively. Prior work (Foerster et al., 2018b) has shown that LOLA-DiCE significantly outperforms LOLA in IPD, and, consequently, we do not compare LOLA-OffPA2 with LOLA. As both LOLA-DiCE and LA-DiCE are based on policy parameter anticipation, with these experiments, we can highlight the benefits of our novel action anticipation approach. Similarly to the implementation of Foerster et al. (2018b), agents in DiCE can access the policy parameters of other agents. Since the original HLA method can only be applied to differentiable games, we do not compare HLA-OffPA2 with HLA.

Unless mentioned otherwise, we evaluate the performance of methods based on (normalized) Average Episode Reward (AER), with higher values indicating better performance. Furthermore, we assess the efficiency of HOG methods based on the Learning Anticipation Time Complexity (LATC), which for HOG method $\mathcal{H}$ is computed as:

$$\text{LATC}(\mathcal{H}) = \frac{\text{per iteration training time of } \mathcal{H}}{\text{per iteration training time of the naïve version of } \mathcal{H}} - 1 \geq 0 \qquad (28)$$

where the naïve version of $\mathcal{H}$ does not perform learning anticipation. The lower values of LATC indicate better efficiency, and LATC $= 0$ implies that learning anticipation adds zero time complexity to the algorithm. In the following sections, we separately evaluate our proposed methods and discuss the results (see Appendix B for implementation details).

5.1 Evaluation of LA-OffPA2

We evaluate the methods on the non-differentiable version of the rotational game proposed by Zhang and Lesser (2010), and we refer to it as the Iterated Rotational Game (IRG). IRG is a one-state, two-agent, one-action (continuous) matrix game with the rewards depicted in Table 2 (for two discrete actions). However, the agents do not have access to the reward table and can only receive a reward for their joint actions. Each agent $i \in \{1, 2\}$ must

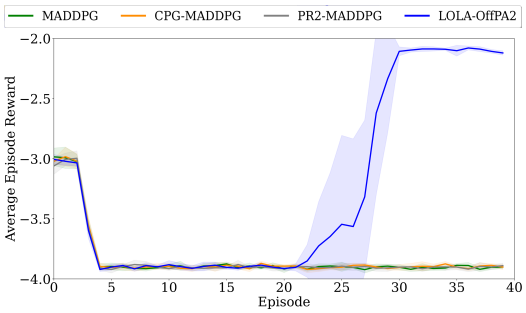


Figure 4: Learning curves in iterated prisoner's dilemma in terms of the average episode reward (↑).

Method	Average Episode Reward ↑
LOLA-DiCE	-2.16±0.12
LOLA-OffPA2 (ours)	**-2.08±0.02**

Method	Learning Anticipation Time Complexity ↓
LOLA-DiCE	1.18
LOLA-OffPA2 (ours)	**0.16**

Table 3: Comparison of LOLA principles in the frameworks of DiCE and our proposed OffPA2 in iterated prisoner's dilemma.

choose a 1-D continuous action ($0 \leq a_i \leq 1$ representing the probability of taking two discrete actions. The game has a unique equilibrium point at $a_1 = a_2 = 0.5$, which is also the fixed point of the game. The rotational game was originally proposed to demonstrate the circular behavior that can emerge if the agents follow the naïve gradient updates. LA agents, on the other hand, can quickly converge to the equilibrium point by considering their opponent's parameter adjustment. We evaluate the performances of methods based on the Distance to Equilibrium (DtE), which is the Euclidean distance between current actions and the equilibrium point.

Figure 3 demonstrates the learning curves for LA-OffPA2 and other, state-of-the-art MADDPG-based algorithms. From this figure, we find that LA-OffPA2 is the only method that can converge to equilibrium actions. These results highlight the importance of learning anticipation in IRG. To further show the effectiveness of LA-OffPA2, we compare our LA-OffPA2 method with LA-DiCE (Foerster et al., 2018b) and report the results in Table 1. Looking at DtE results in Table 1, it is apparent that both methods can solve the games, with LA-OffPA2 achieving slightly better results. However, if we compare the methods regarding learning anticipation time complexity (LATC), it is clear that LA-OffPA2 is significantly more efficient than LA-DiCE, which indicates the benefits of our OffPA2 framework with respect to DiCE.

	discrete action 1	discrete action 2
discrete action 1	$(0, 3)$	$(3, 2)$
discrete action 2	$(1, 0)$	$(2, 1)$

Table 2: Rewards in iterated rotational game.

5.2 Evaluation of LOLA-OffPA2

5.2.1 Iterated prisoner's dilemma

Iterated Prisoner's Dilemma (IPD) (Foerster et al., 2018a) is a five-state, two-agent, two-action game with the reward matrices depicted in Table 4. Each agent must choose between two discrete actions (cooperate or defect). The game is played for 150 time steps ($T =$

150). In the one-shot version of the game, there is only one Nash equilibrium for the agents (Defect, Defect). In the iterated games, (Defect, Defect) is also a Nash equilibrium. However, a better equilibrium is Tit-For-Tat (TFT), where the players start by cooperating and then repeat the previous action of the opponents. The LOLA agents can shape the opponent's learning to encourage cooperation and, therefore, converge to TFT (Letcher et al., 2019). We evaluate the methods' performances based on the Averaged Episode Reward (AER).

In Figure 4, we depict the learning curves for LOLA-OffPA2 and the MADDPG-based methods. From this figure, we find that only LOLA-OffPA2 can solve the game, which once again highlights the importance of learning anticipation. Additionally, we compared the performance of LOLA-OffPA2 with LOLA-DiCE (Foerster et al., 2018b), which is designed specifically for this game, and we reported the results in Table 3. Although both methods demonstrate high values of AER, our LOLA-OffPA2 is significantly more efficient as its LATC value is much lower than that of LOLA-DiCE.

	Cooperate	Defect
Cooperate	$(-1, -1)$	$(-3, 0)$
Defect	$(0, -3)$	$(-3, -3)$

Table 4: Rewards in iterated prisoner's dilemma.

5.2.2 Multi-level Exit-Room game

In the second experiment, we evaluate the capability of LOLA-OffPA2 in games with large state spaces, which is the envisioned use case for our framework. Inspired by Vinitsky et al. (2019), we propose an Exit-Room game with three levels of complexity (see Figure 5). The Exit-Room game is a grid-world variant of the IPD, with two agents (blue and red) and 15^{2l} states where $l \in \{1, 2, 3\}$ is the complexity level of the game. The agents should cooperate and move toward the exit doors on the right. However, they are tempted to exit through the left doors and, in some cases, not exiting at all. In level 1, the agents have three possible actions (*move-left*, *move-right*, or *do nothing*), and the reward is computed as Vinitsky et al. (2019):

$$\begin{aligned} \text{reward}_C &= \lambda_C(\text{cooperation}_{self} + \text{cooperation}_{opponent}) \\ \text{reward}_D &= \lambda_D(1 - \text{cooperation}_{self}) \\ \text{reward} &= \text{reward}_C + \text{reward}_D, \end{aligned} \tag{29}$$

where λ_C and λ_D are some constants, and $\text{cooperation}_{self}$ and $\text{cooperation}_{opponent}$ are the normalized distances of the agent and its opponent to the right door, respectively. In levels 2 and 3, the agents have additional *move-up* and *move-down* actions. In level 3, the door positions are randomly located, resulting in more complex interactions among the agents. In addition to the reward in Eq. (29), the agents receive an additional reward for approaching the doors in levels 2 and 3. Each agent receives four 90×90 RGB images representing the state observations of the last four time steps.

Figure 6 compares the learning curves of LOLA-OffPA2 and MADDPG-based methods in terms of Normalized Average Episode Reward (NAER), which is the AER value normalized between the highest and lowest episode rewards in each game level. In Figure 6, we can clearly see that our LOLA-OffPA2 significantly outperforms the other methods, similarly in the IPD matrix game.

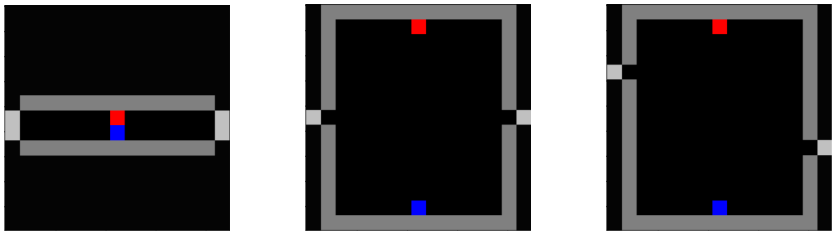
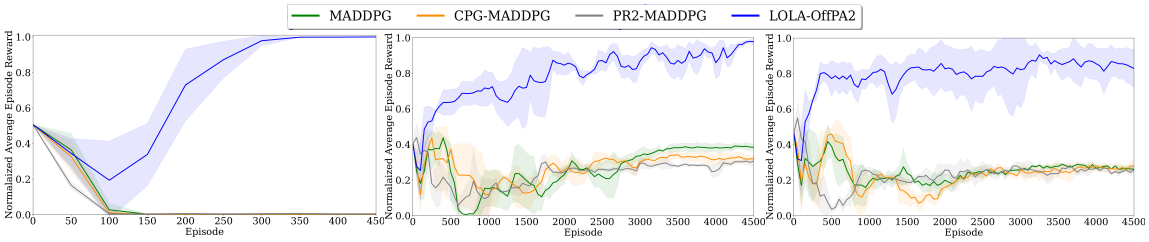


Figure 5: State observation in the Exit-Room game, level one (left), level two (middle), and level three (right).



(a) Exit-Room level one. (b) Exit-Room level two. (c) Exit-Room level three.

Figure 6: Learning curves in different complexity levels of the exit-room game in terms of the normalized average episode reward (↑).

To highlight the benefits of our proposed method with respect to existing HOG methods, we compare our LOLA-OffPA2 with LOLA-DiCE in terms of performance (by comparing NAER) and training efficiency (by comparing LATC) in Table 5. Observing Table 5, it is apparent that LOLA-DiCE fails to acquire the highest rewards, particularly for the second and third levels of the game, where the state-space size is increased. The reason can be attributed to the fact that when the size of the state space increases, the LOLA-DiCE agents fail to properly approximate the higher-order gradients via sampling, and, consequently, cannot perform learning anticipation to achieve a higher reward. This is while our proposed LOLA-OffPA2 performs better in all levels of the game in terms of NAER, and scaling from naïve to higher-order reasoning is significantly more efficient for LOLA-OffPA2 than LOLA-DiCE. This emphasizes that we have overcome the limitations of HOG methods described in Section 1.

5.2.3 Influence of the projected prediction length

In this section, we empirically show that by directly changing the projected prediction length, i.e., $\hat{\eta}_{1\text{st}}$, we can tune the resulting prediction length in the policy parameter space, i.e., η', and consequently control the convergence behavior. We conduct an experimental study to analyze the influence of $\hat{\eta}_{1\text{st}}$ on the convergence behavior of LOLA-OffPA2 in the level-three Exit-room game (See Figure 7). The experiments are repeated four times, and the mean results are reported in terms of NEAR in Figure 7. It is clear from Figure 7 that by tuning $\hat{\eta}_{1\text{st}}$, we can alter the convergence behavior of LOLA-OffPA2. Furthermore, Figure 7 demonstrates that low values of the projected prediction length ($\hat{\eta}_{1\text{st}} = 0.1$) cancel

	Normalized Average Episode Reward ↑			Learning Anticipation Time Complexity ↓				
Methods	$l=1$	$l=2$	$l=3$	Naïve	1st-order	2nd-order	3rd-order	4th-order
LOLA-DiCE	0.91±0.04	0.68±0.06	0.56±0.12	0.00	1.39	2.74	4.12	5.41
LOLA-OffPA2 (ours)	**1.00±0.00**	**0.99±0.01**	**0.93±0.03**	0.00	**0.24**	**0.47**	**0.69**	**0.94**

Table 5: Comparisons of LOLA-DiCE with our proposed LOLA-OffPA2 in the Exit-Room game, in terms of performance (normalized average return in different game levels) and efficiency (learning anticipation time complexity in different reasoning levels).

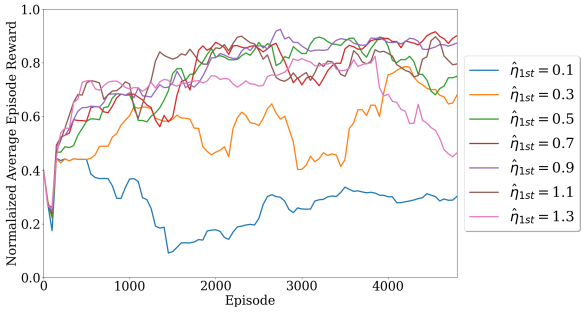


Figure 7: The influence of the projection prediction length ($\hat{\eta}_{1st}$) on the convergence behavior of LOLA-OffPA2 in the level-three Exit-room game.

the effect of learning anticipation in OffPA2, and high values of the projected prediction length ($\hat{\eta}_{1st} = 1.3$) lead to instability of the LOLA-OffPA2 which can be attributed to our findings in Theorem 2.

An important note to consider here is that

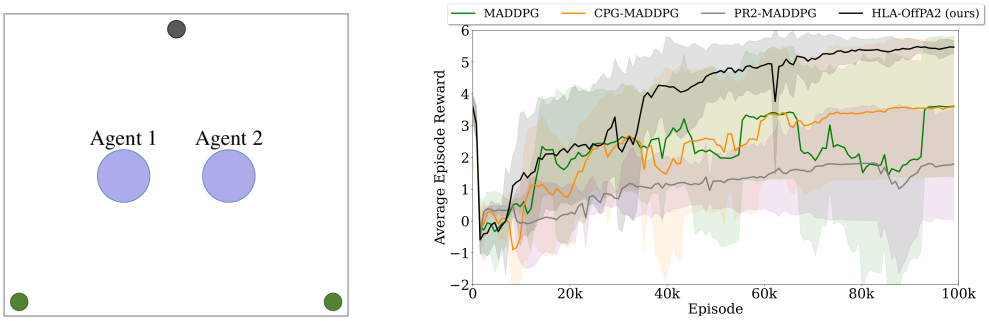
5.3 Evaluation of HLA-OffPA2

5.3.1 Particle-coordination game

To demonstrate the coordination capability of HLA-OffPA2, we propose the Particle-Coordination Game (PCG) in the Particle environment (Lowe et al., 2017). As shown in Figure 8a, each one of the two agents (purple circles) should select and approach one of the three landmarks (one gray and two green circles). Landmarks are selected based on the closest distance between the agent and the landmarks. Suppose the agents select and approach the same landmark. In that case, they receive global (by selecting the green landmarks) or local (by selecting the gray landmark) optimal rewards. They will receive an assigned miscoordination penalty if they select and approach different landmarks (see Table 6). Each agent receives a 10-D state observation vector (velocity and position information of the agent, i.e., 4-D, and location information of the landmarks, i.e., 6-D) and selects a 5-D, one-hot vector representing one of the five discrete

	Green (L)	Gray	Green (R)
Green (L)	2	0	-20
Gray	0	0.4	0
Green (R)	-20	0	2

Table 6: Rewards in particle-coordination game.



(a) Game schematics (b) Learning curves

Figure 8: Particle-coordination game. (a) Schematics of the game with two agents, i.e., purple circles, and three landmarks, i.e., gray and green circles. (b) Learning curves in terms of average episode reward (↑). Best viewed in color.

	Particle Environment			Mujoco Environment		
	Cooperative Navigation	Physical Deception	Predator-Prey	Half-Cheetah	Walker	Reacher
Observation	18-D	10-D (8-D)	14-D (12-D)	11-D	11-D	8-D
Action	5-D	5-D (5-D)	5-D (5-D)	3-D	3-D	1-D
Action type	discrete	discrete	discrete	continuous	continuous	continuous
Horizon (step)	25	25	25	100	300	50

Table 7: Specifications in the standard multi-agent games. In the mixed environments, the dimensions are reported as "d_1 (d_2)" where d_1 is the dimension for common-interested agents and d_2 is the dimension for self-interested ones.

actions: *move-right*, *move-left*, *move-up*, *move-down*, and *stay*. The horizon is set to 25. The game is quite challenging as the agents cannot see the locations of each other, and they can be subject to miscoordination.

In Figure 8b, we depict the learning curves for our HLA-OffPA2 and other MADDPG-based methods in terms of the average episode reward (AER). As demonstrated, MADDPG-based methods have relatively high variance in the convergence points. For instance, the MADDPG agents, which heavily benefit from exploration and randomness during policy parameter updates, can occasionally converge to the global optimal point with the highest AER. However, the high miscoordination penalty forces the agents to choose the safest option (gray landmark), which leads to a zero reward in the worst-case scenario. From this figure, it is clear that our HLA-OffPA2 is the only method that consistently converges to the global optimum of the game, which is consistent with the reported results for HLA in fully-cooperative differentiable games (Bighashdel et al., 2023).

5.3.2 Standard multi-agent games

For the final experiments, we evaluate our HLA-OffPA2 and MADDPG-based methods in three Particle environment games (Lowe et al., 2017): 1) Cooperative Navigation with three common-interested agents, 2) Physical Deception with two common-interested agents

	↑NAER in Particle Environment			↑NAER in Mujoco Environment		
Methods	Cooperative Navigation	Physical Deception	Predator-Prey	Half-Cheetah	Walker	Reacher
DDPG (LB)	0.00	0.00	0.00	0.00	0.00	0.00
C-MADDPG (UB)	1.00	1.00	1.00	1.00	1.00	1.00
MADDPG	0.77	0.61	0.21	0.86	0.45	0.02
CPG-MADDPG	0.78	0.67	0.18	0.88	0.46	0.05
PR2-MADDPG	0.78	0.54	0.08	0.85	0.45	0.01
HLA-OffPA2 (ours)	**0.88**	**0.83**	**0.44**	**0.94**	**0.67**	**0.42**

Table 8: Comparisons of methods in terms of the Normalized Average Episode Reward (NAER) for common-interested agents. LB: Lower Bound. UB: Upper Bound.

	↑NAER in Particle Environment			↑NAER in Mujoco Environment		
Methods	Cooperative Navigation	Physical Deception	Predator-Prey	Half-Cheetah	Walker	Reacher
HLA-OffPA2 (F)	0.85	0.80	0.38	0.92	0.63	0.38
HLA-OffPA2	**0.88**	**0.83**	**0.44**	**0.94**	**0.67**	**0.42**

Table 9: Ablation study on the hierarchy level assignments in our HLA-OffPA2 method.

and one self-interested agent, and 3) Predator-Prey with two common-interested (predator) agents and one self-interested (prey) agent. Furthermore, we compare the methods in three games within the multi-agent Mujoco environment (Peng et al., 2021): 1) two-agent Half-Cheetah, 2) two-agent Walker, and 3) two-agent Reacher. In the mixed environments (Physical Deception and Predator-Prey), we have employed the MADDPG method for the self-interested agents in all experiments. Games' specifications are reported in Table 7. We created separate validation and test sets for each game that included 100 and 300 randomly generated scenarios, respectively. In each game, we save the models that have the best performance on the validation set and test them on the test set to report the results. All experiments are repeated five times, and the mean results are reported in Table 8 in terms of the Normalized AER (NAER). The normalization is done between the single-agent variant of MADDPG, i.e., DDPG (Lillicrap et al., 2016), and a fully centralized (in learning and execution) variant of MADDPG, referred to as C-MADDPG. As all of the games are non-differentiable, the original HLA method is no longer applicable.

In Table 8, we observe that our proposed HLA-OffPA2 consistently and significantly outperforms all the state-of-the-art MADDPG-based methods. Again, these results confirm that learning anticipation, and in particular, our proposed HLA-OffPA2 improves coordination among common-interested agents, leading to better results.

5.3.3 Ablation study on hierarchy-level assignments

We have additionally conducted an ablation study on the hierarchy-level assignment in the HLA-OffPA2. Rather than iteratively sorting the agents based on their shaping capacities through Eq. (23), we randomly assigned the agents to hierarchy levels in the beginning and fixed the hierarchy levels throughout the optimization. This variant of the HLA-OffPA2, i.e., referred to as HLA-OffPA2 (F), is evaluated and compared in Table 9. As can be seen, using the proposed sorting strategy based on the shaping capacities of the agents, as done in our HLA-OffPA2, constantly improves performance.

6. Limitations and Future Directions

Despite the promising empirical results achieved with our proposed OffPA2 framework, acknowledging certain limitations inherent in our approach is crucial. As previously mentioned, our experimental setup utilized a fixed projected prediction length due to its ease of tuning and the consistency it offered across various trials. However, this fixed approach does not fully address the non-linear relationship between the projected prediction length and the actual prediction length, which is influenced by several factors, including game dynamics. This non-linearity becomes more pronounced in complex games with large state-action spaces, where even a small adjustment in the projected prediction length can significantly alter the prediction length in the policy parameter space, thereby affecting the algorithm's convergence behavior. We have observed instances where certain values of the projected prediction length resulted in performance inferior to that of methods not utilizing learning anticipation.

While the initial solution involved empirical tuning of the fixed projected prediction length, pursuing more sophisticated approaches promises significant improvements. Accordingly, we propose exploring a dynamic scheduling approach for the projected prediction length as a promising direction for future research. However, it should be noted that standard scheduling methods, such as decaying the projected prediction length over time (e.g., using a decay rate of $1/t$, where t represents time or iterations), do not effectively account for the game dynamics and may result in undesirable prediction lengths. Consequently, we advocate for a more sophisticated dynamic scheduling approach that better adapts to game dynamics. Such an approach can potentially lead to more stable prediction lengths over time and enhancements in the adaptability and robustness of learning algorithms. Investigating these mechanisms could provide valuable insights into optimizing our proposed HOG methods for complex MARL challenges.

7. Conclusion

In this paper, we proposed the OffPA2 framework that enables the applicability of HOG methods to non-differentiable games with large state spaces. To indicate the advantages of our framework, we developed three novel HOG methods, LA-OffPA2, LOLA-OffPA2, and HLA-OffPA2. By conducting several experiments, we demonstrated that our proposed methods outperform the existing HOG methods in terms of performance and efficiency. Furthermore, we extensively compared our methods with various DPG-based methods, which do not use learning anticipation, and we signified that learning anticipation improves coordination among agents and leads to higher rewards. As a result of our framework, the benefits of learning anticipation can now be used in many more MARL problems.

Acknowledgments

This work made use of the Dutch national e-infrastructure with the support of the SURF Cooperative using grant no. EINF-6816, which is financed by the Dutch Research Council (NWO).

Appendix

Appendix A. Proofs

A.1 Proof of Theorem 1

At each state $s \sim \rho^b(s)$, agent One anticipates the changes in the policy parameters of agent Two, i.e., $\Delta\theta_2(s) = \eta\nabla_{\theta_2} Q_2(s, \mu_{\theta_1}(s), \mu_{\theta_2}(s))$, and updates the policy parameters in the direction of:

$$\nabla_{\theta_1} J_1 = \mathbb{E}_{\rho^\beta(s)} \nabla_{\theta_1} Q_1(s, \mu_{\theta_1}, \mu_{\theta_2+\Delta\theta_2(s)}(s)), \tag{30}$$

In order to prove Theorem 1, we need two assumptions: 1) neglecting the direct dependencies of the state-action value function on the policy parameters, and 2) first-order Taylor expansion. The first assumption is standard in off-policy reinforcement learning, including both deterministic and stochastic off-policy Actor-Critic algorithms (Silver et al., 2014; Degris et al., 2012), and justification to support this assumption is provided in (Degris et al., 2012). As to the second assumption, please refer to Theorem 2 to see how this assumption influences the performance. Given the first assumption, we can rewrite Eq. (30) as

$$\nabla_{\theta_1} J_1 = \mathbb{E}_{\rho^\beta(s)} \nabla_{\theta_1} \mu_{\theta_1}(s) Q_1(s, a_1, \tilde{a}_2)|_{a_1=\mu_{\theta_1}(s), \tilde{a}_2=\mu_{\theta_2+\Delta\theta_2(s)}(s)}, \tag{31}$$

Similarly, we can set $\Delta\theta_2(s) = \eta\nabla_{\theta_2}\mu_{\theta_2}(s)\nabla_{a_2} Q_2(s, a_1, a_2)|_{a_2=\mu_{\theta_2}(s)}$. We now use the first-order Taylor expansion to map the anticipated gradient information to the action space:

$$\begin{aligned} \tilde{a}_2 &= \mu_{\theta_2+\Delta\theta_2(s)}(s) \\ &\approx \mu_{\theta_2}(s) + (\Delta\theta_2(s))^\intercal \nabla_{\theta_2}\mu_{\theta_2}(s), \end{aligned} \tag{32}$$

Given the definition of $\Delta\theta_2(s)$, we have:

$$\begin{aligned} \tilde{a}_2 &\approx \mu_{\theta_2}(s) + (\eta\nabla_{\theta_2}\mu_{\theta_2}(s) (\nabla_{d_2} Q_2(s, a_1, a_2))^\intercal)^\intercal \nabla_{\theta_2}\mu_{\theta_2}(s) \\ &= \mu_{\theta_2}(s) + \nabla_{a_2} Q_2(s, a_1, a_2) (\eta\nabla_{\theta_2}\mu_{\theta_2}(s))^\intercal \nabla_{\theta_2}\mu_{\theta_2}(s) \\ &= \mu_{\theta_2}(s) + \nabla_{a_2} Q_2(s, a_1, a_2)\eta \left\|\nabla_{\theta_2}\mu_{\theta_2(s)}\right\|^2 \\ &= \mu_{\theta_2}(s) + \nabla_{a_2} Q_2(s, a_1, a_2)\hat{\eta}_{1\text{st}}, \end{aligned} \tag{33}$$

where $\|.\|$ is the l^2-norm and we have defined the projected prediction length $\hat{\eta}_{1\text{st}} = \eta \left\|\nabla_{\theta_2}\mu_{\theta_2(s)}\right\|^2$, since $\left\|\nabla_{\theta_2}\mu_{\theta_2(s)}\right\|^2$ is a positive number and independent of θ_1. Therefore:

$$\tilde{a}_2 \approx a_2 + \Delta a_2. \tag{34}$$

where we have defined $\Delta a_2 = \hat{\eta}_{1\text{st}}\nabla_{a_2} Q_2(s, a_1, a_2)$ Replacing Eq. 34 in Eq. 30 yields:

$$\nabla_{\theta_1} J_1^{\text{LA}} \approx \mathbb{E}_{\rho^\beta(s)} \nabla_{\theta_1}\mu_{\theta_1}(s)\nabla_{a_1} Q_1(s, a_1, a_2 + \Delta a_2)|_{a_1=\mu_{\theta_1}(s), a_2=\mu_{\theta_2}(s)}, \tag{35}$$

and consequently, Theorem 1 is proved.

A.2 Proof of Theorem 2

In order to prove Theorem 2, we first need to show that:

Lemma 3 *If the anticipated changes are mapped from the policy parameter space to the action space using full-order Taylor expansion, there exists $\hat{\eta}_{full} \in \mathbb{R}$ such that*

$$\mu_{\theta_2+\Delta\theta_2}(s) = \mu_{\theta_2}(s) + \hat{\eta}_{full}\nabla_{a_2}Q_2(s,a_1,a_2), \tag{36}$$

where

$$\begin{aligned} &\Delta\theta_2 = \eta\nabla_{\theta_2}\mu_{\theta_2}(s)\nabla_{a_2}Q_2(s,a_1,a_2) \\ &\eta \in \mathbb{R}^+ \end{aligned} \tag{37}$$

Proof. The full-order Taylor expansion of the anticipated gradient yields:

$$\mu_{\theta_2+\Delta\theta_2}(s) = \mu_{\theta_2}(s) + (\Delta\theta_2)^\intercal\nabla_{\theta_2}\mu_{\theta_2}(s) + \frac{1}{2}(\Delta\theta_2)^\intercal H_{\mu_{\theta_2}}(s)\Delta\theta_2 + O(\|\Delta\theta_2\|^3), \tag{38}$$

where $H_{\mu_{\theta_2}}(s)$ denotes the Hessian of μ_{θ_2} at s. Given that:

$$\Delta\theta_2 = (\eta\nabla_{\theta_2}\mu_{\theta_2}(s)\nabla_{a_2}Q_2(s,a_1,a_2))^\intercal, \tag{39}$$

we have

$$\begin{aligned} \mu_{\theta_2+\Delta\theta_2}(s) =& \mu_{\theta_2}(s) + \nabla_{a_2}Q_2(s,a_1,a_2)\eta\left\|\nabla_{\theta_2}\mu_{\theta_2(s)}\right\|^2 \\ &+ \frac{1}{2}\nabla_{a_2}Q_2(s,a_1,a_2)\eta^2\left(\nabla_{\theta_2}\mu_{\theta_2}(s)\right)^\intercal H_{\mu_{\theta_2}}(s)\nabla_{\theta_2}\mu_{\theta_2}(s)\left(\nabla_{a_2}Q_2(s,a_1,a_2)\right)^\intercal \\ &+ O(\eta^3). \end{aligned} \tag{40}$$

By defining

$$\begin{aligned} C1(s) &= \left\|\nabla_{\theta_2}\mu_{\theta_2(s)}\right\|^2 \\ C_2(s) &= \frac{1}{2}\left(\nabla_{\theta_2}\mu_{\theta_2}(s)\right)^\intercal H_{\mu_{\theta_2}}(s)\nabla_{\theta_2}\mu_{\theta_2}(s)\left(\nabla_{a_2}Q_2(s,a_1,a_2)\right)^\intercal, \end{aligned} \tag{41}$$

we have:

$$\mu_{\theta_2+\Delta\theta_2}(s) = \mu_{\theta_2}(s) + \nabla_{a_2}Q_2(s,a_1,a_2)(\eta C_1(s) + \eta^2 C_2(s) + O(\eta^3)), \tag{42}$$

Given the definition of $C_2(s)$ and the dimension constraint implied by $\nabla_{a_2}Q_2(s,a_1,a_2)$, it can be concluded that $C_1(s) \in \mathbb{R}^+$ and $C_2(s) \in \mathbb{R}$. Therefore:

$$\mu_{\theta_2+\Delta\theta_2}(s) = \mu_{\theta_2}(s) + \hat{\eta}_{\text{full}}\nabla_{a_2}Q_2(s,a_1,a_2), \tag{43}$$

Algorithm 2: LA-OffPA2 for a set of n self-interested agents ($\mathcal{N}$).

Initialize μ_{θ_i}, Q_i, μ'_i, and Q'_i $\forall i \in \mathcal{N}$, and set $\hat{\eta}_{\text{1st}}$
for episode = 1 to max-num-episodes **do**
  Receive initial state s
  for $t = 1$ to max-episode-length **do**
    Select action a_i from $\pi^b_{\theta_i}(s)$ $\forall i \in \mathcal{N}$
    Execute actions $a = \{a_i\}_{\forall i \in \mathcal{N}}$ and observe rewards $r = \{r_i\}_{\forall i \in \mathcal{N}}$ and new state s'
    Store the tuple (s, a, r, s') in replay buffer $\mathcal{D}$
    Set $s = s'$

    Sample a random K tuples $\{(s^k, a^k, r^k, s'^k)\}_{k \in \{1,...,K\}}$ from $\mathcal{D}$
    for agent $i = 1$ to n **do**
      Set $y^k_i = r^k_i + \gamma Q'_i(s'^k, a'_1, ..., a'_n)|_{a'_h = \mu'_h(s'^k)}$, for $k \in \{1, ..., K\}$
      Update state-action value function Q_i by minimizing:

$$\mathcal{L}(\omega_i) = \frac{1}{K} \sum_{k \in \{1,...,K\}} [(Q_i(s^k, a^k_1, ..., a^k_n; \omega_i) - y^k_i)^2]$$

    end for
    Set $a^k_i = \mu_{\theta_i}(s^k)$, for $k \in \{1, ..., K\}$ and $i \in \mathcal{N}$
    for agent $i = 1$ to n **do**
      for agent $j = 1$ to n **do**
        if $j = i$ **then** continue
        Set $\Delta a^k_j = \hat{\eta}_{\text{1st}} \frac{\partial}{\partial a^k_j} Q_j(s^k, a^k_1, ..., a^k_n)$ for $k \in \{1, ..., K\}$
      end for
      Update policy parameters θ_i via: $\nabla_{\theta_i} J^{\text{LA-OffPA2}}_i =$

$$\frac{1}{K} \sum_{k \in \{1,...,K\}} \nabla_{\theta_i} \mu_{\theta_i}(s^k) \frac{\partial}{\partial a^k_i} Q_i(s^k, a^k_1 + \perp \Delta a^k_1, ..., a^k_i, ..., a^k_n + \perp \Delta a^k_n)$$

    end for
    Update Q'_i and μ'_i $\forall i \in \mathcal{N}$
  end for
end for

where $\hat{\eta}_{\text{full}} = \eta C_1(s) + \eta^2 C_2(s) + O(\eta^3) \in \mathbb{R}$. Consequently, we have proved Lemma 3

If we now map the anticipated changes of policy parameters, with a prediction length $\eta' \in \mathbb{R}^+$, to the action space using full-order Taylor expansion, we have:

$$\mu_{\theta_2 + \Delta\theta'_2}(s) = \mu_{\theta_2}(s) + \hat{\eta'}_{\text{full}} \nabla_{a_2} Q_2(s, a_1, a_2), \tag{44}$$

where $\hat{\eta'}_{\text{full}} = \eta' C_1(s) + \eta'^2 C_2(s) + O(\eta'^3)$. In order to prove Theorem 2, we need to find the values of $\hat{\eta}_{\text{1st}}$ that yields:

$$\begin{aligned} \hat{\eta}_{\text{1st}} &= \hat{\eta'}_{\text{full}} \\ &= \eta' C_1(s) + \eta'^2 C_2(s) + O(\eta'^3), \end{aligned} \tag{45}$$

Algorithm 3: HLA-OffPA2 for a set of m common-interested agents ($\mathcal{M}$).

Initialize μ_{θ_i}, Q_i, μ'_i, and Q'_i $\forall i \in \mathcal{M}$, and set $\hat{\eta}_{1\text{st}}$
for episode = 1 to max-num-episodes **do**
  Receive initial state s
  for $t = 1$ to max-episode-length **do**
    Select action a_i from $\mu_{\theta_i}(s)$ and the exploration strategy $\forall i \in \mathcal{M}$
    Execute actions $a = \{a_i\}_{\forall i \in \mathcal{M}}$ and observe common reward r and new state s'
    Store the tuple (s, a, r, s') in replay buffer $\mathcal{D}$
    Set $s = s'$

    Sample a random K tuples $\{(s^k, a^k, r^k, s'^k)\}_{k\in\{1,...,K\}}$ from $\mathcal{D}$
    Set $y^k = r^k + \gamma Q'(s'^k, a'_1, ..., a'_m)|_{a'_h=\mu'_h(s'^k)}$, for $k \in \{1, ..., K\}$
    Update state-action value function Q by minimizing:

$$\mathcal{L} = \frac{1}{K} \sum_{k\in\{1,...,K\}} [(Q(s^k, a^k_1, ..., a^k_m) - y^k)^2]$$

    Assign the agents into m hierarchy levels using Eq. 23
    Rename the agent assigned to level i as the agent i, $\forall i \in \mathcal{M}$
    Set $a^k_i = \mu_{\theta_i}(s^k)$, for $\forall k \in \{1, ..., K\}$ and $\forall i \in \mathcal{M}$
    for agent $i = m$ to 1 **do**
      for agent $j = 1$ to i **do**
        Compute Δa^k_j, for $k \in \{1, ..., K\}$:
        if $j = 1$ & $i \neq m$ **then** $\Delta a^k_1 = \hat{\eta}_{1\text{st}} \frac{\partial}{\partial a^k_1} Q(s^k, a^k_1, ..., a^k_i, \bar{a}^k_{i+1}, ..., \bar{a}^k_m)$
        elif $j \neq 1$ & $i = m$ **then**
        $\Delta a^k_j = \hat{\eta}_{1\text{st}} \frac{\partial}{\partial a^k_j} Q(s^k, a^k_1 + \Delta a^k_1, ..., a^k_{j-1} + \Delta a^k_{j-1}, a^k_j, ..., a^k_m)$
        elif $j = 1$ & $i = m$ **then** $\Delta a^k_1 = \hat{\eta}_{1\text{st}} \frac{\partial}{\partial a^k_1} Q(s^k, a^k_1, ..., a^k_m)$
        else $\Delta a^k_j = \hat{\eta}_{1\text{st}} \frac{\partial}{\partial a^k_j} Q(s^k, a^k_1 + \Delta a^k_1, ..., a^k_{j-1} + \Delta a^k_{j-1}, a^k_j, \bar{a}^k_{j+1}, ...\bar{a}^k_m)$
      end for
      Update policy parameters θ_i via:

$$\nabla_{\theta_i} J^{\text{HR}}_i \approx \frac{1}{K} \sum_{k\in\{1,...,K\}} \nabla_{\theta_i} \mu_{\theta_1}(s^k) \Delta a^k_i$$

      Set $\bar{a}^k_i = \text{detach}(a^k_i + \Delta a^k_i)$, for $k \in \{1, ..., K\}$
    end for
    Update Q'_i and μ'_i $\forall i \in \mathcal{M}$
  end for
end for

and at the same time $\eta' \in \mathbb{R}^+$. By neglecting $O(\eta'^3)$ and given that $\hat{\eta}_{1\text{st}} \in \mathbb{R}^+$, there are two cases to be considered:

- if $C_2(s)$ is non-negative, then for any value of $\hat{\eta}_{1\text{st}} \in \mathbb{R}^+$, there exists $\eta \in \mathbb{R}^+$.

- if $C_2(s)$ is negative, then for $\hat{\eta}_{\text{1st}} < \frac{C_1(s)^2}{4|C_2(s)|}$, there exists $\eta \in \mathbb{R}^+$.

Therefore, for sufficiently small $\hat{\eta}_{\text{1st}}$, i.e., $\hat{\eta}_{\text{1st}} < \frac{C_1(s)^2}{4|C_2(s)|}$, there always exists $\eta \in \mathbb{R}^+$, and consequently, Theorem 2 is proved.

Appendix B. Implementations details

In this section, we describe the implementations of methods in detail. In order to have fair comparisons between the methods, we have used policies and value functions with the same neural network architecture in all methods. Algorithms 1, 2, and 3 illustrates the optimization frameworks for LOLA-OffPA2, LA-OffPA2, and HLA-OffPA2, respectively. The source code of our OffPA2 framework is available at a GitHub repository[1].

A note on partial observability. So far, we have formulated the MARL setup as an MG, where it is assumed that the agents have access to the state space. However, in many games, the agents only receive a private state observation of the current state. In this case, the MARL setup can be formulated as a Partially Observable Markov Game (PO-MG) (Littman, 1994). A PO-MG is a tuple $(\mathcal{N}, \mathcal{S}, \{\mathcal{A}_i\}_{i\in\mathcal{N}}, \{\mathcal{O}_i\}_{i\in\mathcal{N}}, \{\mathcal{R}_i\}_{i\in\mathcal{N}}, \mathcal{T}, \{\Omega_i\}_{i\in\mathcal{N}}, \rho, \gamma)$, where $\mathcal{O}_i$ is the set of sate observations for agent $i \in \mathcal{N}$. Each agent i chooses its action $a_i \in \mathcal{A}_i$ through the policy $\mu_{\theta_i} : \mathcal{O}_i \to \mathcal{A}_i$ parameterized by θ_i conditioning on the given state observation $o_i \in \mathcal{O}_i$. After the transition to a new state, each agent i receives a private state observation through its observation function $\Omega_i : \mathcal{S} \to \mathcal{O}_i$. In this case, the centralized state-action value function for each agent i is defined as $Q_i(o_1, ..., o_n, a_1, ..., a_n) = \mathbb{E}[G_i^t(\tau | s^t = s, o_i = \Omega_i(s) \;\&\; a_i^t = a_i \;\; \forall i \in \mathcal{N})]$. Therefore, the proposed OffPA2 framework can be modified accordingly.

B.1 Iterated rotational game and iterated prisoner's dilemma

We employed Multi-Layer Perceptron (MLP) networks with two hidden layers of dimension 64 for policies and value functions. In order to make the state-action value functions any-order differentiable, we used SiLU nonlinear function (Elfwing et al., 2018) in between the hidden layers. For IRG, we used the Sigmoid function in the policies to output 1-D continues action, and for IPD, we used the Gumble-softmax function (Jang et al., 2017) in the policies to output two discrete actions. The algorithms are trained for 900 (in IRG) and 50 (in IPD) episodes by running Adam optimizer (Kingma and Ba, 2015) with a fixed learning rate of 0.01. The (projected) prediction lengths in OffPA2 and DiCE frameworks are tuned and set to 0.8 and 0.3, respectively. All experiments are repeated five times, and the results are reported in terms of mean and standard deviation.

B.2 Exit-Room game

Both policy and value networks consist of two parts: encoder and decoder. The encoders are CNN networks with three convolutional layers ($12 \times 90 \times 90 \to 32 \times 21 \times 21 \to 64 \times 9 \times 9 \to 64 \times 7 \times 7$) and two fully connected layers ($3136 \to 512 \to 128$), with SiLU nonlinear functions (Elfwing et al., 2018) in between. The decoders are MLP networks with

1. `https://github.com/tue-mps/OffPA2`

	$\hat{\eta}_{1st}$ in Particle Environment			$\hat{\eta}_{1st}$ in Mujoco Environment		
	Cooperative Navigation	Physical Deception	Predator-Prey	Half-Cheetah	Walker	Reacher
HLA-OffPA2	0.003	0.01	0.04	0.004	0.004	0.007

Table 10: The optimized projected prediction lengths for OffPA2-based methods.

two hidden layers of dimension 64 for policies and value functions. We used the Gumble-softmax function in the policies (Jang et al., 2017) to output the discrete actions. The algorithms are trained for 450 (in level one) and 4500 (in levels two and three) episodes by running Adam optimizer (Kingma and Ba, 2015) with a fixed learning rate of 0.01. The (projected) prediction lengths in OffPA2 and DiCE frameworks are tuned and set to 1 and 0.4, respectively. All experiments are repeated five times, and the results are reported in terms of mean and standard deviation.

B.3 Particle-coordination game

We employed MLP networks with two hidden layers of dimension 64 for policies and state-action value functions with SiLU nonlinear functions (Elfwing et al., 2018) in between. We used the Gumble-softmax function (Jang et al., 2017) in the policies to output the discrete actions. The algorithms are trained for 100k episodes by running Adam optimizer (Kingma and Ba, 2015) with a fixed learning rate of 0.01. We set the projected prediction length to 0.1 for HLA-OffPA2 agents. All experiments are repeated five times, and the results are reported in terms of mean and standard deviation.

B.4 Standard multi-agent games

As before, we used policies and state-action value functions with the same neural network architecture in all methods. We employed MLP networks with two hidden layers (of dimension 64 for the Particle environment and 256 for the Mujoco environment) for policies and state-action value functions with SiLU nonlinear functions (Elfwing et al., 2018). In the Particle environment, We used the Gumble-softmax function (Jang et al., 2017) in the policies to output the discrete actions and trained the algorithms for 100k episodes by running Adam optimizer (Kingma and Ba, 2015) with a fixed learning rate of 0.01. In the Mujoco environment, we used the Tanh function in the policies to output the continuous actions and train the algorithms for 10k episodes by running Adam optimizer (Kingma and Ba, 2015) with a fixed learning rate of 0.001. The projected prediction lengths for HLA-OffPA2 agents are optimized between $0.001 - 0.1$ in all games. The optimized projected prediction lengths are reported in Table 10.

References

Stefano V Albrecht and Peter Stone. Autonomous agents modelling other agents: A comprehensive survey and open problems. *Artificial Intelligence*, 258:66–95, 2018.

David Balduzzi, Sebastien Racaniere, James Martens, Jakob Foerster, Karl Tuyls, and Thore Graepel. The mechanics of n-player differentiable games. In *International Confer-*

ence on Machine Learning, pages 354–363. PMLR, 2018.

Ariyan Bighashdel, Daan de Geus, Pavol Jancura, and Gijs Dubbelman. Coordinating Fully-Cooperative Agents Using Hierarchical Learning Anticipation. *arXiv preprint arXiv:2303.08307*, 2023.

Xuxi Chen, Nelson Vadori, Tianlong Chen, and Zhangyang Wang. Learning to optimize differentiable games. In *International Conference on Machine Learning*, pages 5036–5051. PMLR, 2023.

Thomas Degris, Martha White, and Richard Sutton. Off-Policy Actor-Critic. In *International Conference on Machine Learning*, 2012.

Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, 2018.

Jakob Foerster, Richard Y Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. Learning with Opponent-Learning Awareness. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 122–130, 2018a.

Jakob Foerster, Gregory Farquhar, Maruan Al-Shedivat, Tim Rocktäschel, Eric Xing, and Shimon Whiteson. Dice: The infinitely differentiable monte carlo estimator. In *International Conference on Machine Learning*, pages 1529–1538. PMLR, 2018b.

Jakob N Foerster, Yannis M Assael, Nando De Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. *arXiv preprint arXiv:1605.06676*, 2016.

Adam S Goodie, Prashant Doshi, and Diana L Young. Levels of theory-of-mind reasoning in competitive games. *Journal of Behavioral Decision Making*, 25(1):95–108, 2012.

He He, Jordan Boyd-Graber, Kevin Kwok, and Hal Daumé III. Opponent modeling in deep reinforcement learning. In *International conference on machine learning*, pages 1804–1813. PMLR, 2016.

Zhang-Wei Hong, Shih-Yang Su, Tzu-Yun Shann, Yi-Hsiang Chang, and Chun-Yi Lee. A Deep Policy Inference Q-Network for Multi-Agent Systems. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1388–1396, 2018.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical Reparametrization with Gumble-Softmax. In *International Conference on Learning Representations (ICLR 2017)*. OpenReview. net, 2017.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Sachin Konan, Esmaeil Seraj, and Matthew Gombolay. Iterated Reasoning with Mutual Information in Cooperative and Byzantine Decentralized Teaming. *arXiv preprint arXiv:2201.08484*, 2022.

Alistair Letcher, Jakob Foerster, David Balduzzi, Tim Rocktäschel, and Shimon Whiteson. Stable Opponent Shaping in Differentiable Games. In *International Conference on Learning Representations*, 2019.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *ICLR (Poster)*, 2016.

Raymond Lister and James V Stone. An empirical study of the time complexity of various error functions with conjugate gradient backpropagation. In *Proceedings of ICNN'95-International Conference on Neural Networks*, volume 1, pages 237–241. IEEE, 1995.

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

Daxin Liu and Gerhard Lakemeyer. Reasoning about Beliefs and Meta-Beliefs by Regression in an Expressive Probabilistic Action Logic. In *international joint conference on artificial intelligence*. IJCAI, 2021.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *NIPS*, 2017.

Christopher Lu, Timon Willi, Christian A Schroeder De Witt, and Jakob Foerster. Model-Free Opponent Shaping. In *International Conference on Machine Learning*, pages 14398–14411. PMLR, 2022.

Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. The numerics of gans. *Advances in neural information processing systems*, 30, 2017.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, and Others. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Bei Peng, Tabish Rashid, Christian Schroeder de Witt, Pierre-Alexandre Kamienny, Philip Torr, Wendelin Boehmer, and Shimon Whiteson. FACMAC: Factored Multi-Agent Centralised Policy Gradients. *NeurIPS*, 2021.

Giorgia Ramponi and Marcello Restelli. Newton optimization on helmholtz decomposition for continuous games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11325–11333, 2021.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pages 387–395. PMLR, 2014.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Eugene Vinitsky, Natasha Jaques, Joel Leibo, Antonio Castenada, and Edward Hughes. An open source implementation of sequential social dilemma games, 2019. GitHub repository.

Ying Wen, Yaodong Yang, Rui Luo, Jun Wang, and Wei Pan. Probabilistic recursive reasoning for multi-agent reinforcement learning. In *7th International Conference on Learning Representations, ICLR 2019*, 2019.

Ying Wen, Yaodong Yang, and Jun Wang. Modelling Bounded Rationality in Multi-Agent Interactions by Generalized Recursive Reasoning. In *IJCAI*, 2020.

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

Chongjie Zhang and Victor Lesser. Multi-agent learning with policy prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 24, 2010.