PLAN BETTER AMID CONSERVATISM: OFFLINE MULTI-AGENT REINFORCEMENT LEARNING WITH ACTOR RECTIFICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

The idea of conservatism has led to significant progress in offline reinforcement learning (RL) where an agent learns from pre-collected datasets. However, it is still an open question to resolve offline RL in the more practical multi-agent setting as many real-world scenarios involve interaction among multiple agents. Given the recent success of transferring online RL algorithms to the multi-agent setting, one may expect that offline RL algorithms will also transfer to multi-agent settings directly. Surprisingly, when conservatism-based algorithms are applied to the multi-agent setting, the performance degrades significantly with an increasing number of agents. Towards mitigating the degradation, we identify that a key issue that the landscape of the value function can be non-concave and policy gradient improvements are prone to local optima. Multiple agents exacerbate the problem since the suboptimal policy by any agent could lead to uncoordinated global failure. Following this intuition, we propose a simple yet effective method, Offline Multi-Agent RL with Actor Rectification (OMAR), to tackle this critical challenge via an effective combination of first-order policy gradient and zeroth-order optimization methods for the actor to better optimize the conservative value function. Despite the simplicity, OMAR significantly outperforms strong baselines with state-of-the-art performance in multi-agent continuous control benchmarks.

1 INTRODUCTION

Offline reinforcement learning (RL) has shown great potential in advancing the deployment of RL in real-world tasks where interaction with the environment is prohibitive, costly, or risky (Thomas, 2015). Since an agent has to learn from a given pre-collected dataset in offline RL, it becomes challenging for regular online RL algorithms such as DDPG (Lillicrap et al., 2016) and TD3 (Fujimoto et al., 2018) due to extrapolation error (Lee et al., 2021).

There has been recent progress in tackling the problem based on conservatism. Behavior regularization (Wu et al., 2019; Kumar et al., 2019), *e.g.*, TD3 with Behavior Cloning (TD3+BC) (Fujimoto & Gu, 2021), compels the learning policy to stay close to the manifold of the datasets. Yet, its performance highly depends on the quality of the dataset. Another line of research investigates incorporating conservatism into the value function by critic regularization (Nachum et al., 2019; Kostrikov et al., 2021), *e.g.*, Conservative Q-Learning (Kumar et al., 2020), which usually learns a conservative estimate of the value function to directly address the extrapolation error.

However, many practical scenarios involve multiple agents, *e.g.*, multi-robot control (Amato, 2018), autonomous driving (Pomerleau, 1989; Sadigh et al., 2016). Therefore, offline multi-agent reinforcement learning (MARL) (Yang et al., 2021; Jiang & Lu, 2021) is crucial for solving real-world tasks. Observing recent success of Independent PPO (de Witt et al., 2020) and Multi-Agent PPO (Yu et al., 2021), both of which are based on the PPO (Schulman et al., 2017) algorithm, we find that online RL algorithms can be transferred to multi-agent scenarios through either decentralized training or a centralized value function without bells and whistles. Hence, we naturally expect that offline RL algorithms would also transfer easily when applied to multi-agent tasks.

Surprisingly, we observe that the performance of the state-of-the-art conservatism-based CQL (Kumar et al., 2020) algorithm in offline RL degrades dramatically with an increasing number of agents

as shown in Figure 1(c) in our experiments. Towards mitigating the degradation, we identify a critical issue in CQL: solely regularizing the critic is insufficient for multiple agents to learn good policies for coordination in the offline setting. The primary cause is that first-order policy gradient methods are prone to local optima (Nachum et al., 2016; Ge et al., 2017; Safran & Shamir, 2017), saddle points (Vlatakis-Gkaragkounis et al., 2019; Sun et al., 2020), or noisy gradient estimates (Such et al., 2017). As a result, this can lead to uncoordinated suboptimal learning behavior because the actor cannot leverage the global information in the critic well. The issue is exacerbated more in the multi-agent settings due to the exponentially-sized joint action space (Yang et al., 2021) as well as the nature of the setting that requires *each* of the agent to learn a good policy for a successful joint policy. For example, in a basketball game, where there are two competing teams each consisting of five players. When one of the players passes the ball among them, it is important for all teammates to perform their duties well in their roles to win the game. As a result, if one of the agents in the team fails to learn a good policy, it can fail to cooperate with other agents for coordinated behaviors and lose the ball.

In this paper, we propose a surprisingly simple yet effective method for offline multi-agent continuous control, Offline MARL with Actor Rectification (OMAR), to better leverage the conservative value function via an effective combination of first-order policy gradient and zeroth-order optimization methods. Towards this goal, we add a regularizer to the actor loss, which encourages the actor to mimic actions from the zeroth-order optimizer that maximizes Q-values so that we can combine the best of both first-order policy gradient and zeroth-order optimization. The sampling mechanism is motivated by evolution strategies (Such et al., 2017; Conti et al., 2017; Mania et al., 2018), which recently emerged as another paradigm for solving sequential decision making tasks (Salimans et al., 2017). Specifically, the zeroth-order optimization part maintains an iteratively updated and refined Gaussian distribution to find better actions based on Q-values. Then, we rectify the policy towards this action to better leverage the conservative value function. We conduct extensive experiments in standard continuous control multi-agent particle environments and the complex multi-agent locomotion task to demonstrate its effectiveness. On all the benchmark tasks, OMAR outperforms the multi-agent version of offline RL algorithms including CQL (Kumar et al., 2020) and TD3+BC (Fujimoto & Gu, 2021), as well as a recent offline MARL algorithm MA-ICQ (Yang et al., 2021), and achieves the state-of-the-art performance.

The main contribution of this work can be summarized as follows. We propose the OMAR algorithm that effectively leverages both first-order and zero-order optimization for solving offline MARL tasks. In addition, we theoretically prove that OMAR leads to safe policy improvement. Finally, extensive experimental results demonstrate the effectiveness of OMAR, which significantly outperforms strong baseline methods and achieves state-of-the-art performance in datasets with different qualities in both decentralized and centralized learning paradigms.

2 BACKGROUND

We consider the framework of partially observable Markov games (POMG) (Littman, 1994; Hu et al., 1998), which extends Markov decision processes to the multi-agent setting. A POMG with N agents is defined by a set of global states $\mathcal{S}$, a set of actions $\mathcal{A}_1, \ldots, \mathcal{A}_N$, and a set of observations $\mathcal{O}_1, \ldots, \mathcal{O}_N$ for each agent. At each timestep, each agent i receive an observation o_i and chooses an action based on its policy π_i. The environment transits to the next state according to the state transition function $\mathcal{P} : \mathcal{S} \times \mathcal{A}_1 \times \ldots \times \mathcal{A}_N \times \mathcal{S} \to [0, 1]$. Each agent receives a reward based on the reward function $r_i : \mathcal{S} \times \mathcal{A}_1 \ldots \times \mathcal{A}_N \to \mathbb{R}$ and a private observation $o_i : \mathcal{S} \to \mathcal{O}_i$. The initial state distribution is defined by $\rho : \mathcal{S} \to [0, 1]$. The goal is to find a set of optimal policies $\boldsymbol{\pi} = \{\pi_1, \ldots, \pi_N\}$, where each agent aims to maximize its own discounted return $\sum_{t=0}^{\infty} \gamma^t r_i^t$ with γ denoting the discount factor. In the offline setting, agents learn from a fixed dataset $\mathcal{D}$ generated from the behavior policy $\boldsymbol{\pi}_\beta$ without interaction with the environments.

2.1 MULTI-AGENT ACTOR CRITIC

Centralized critic. Lowe et al. (2017) propose Multi-Agent Deep Deterministic Policy Gradients (MADDPG) under the centralized training with decentralized execution (CTDE) paradigm by extending the DDPG algorithm (Lillicrap et al., 2016) to the multi-agent setting. In CTDE, agents are trained in a centralized way where they can access to extra global information dur-

ing training while they need to learn decentralized policies in order to act based only on local observations during execution. In MADDPG, for an agent i, the centralized critic Q_i is parameterized by θ_i. It takes the global state action joint action as inputs, and aims to minimize the temporal difference error defined by $\mathcal{L}(\theta_i) = \mathbb{E}_{\mathcal{D}}\left[\left(Q_i\left(s, a_1, \ldots, a_n\right) - y_i\right)^2\right]$, where $y_i = r_i + \gamma \bar{Q}_i(s', a'_1, \cdots, a'_n)|_{a'_j = \bar{\pi}_j(o'_j)}$ and $\bar{Q}_i$ and $\bar{\pi}_i$ denote target networks. To reduce the overestimation problem in MADDPG, MATD3 (Ackermann et al., 2019) estimates the target value using double estimators based on TD3 (Fujimoto et al., 2018), where $y_i = r_i + \gamma \min_{k=1,2} \bar{Q}_i^k(s', a'_1, \cdots, a'_n)|_{a'_j = \bar{\pi}_j(o'_j)}$. Agents learn decentralized policies π_i parameterized by ϕ_i, which take only local observations as inputs, and are trained by multi-agent policy gradients according to $\nabla_{\phi_i} J(\pi_i) = \mathbb{E}_{\mathcal{D}}\left[\nabla_{\phi_i} \pi_i(a_i|o_i) \nabla_{a_i} Q_i\left(s, a_1, \ldots, a_n\right)|_{a_i = \pi_i(o_i)}\right]$, where a_i is predicted from its policy while a_{-i} are sampled from the replay buffer.

Decentralized critic. Although using centralized critics is widely-adopted in multi-agent actor-critic methods, it introduces scalability issues due to the exponentially sized joint action space w.r.t. the number of agents (Iqbal & Sha, 2019). On the other hand, independent learning approaches train decentralized critics that take only the local observation and action as inputs. It is shown in de Witt et al. (2020); Lyu et al. (2021) that decentralized value functions can result in more robust performance and be beneficial in practice compared with centralized critic approaches. de Witt et al. (2020) propose Independent Proximal Policy Optimization (IPPO) based on PPO (Schulman et al., 2017), and show that it can match or even outperform CTDE approaches in the challenging discrete control benchmark tasks (Samvelyan et al., 2019). We can also obtain the Independent TD3 (ITD3) algorithm based on decentralized critics, which is trained to minimize the temporal difference error defined by $\mathcal{L}(\theta_i) = \mathbb{E}_{\mathcal{D}}\left[\left(Q_i\left(o_i, a_i\right) - y_i\right)^2\right]$, where $y_i = r_i + \gamma \min_{k=1,2} \bar{Q}_i^k(o'_i, \bar{\pi}_i(o'_i))$.

2.2 Conservative Q-Learning

Conservative Q-Learning (CQL) (Kumar et al., 2020) adds a regularizer to the critic loss to address the extrapolation error and learns lower-bounded Q-values. It penalizes Q-values of state-action pairs sampled from a uniform distribution or a policy while encouraging Q-values for state-action pairs in the dataset to be large. Specifically, when built upon decentralized critic methods in MARL, the critic loss is defined as in Eq. (1), where α denotes the regularization coefficient and $\hat{\pi}_{\beta_i}$ is the empirical behavior policy of agent i.

$$\mathbb{E}_{\mathcal{D}_i}\left[(Q_i(o_i, a_i) - y_i)^2\right] + \alpha \mathbb{E}_{\mathcal{D}_i}\left[\log \sum_{a_i} \exp(Q_i(o_i, a_i)) - \mathbb{E}_{a_i \sim \hat{\pi}_{\beta_i}(a_i|o_i)}[Q_i(o_i, a_i)]\right] \tag{1}$$

3 Proposed Method

In this section, we first provide a motivating example where previous methods, such as CQL (Kumar et al., 2020) and TD3+BC (Fujimoto & Gu, 2021) can be inefficient in the face of the multi-agent setting. Then, we propose a method called Offline Multi-Agent Reinforcement Learning with Actor Rectification (OMAR), where we effectively combine first-order policy gradients and zeroth-order optimization methods for the actor to better optimize the conservative value function.

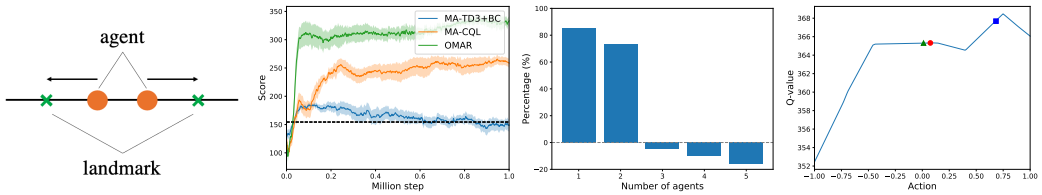
3.1 The Motivating Example

We design a Spread environment as shown in Figure 1(a) which involves n agents and n landmarks ($n \geq 1$) with 1-dimensional action space to demonstrate the problem and reveal interesting findings. For the multi-agent setting in the Spread task, n agents need to learn how to cooperate to cover all landmarks and avoid colliding with each other or arriving at the same landmark by coordinating their actions. The experimental setup is the same as in Section 4.1.1.

Figure 1(b) demonstrates the performance of the multi-agent version of TD3+BC (Fujimoto & Gu, 2021), CQL (Kumar et al., 2020), and OMAR based on ITD3 in the medium-replay dataset from the two-agent Spread environment. As MA-TD3+BC is based on behavior regularization that compels the learned policy to stay close to the behavior policy, its performance largely depends on the quality of the dataset. Moreover, it can be detrimental to regularize policies to be close to the dataset in

multi-agent settings due to decentralized training and the resulting partial observations. MA-CQL outperforms MA-TD3+BC, which pushes down Q-values of state-action pairs that are sampled from a random or the current policy while pushing up Q-values for state-action pairs in the dataset.

Figure 1(c) demonstrates the performance improvement percentage of MA-CQL over the behavior policy with an increasing number of agents ranging from one to five. From Figure 1(c), we observe that its performance degrades dramatically as there are more agents.



(a) (b) (c) (d)

Figure 1: Analysis of MA-TD3+BC, MA-CQL, and OMAR in the medium-replay dataset from Spread. (a) Spread. (b) Performance. (c) Performance improvement percentage of MA-CQL over the behavior policy with varying number of agents. (d) Visualization of the Q-function landscape. The red circle represents the predicted action from the agent using MA-CQL. The green triangle and blue square represent the predicted action from the updated policy of MA-CQL and OMAR.

Towards mitigating the performance degradation, we identify a key issue in MA-CQL that solely regularizing the critic is insufficient for multiple agents to learn good policies for coordination. In Figure 1(d), we visualize the Q-function landscape of MA-CQL during training for an agent in a timestep, with the red circle corresponding to the predicted action from the actor. The green triangle represents the action predicted from the actor after the training step, where the policy gets stuck in a bad local optimum. The first-order policy gradient method is prone to local optima (Dauphin et al., 2014; Ahmed et al., 2019), where the agent can fail to globally leverage the conservative value function well and thus leading to suboptimal, uncoordinated learning behavior. Note that the problem is exacerbated more in the offline multi-agent setting due to the exponentially sized joint action space w.r.t. the number of agents (Yang et al., 2021). In addition, it usually requires *each* of the agent to learn a good policy for coordination to solve the task, and the suboptimal policy by any agent could result in uncoordinated global failure.

Tables 1 and 2 show the performance of MA-CQL by increasing the learning rate or the number of updates for the actor. The results illustrate that, to solve this challenging problem, we need a better solution than blindly tuning hyperparameters. In the next section, we introduce how we tackle this problem by combining zeroth-order optimization with current RL algorithms.

Table 1: Performance of MA-CQL with larger learning rate for the actor.

Learning rate	$1e-2$	$5e-2$	$1e-1$
Performance	267.9 ±19.0	202.0 ±38.9	100.1 ±36.4

Table 2: Performance of MA-CQL with larger number of updates for the actor.

# Updates	1	5	20
Performance	267.9 ±19.0	278.6 ±14.8	263.7 ±23.1

3.2 Offline Multi-Agent Reinforcement Learning with Actor Rectification

Our key identification as above is that policy gradient improvements are prone to local optima given a bad value function landscape. It is important to note that this presents a particularly critical challenge in the multi-agent setting since it is sensitive to suboptimal actions. Zeroth-order optimization methods, *e.g.*, evolution strategies (Rubinstein & Kroese, 2013; Such et al., 2017; Conti et al., 2017; Salimans et al., 2017; Mania et al., 2018), offer an alternative for policy optimization and are also robust to local optima (Rubinstein & Kroese, 2013).

We propose Offline Multi-Agent Reinforcement Learning with Actor Rectification (OMAR) which incorporates sampled actions based on Q-values to rectify the actor so that it can escape from bad

local optima. For simplicity of presentation, we demonstrate our method based on the decentralized training paradigm introduced in Section 2.1. Note that it can also be applied to centralized critics, as shown in Section 4.1.4. Specifically, we add a regularizer to the policy objective:

$$\mathbb{E}_{\mathcal{D}_i}\left[(1-\tau)Q_i(o_i,\pi_i(o_i)) - \tau\left(\pi_i(o_i)-\hat{a}_i\right)^2\right] \tag{2}$$

where $\hat{a}_i$ is the action provided by the zeroth-order optimizer and $\tau \in [0,1]$ denotes the regularization coefficient. Note that TD3+BC (Fujimoto & Gu, 2021) uses the seen action in the dataset for $\hat{a}_i$. The distinction between *optimized* and *seen* actions enables OMAR to perform well even if the dataset quality is from mediocre to low.

We borrow intuition for sampling actions from recent evolution strategy (ES) algorithms, which show a welcoming avenue towards using zeroth-order method for policy optimization. For example, the cross-entropy method (CEM) (Rubinstein & Kroese, 2013), a popular ES algorithm, has shown great potential in RL (Lim et al., 2018), especially by sampling in the parameter space of the actor (Pourchot & Sigaud, 2019). However, CEM does not scale to tasks with high-dimensional space well (Nagabandi et al., 2020). We therefore propose to sample actions in a softer way motivated by Williams et al. (2015); Lowrey et al. (2018). Specifically, we sample actions according to an iteratively refined Gaussian distribution $\mathcal{N}(\mu_i,\sigma_i)$. At each iteration j, we draw K candidate actions by $a_i^j \sim \mathcal{N}(\mu_i^j,\sigma_i^j)$ and evaluate their Q-values. The mean and standard deviation of the sampling distribution is updated and refined by Eq. (3), which produces a softer update and leverages more samples in the update (Nagabandi et al., 2020). The OMAR algorithm is shown in Algorithm 1.

$$\mu_i^{j+1} = \frac{\sum_{k=1}^{K}\exp(\beta Q_i^k)a_i^k}{\sum_{m=1}^{K}\exp(\beta Q_i^m)}, \quad \sigma_i^{j+1} = \sqrt{\sum_{k=1}^{K}\left(a_i^k-\mu_i^j\right)^2}. \tag{3}$$

Besides the algorithmic design, we also prove that OMAR gives a safe policy improvement guarantee. Let $J(\pi_i)$ denote the discounted return of a policy π_i in the empirical MDP $\hat{M}_i$ which is induced by transitions in the dataset $\mathcal{D}_i$, i.e., $\hat{M}_i = \{(o_i,a_i,r_i,o_i') \in \mathcal{D}_i\}$. In Theorem 1, we give a lower bound on the difference between the policy performance of OMAR over the empirical behavior policy $\hat{\pi}_{\beta_i}$ in the empirical MDP $\hat{M}_i$. The proof can be found in Appendix A.

Theorem 1. *Let π_i^* be the policy obtained by optimizing Eq. (2). Then, we have that $J(\pi_i^*) - J(\hat{\pi}_{\beta_i}) \geq \frac{\alpha}{1-\gamma}\mathbb{E}_{o_i\sim d^{\pi_i^*}(o_i)}\left[D(\pi_i^*,\hat{\pi}_{\beta_i})(o_i)\right] + \frac{\tau}{1-\tau}\mathbb{E}_{o_i\sim d^{\pi_i^*}(o_i)}\left[(\pi_i^*(o_i)-\hat{a}_i)^2\right] - \frac{\tau}{1-\tau}\mathbb{E}_{o_i\sim d^{\hat{\pi}_{\beta_i}}(o_i),a_i\sim\hat{\pi}_{\beta_i}}\left[(a_i-\hat{a}_i)^2\right]$, where $D(\pi_i,\hat{\pi}_{\beta_i})(o_i) = \frac{1}{\hat{\pi}_{\beta_i}(\pi_i(o_i)|o_i)} - 1$, and $d^{\pi_i}(o_i)$ is the marginal discounted distribution of observations of policy π_i.*

As shown in Theorem 1, the difference between the second and third terms on the right-hand side is the difference between two expected distances. The former corresponds to the gap between the optimal action and the action from our zeroth-order optimizer, while the latter corresponds to the gap between the action from the behavior policy and the optimized action. Since both terms can be bounded, we find that OMAR gives a safe policy improvement guarantee over $\hat{\pi}_{\beta_i}$.

Discussion of the effect of OMAR in the Spread environment. We now investigate whether OMAR can address the identified problem and analyze its effect in the Spread environment introduced in Section 3.1. In Figure 1(d), the blue square corresponds to the action from the updated actor using OMAR according to Eq. (2). In contrast to the policy update in MA-CQL, OMAR can better leverage the global information in the critic and help the actor to escape from the bad local optima. Figure 1(b) further validates that OMAR significantly improves MA-CQL in terms of both performance and efficiency. Figure 2 shows the performance improvement percentage of OMAR over MA-CQL with varying number of agents, where OMAR always outperforms MA-CQL. We also notice that the performance improvement of OMAR over MA-CQL is much more significant in the multi-agent setting in the Spread task than the single-agent setting, which echoes with what is discussed above that the problem becomes more critical in scenarios with more agents that requires each of the agents to learn a good policy to cooperate for solving the task.

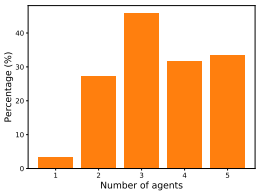


Figure 2: Performance improvement percentage of OMAR over MA-CQL with varying number of agents.

Algorithm 1 Offline Multi-Agent Reinforcement Learning with Actor Rectification (OMAR).

Initialize Q-networks Q_i^1, Q_i^2 and policy networks π_i with random parameters θ_1^i, θ_2^i, ϕ_i, and target networks with $\bar{\theta}_i^1 \leftarrow \theta_i^1$, $\bar{\theta}_i^2 \leftarrow \theta_i^2$, and $\bar{\phi}_i \leftarrow \phi_i$ for each agent $i \in [1, N]$
for training step $t = 1$ to T **do**
  for agent $i = 1$ to N **do**
    Sample a random minibatch of S samples (o_i, a_i, r_i, o_i') from $\mathcal{B}$
    Set $y = r_i + \gamma \min\left(\bar{Q}_i^1(o_i', \pi_i(o_i' + \epsilon)), \bar{Q}_i^2(o_i', \pi_i(o_i' + \epsilon))\right)$
    Update critics θ_i to minimize Eq. (1)
    Initialize $\mathcal{N}(\mu_i, \sigma_i)$
    for iteration $j = 1$ to J **do**
      Draw a population with K individuals $\hat{\mathcal{A}}_i = \{\hat{a}_i^k \sim \mathcal{N}(\mu_i, \sigma_i)\}_{k=1}^K$
      Estimate Q-values for K individuals in the population $\{Q_i^1(o_i, \hat{a}_i^k)\}_{k=1}^K$
      Update μ_i and σ_i of the distribution according to Eq. (3)
    Obtain the picked candidate action $\hat{a}_i = \arg\max_{\hat{a}_i \in \hat{\mathcal{A}}_i \cup \pi_i(o_i)} Q_i^1(o_i, \hat{a}_i)$
    Update the actor $\phi_i \leftarrow \max_{\phi_i} \frac{1}{S} \sum (1-\tau) Q_i^1(o_i, \pi_i(o_i)) - \tau \left(\pi_i(o_i) - \hat{a}_i\right)^2$
    Update target networks: $\bar{\theta}_i^j \leftarrow \rho \theta_i^j + (1-\rho)\bar{\theta}_i^j$ and $\bar{\phi}_i \leftarrow \rho \phi_i + (1-\rho)\bar{\phi}_i$

4 EXPERIMENTS

In this section, we conduct a series of experiments to study the following key questions: i) How does OMAR compare against state-of-the-art offline RL and offline MARL methods? ii) What is the effect of critical hyperparameters and the sampling scheme? iii) Does the method help in both centralized training and decentralized training paradigms? iv) Can OMAR scale to the more complex continuous multi-agent locomotion tasks?

4.1 MULTI-AGENT PARTICLE ENVIRONMENTS

4.1.1 EXPERIMENTAL SETUP

We first conduct a series of experiments in the widely-adopted multi-agent particle tasks (Lowe et al., 2017) as shown in Figure 5 in Appendix B.1. The cooperative navigation task includes 3 agents and 3 landmarks, where agents are rewarded based on the distance to the landmarks and penalized for colliding with each other. Thus, it is important for agents to cooperate to cover all landmarks without collision. In predator-prey, 3 predators aim to catch the prey. The predators need to cooperate to surround and catch the prey as the predators are slower than the prey. The world task involves 4 slower cooperating agents that aim to catch 2 faster adversaries, where adversaries desire to eat foods while avoiding being captured.

We construct a variety of datasets according to behavior policies with different qualities based on adding noises to the MATD3 algorithm to increase diversity following previous work (Fu et al., 2020). The random dataset is generated by rolling out a randomly initialized policy for 1 million (M) steps. We obtain the medium-replay dataset by recording all samples in the experience replay buffer during the training process until the policy reached the medium level of performance. The medium dataset consists of 1M samples by unrolling a partially-pretrained policy in the online setting whose performance reaches a medium level of the performance. The expert dataset is constructed by 1M expert demonstrations from an online fully-trained policy.

We compare OMAR against state-of-the-art offline RL algorithms including CQL (Kumar et al., 2020) and TD3+BC (Fujimoto & Gu, 2021). We also compare with a recent offline MARL algorithm MA-ICQ (Yang et al., 2021). We build all methods on independent TD3 based on decentralized critics following de Witt et al. (2020), while we also consider centralized critics based on MATD3 following Yu et al. (2021) in Section 4.1.4. All baselines are implemented based on the open-source code[1]. Each algorithm is run for five random seeds, and we report the mean performance with standard deviation. A detailed description of the construction of the datasets and hyperparameters can be found in Appendix B.1.

[1] https://github.com/shariqiqbal2810/maddpg-pytorch

4.1.2 PERFORMANCE COMPARISON

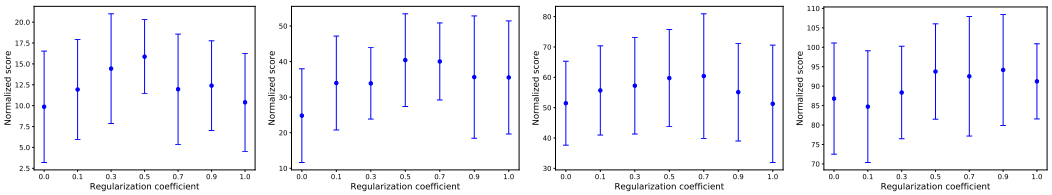
Table 3 summarizes the average normalized scores in different datasets in multi-agent particle environments, where the learning curves are shown in Appendix B.2. The normalized score is computed as $100 \times (S - S_{\text{random}})/(S_{\text{expert}} - S_{\text{random}})$ following Fu et al. (2020) As shown, the performance of MA-TD3+BC highly depends on the quality of the dataset. As the MA-ICQ method is based on only trusting seen state-action pairs in the dataset, it does not perform well in datasets with more diverse data distribution including random and medium-replay datasets, while generally matches the performance of MA-TD3+BC in datasets with more narrow distribution including medium and expert. MA-CQL matches or outperforms MA-TD3+BC in datasets with lower quality except for the expert dataset, as it does not rely on constraining the learning policy to stay close to the behavior policy. Our OMAR method significantly outperforms all baseline methods and achieves state-of-the-art performance. We attribute the performance gain to the actor rectification scheme that is independent of data quality and improves global optimization. In addition, OMAR does not incur much computation cost and only takes 4.7% more runtime on average compared with that of MA-CQL.

Table 3: Averaged normalized score of OMAR and baselines in multi-agent particle environments.

		MA-ICQ	MA-TD3+BC	MA-CQL	OMAR
Random	Cooperative navigation	6.3 ± 3.5	9.8 ± 4.9	24.0 ± 9.8	**34.4** ± 5.3
	Predator-prey	2.2 ± 2.6	5.7 ± 3.5	5.0 ± 8.2	**11.1** ± 2.8
	World	1.0 ± 3.2	2.8 ± 5.5	0.6 ± 2.0	**5.9** ± 5.2
Medium -replay	Cooperative navigation	13.6 ± 5.7	15.4 ± 5.6	20.0 ± 8.4	**37.9** ± 12.3
	Predator-prey	34.5 ± 27.8	28.7 ± 20.9	24.8 ± 17.3	**47.1** ± 15.3
	World	12.0 ± 9.1	17.4 ± 8.1	29.6 ± 13.8	**42.9** ± 19.5
Medium	Cooperative navigation	29.3 ± 5.5	29.3 ± 4.8	34.1 ± 7.2	**47.9** ± 18.9
	Predator-prey	63.3 ± 20.0	65.1 ± 29.5	61.7 ± 23.1	**66.7** ± 23.2
	World	71.9 ± 20.0	73.4 ± 9.3	58.6 ± 11.2	**74.6** ± 11.5
Expert	Cooperative navigation	104.0 ± 3.4	108.3 ± 3.3	98.2 ± 5.2	**114.9** ± 2.6
	Predator-prey	113.0 ± 14.4	115.2 ± 12.5	93.9 ± 14.0	**116.2** ± 19.8
	World	109.5 ± 22.8	110.3 ± 21.3	71.9 ± 28.1	**110.4** ± 25.7

4.1.3 ABLATION STUDY

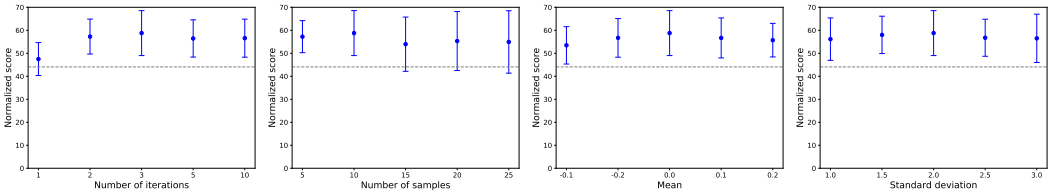
The effect of the regularization coefficient. We first investigate the effect of the regularization coefficient τ in the actor loss in Eq. (2). Figure 3 shows the averaged normalized score of OMAR over different tasks with different values of τ in each kind of dataset. As shown, the performance of OMAR is sensitive to this hyperparameter, which controls the exploitation level of the critic. We find the best value of τ is neither close to 1 nor 0, showing that it is the combination of both policy gradients and the actor rectification that performs well. We also notice that the optimal value of τ is smaller for datasets with lower quality and more diverse data distribution including random and medium-replay, but larger for medium and expert datasets. In addition, the performance of OMAR with all values of τ matches or outperforms that of MA-CQL. This is the only hyperparameter that needs to be tuned in OMAR beyond MA-CQL.



(a) Random. (b) Medium-replay. (c) Medium. (d) Expert.

Figure 3: Ablation study on the effect of the regularization coefficient in different types of datasets.

The effect of key hyperparameters in the sampling scheme. Core hyperparameters for our sampling mechanism involves the number of iterations, the number of sampled actions, and the initial mean and standard deviation of the Gaussian distribution. Figures 4(a)-(d) show the performance comparison of OMAR with different values of these hyperparameters in the cooperative navigation task, where the grey dotted line corresponds to the normalized score of MA-CQL. As shown, our sampling mechanism is not sensitive to these hyperparameters, and we fix them to be the set with the best performance.



(a) Number of iterations. (b) Number of samples. (c) Mean. (d) Standard deviation.

Figure 4: Ablation study on the effect of key hyperparameters in the sampling mechanism averaged over different types of datasets.

The effect of the sampling mechanism. We now analyze the effect of the zeroth-order optimization methods in OMAR, and compare it against random shooting and the cross-entropy method (CEM) (De Boer et al., 2005) in the cooperative navigation task. As shown in Table 4, our sampling mechanism significantly outperforms the random sampling scheme and CEM, with a larger margin in datasets with lower quality including random and medium-replay. The proposed sampling technique incorporates more samples into the distribution updates more effectively.

Table 4: Ablation study of OMAR with different sampling mechanisms in different types of datasets.

	Random	Medium-replay	Medium	Expert
OMAR (random)	24.3 ± 7.0	23.5 ± 5.3	41.2 ± 11.1	101.0 ± 5.2
OMAR (CEM)	25.8 ± 7.3	32.6 ± 5.1	45.0 ± 13.3	106.4 ± 13.8
OMAR	**34.4** ± 5.3	**37.9** ± 12.3	**47.9** ± 18.9	**114.9** ± 2.6

4.1.4 Applicability on Centralized Training with Decentralized Execution

In this section, we demonstrate the versatility of the method and show that it can also be applied and beneficial to methods based on centralized critics under the CTDE paradigm. Specifically, all baseline methods are built upon the MATD3 algorithm (Ackermann et al., 2019) using centralized critics as detailed in Section 2.1. Table 5 summarizes the averaged normalized score of different algorithms in each kind of dataset. As shown, OMAR (centralized) also significantly outperforms MA-ICQ (centralized) and MA-CQL (centralized), and matches the performance of MA-TD3+BC (centralized) in the expert dataset while outperforming it in other datasets.

Table 5: The average normalized score of different methods based on MATD3 with centralized critics under the CTDE paradigm.

	Random	Medium-reply	Medium	Expert
MA-ICQ	5.2 ± 5.5	10.1 ± 4.6	27.4 ± 5.3	96.7 ± 4.1
MA-TD3+BC	7.9 ± 2.2	9.3 ± 9.1	29.4 ± 3.7	**108.1** ± 3.3
MA-CQL	12.8 ± 4.9	11.2 ± 6.6	26.3 ± 13.3	69.5 ± 15.7
OMAR	**21.6** ± 4.6	**19.1** ± 9.2	**33.7** ± 14.5	**105.9** ± 3.6

4.2 Multi-Agent MuJoCo

In this section, we investigate whether OMAR can scale to more complex continuous control multi-agent tasks. Peng et al. (2020) introduce multi-agent locomotion tasks which extends the high-

dimensional MuJoCo locomotion tasks in the single-agent setting to the multi-agent case. We consider the two-agent HalfCheetah task (Kim et al., 2021) as shown in Appendix B.1, where the first and second agents control different parts of joints of the robot. Agents need to cooperate to make the robot run forward by coordinating the actions. We also construct different types of datasets following Fu et al. (2020) the same as in Section 4.1.1. Table 6 summarizes the average normalized scores in each kind of dataset in multi-agent HalfCheetah. As shown, OMAR significantly outperforms baseline methods in random, medium-replay, and medium datasets, and matches the performance of MA-TD3+BC in expert, demonstrating its effectiveness to scale to more complex control tasks.

Table 6: Average normalized score of different methods in multi-agent HalfCheetah.

	Random	Medium-reply	Medium	Expert
MA-ICQ	7.4 ± 0.0	35.6 ± 2.7	73.6 ± 5.0	110.6 ± 3.3
MA-TD3+BC	7.4 ± 0.0	27.1 ± 5.5	75.5 ± 3.7	$\mathbf{114.4} \pm 3.8$
MA-CQL	7.4 ± 0.0	41.2 ± 10.1	50.4 ± 10.8	64.2 ± 24.9
OMAR	$\mathbf{15.4} \pm 12.3$	$\mathbf{57.7} \pm 5.1$	$\mathbf{80.7} \pm 10.2$	$\mathbf{113.5} \pm 4.3$

5 Related Work

Offline reinforcement learning. Many recent papers achieve improvements in offline RL (Wu et al., 2019; Kumar et al., 2019; Yu et al., 2020; Kidambi et al., 2020) that address the extrapolation error. Behavior regularization typically compels the learning policy to stay close to the behavior policy. Yet, its performance relies heavily on the quality of the dataset. Critic regularization approaches typically add a regularizer to the critic loss that pushes down Q-values for actions sampled from a given policy (Kumar et al., 2020). As discussed above, it can be difficult for the actor to best leverage the global information in the critic as policy gradient methods are prone to local optima, which is particularly important in the offline multi-agent setting.

Multi-agent reinforcement learning. A number of multi-agent policy gradient algorithms train agents based on centralized value functions (Lowe et al., 2017; Foerster et al., 2018; Yu et al., 2021) while another line of research focuses on decentralized training (de Witt et al., 2020). Yang et al. (2021) show that the extrapolation error in offline RL can be more severe in the multi-agent setting than the single-agent case due to the exponentially sized joint action space w.r.t. the number of agents. In addition, it presents a critical challenge in the decentralized setting when the datasets for each agent only consist of its own action instead of the joint action (Jiang & Lu, 2021). Jiang & Lu (2021) address the challenges based on the behavior regularization BCQ (Fujimoto et al., 2019) algorithm while Yang et al. (2021) propose to estimate the target value based on the next action from the dataset. As a result, both methods largely depend on the quality of the dataset.

Zeroth-order optimization method. It has been recently shown in (Such et al., 2017; Conti et al., 2017; Mania et al., 2018) that evolutionary strategies (ES) emerge as another paradigm for continuous control. Recent research shows that it is potential to combine RL with ES to reap the best of both worlds (Khadka & Tumer, 2018; Pourchot & Sigaud, 2019) in the high-dimensional parameter space for the actor. Sun et al. (2020) replace the policy gradient update via supervised learning based on sampled noises from random shooting. Kalashnikov et al. (2018); Lim et al. (2018); Simmons-Edler et al. (2019); Peng et al. (2020) extend Q-learning based approaches to handle continuous action space based on the popular cross-entropy method (CEM) in ES.

6 Conclusion

In this paper, we identify the problem that when extending conservatism-based RL algorithms to offline multi-agent scenarios, the performance degrades along increasing number of agents. To tackle this problem, propose Offline Multi-Agent RL with Actor Rectification (OMAR) that combines first-order policy gradient with zeroth-order optimization. We find that OMAR can successfully help the actor escape from bad local optima and consequently find better actions. OMAR achieves state-of-the-art performance on multi-agent continuous control tasks empirically.

REPRODUCIBILITY STATEMENT

We include all details for our experiments in Appendix B.1 including open-source implementations of environments and algorithms, with a detailed description for the hyperparameters and network structures. For Theorem 1, its proof can be found in Appendix A. The code will be open-sourced upon publication of the work at `https://sites.google.com/view/iclr2022omar`.

REFERENCES

Johannes Ackermann, Volker Gabler, Takayuki Osa, and Masashi Sugiyama. Reducing overestimation bias in multi-agent domains using double centralized critics. *arXiv preprint arXiv:1910.01465*, 2019.

Zafarali Ahmed, Nicolas Le Roux, Mohammad Norouzi, and Dale Schuurmans. Understanding the impact of entropy on policy optimization. In *International Conference on Machine Learning*, pp. 151–160. PMLR, 2019.

Christopher Amato. Decision-making under uncertainty in multi-agent and multi-robot systems: Planning and learning. In *IJCAI*, pp. 5662–5666, 2018.

Edoardo Conti, Vashisht Madhavan, Felipe Petroski Such, Joel Lehman, Kenneth O Stanley, and Jeff Clune. Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents. *arXiv preprint arXiv:1712.06560*, 2017.

Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in Neural Information Processing Systems*, pp. 2933–2941, 2014.

Pieter-Tjerk De Boer, Dirk P Kroese, Shie Mannor, and Reuven Y Rubinstein. A tutorial on the cross-entropy method. *Annals of operations research*, 134(1):19–67, 2005.

Christian Schroeder de Witt, Tarun Gupta, Denys Makoviichuk, Viktor Makoviychuk, Philip HS Torr, Mingfei Sun, and Shimon Whiteson. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533*, 2020.

Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *arXiv preprint arXiv:2106.06860*, 2021.

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pp. 1587–1596. PMLR, 2018.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pp. 2052–2062. PMLR, 2019.

Rong Ge, Jason D Lee, and Tengyu Ma. Learning one-hidden-layer neural networks with landscape design. *arXiv preprint arXiv:1711.00501*, 2017.

Junling Hu, Michael P Wellman, et al. Multiagent reinforcement learning: theoretical framework and an algorithm. In *ICML*, volume 98, pp. 242–250. Citeseer, 1998.

Shariq Iqbal and Fei Sha. Actor-attention-critic for multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 2961–2970. PMLR, 2019.

Jiechuan Jiang and Zongqing Lu. Offline decentralized multi-agent reinforcement learning. *arXiv preprint arXiv:2108.01832*, 2021.

Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

Shauharda Khadka and Kagan Tumer. Evolution-guided policy gradient in reinforcement learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 1196–1208, 2018.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. *arXiv preprint arXiv:2005.05951*, 2020.

Dong Ki Kim, Miao Liu, Matthew D Riemer, Chuangchuang Sun, Marwa Abdulhai, Golnaz Habibi, Sebastian Lopez-Cot, Gerald Tesauro, and Jonathan How. A policy gradient algorithm for learning to learn in multiagent reinforcement learning. In *International Conference on Machine Learning*, pp. 5541–5550. PMLR, 2021.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Ilya Kostrikov, Rob Fergus, Jonathan Tompson, and Ofir Nachum. Offline reinforcement learning with fisher divergence critic regularization. In *International Conference on Machine Learning*, pp. 5774–5783. PMLR, 2021.

Aviral Kumar, Justin Fu, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *arXiv preprint arXiv:1906.00949*, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1179–1191, 2020.

Seunghyun Lee, Younggyo Seo, Kimin Lee, Pieter Abbeel, and Jinwoo Shin. Offline-to-online reinforcement learning via balanced replay and pessimistic q-ensemble. *arXiv preprint arXiv:2107.00591*, 2021.

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.

Sungsu Lim, Ajin Joseph, Lei Le, Yangchen Pan, and Martha White. Actor-expert: A framework for using q-learning in continuous action spaces. *arXiv preprint arXiv:1810.09103*, 2018.

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

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in Neural Information Processing Systems*, 30:6379–6390, 2017.

Kendall Lowrey, Aravind Rajeswaran, Sham Kakade, Emanuel Todorov, and Igor Mordatch. Plan online, learn offline: Efficient learning and exploration via model-based control. *arXiv preprint arXiv:1811.01848*, 2018.

Xueguang Lyu, Yuchen Xiao, Brett Daley, and Christopher Amato. Contrasting centralized and decentralized critics in multi-agent reinforcement learning. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 844–852, 2021.

Horia Mania, Aurelia Guy, and Benjamin Recht. Simple random search provides a competitive approach to reinforcement learning. *arXiv preprint arXiv:1803.07055*, 2018.

Ofir Nachum, Mohammad Norouzi, and Dale Schuurmans. Improving policy gradient by exploring under-appreciated rewards. *arXiv preprint arXiv:1611.09321*, 2016.

Ofir Nachum, Bo Dai, Ilya Kostrikov, Yinlam Chow, Lihong Li, and Dale Schuurmans. Algaedice: Policy gradient from arbitrary experience. *arXiv preprint arXiv:1912.02074*, 2019.

Anusha Nagabandi, Kurt Konolige, Sergey Levine, and Vikash Kumar. Deep dynamics models for learning dexterous manipulation. In *Conference on Robot Learning*, pp. 1101–1112. PMLR, 2020.

Bei Peng, Tabish Rashid, Christian A Schroeder de Witt, Pierre-Alexandre Kamienny, Philip HS Torr, Wendelin Böhmer, and Shimon Whiteson. Facmac: Factored multi-agent centralised policy gradients. *arXiv preprint arXiv:2003.06709*, 2020.

Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA ARTIFICIAL INTELLIGENCE AND PSYCHOLOGY ..., 1989.

Pourchot and Sigaud. CEM-RL: Combining evolutionary and gradient-based methods for policy search. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=BkeU5j0ctQ`.

Reuven Y Rubinstein and Dirk P Kroese. *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation and machine learning*. Springer Science & Business Media, 2013.

Dorsa Sadigh, Shankar Sastry, Sanjit A Seshia, and Anca D Dragan. Planning for autonomous cars that leverage effects on human actions. In *Robotics: Science and Systems*, volume 2, pp. 1–9. Ann Arbor, MI, USA, 2016.

Itay Safran and Ohad Shamir. Spurious local minima are common in two-layer relu neural networks. *arXiv preprint arXiv:1712.08968*, 2017.

Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. The starcraft multi-agent challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2186–2188, 2019.

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

Riley Simmons-Edler, Ben Eisner, Eric Mitchell, Sebastian Seung, and Daniel Lee. Q-learning for continuous actions with cross-entropy guided policies. *arXiv preprint arXiv:1903.10605*, 2019.

Felipe Petroski Such, Vashisht Madhavan, Edoardo Conti, Joel Lehman, Kenneth O Stanley, and Jeff Clune. Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning. *arXiv preprint arXiv:1712.06567*, 2017.

Hao Sun, Ziping Xu, Yuhang Song, Meng Fang, Jiechao Xiong, Bo Dai, Zhengyou Zhang, and Bolei Zhou. Zeroth-order supervised policy improvement. *arXiv preprint arXiv:2006.06600*, 2020.

Philip S Thomas. *Safe reinforcement learning*. PhD thesis, University of Massachusetts Libraries, 2015.

Emmanouil-Vasileios Vlatakis-Gkaragkounis, Lampros Flokas, and Georgios Piliouras. Efficiently avoiding saddle points with zero order methods: No gradients required. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Grady Williams, Andrew Aldrich, and Evangelos Theodorou. Model predictive path integral control using covariance variable importance sampling. *arXiv preprint arXiv:1509.01149*, 2015.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Yiqin Yang, Xiaoteng Ma, Chenghao Li, Zewu Zheng, Qiyuan Zhang, Gao Huang, Jun Yang, and Qianchuan Zhao. Believe what you see: Implicit constraint approach for offline multi-agent reinforcement learning. *arXiv preprint arXiv:2106.03400*, 2021.

Chao Yu, Akash Velu, Eugene Vinitsky, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of mappo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955*, 2021.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020.

A PROOF OF THEOREM 1

Theorem 1. *Let π_i^* be the policy obtained by optimizing Eq. (2). Then, we have that $J(\pi_i^*) - J(\hat{\pi}_{\beta_i}) \geq \frac{\alpha}{1-\gamma}\mathbb{E}_{o_i\sim d^{\pi_i^*}(o_i)}\left[D(\pi_i^*, \hat{\pi}_{\beta_i})(o_i)\right] + \frac{\tau}{1-\tau}\mathbb{E}_{o_i\sim d^{\pi_i^*}(o_i)}\left[(\pi_i^*(o_i) - \hat{a}_i)^2\right] - \frac{\tau}{1-\tau}\mathbb{E}_{o_i\sim d^{\hat{\pi}_{\beta_i}}(o_i), a_i\sim\hat{\pi}_{\beta_i}}\left[(a_i - \hat{a}_i)^2\right]$, where $D(\pi_i, \hat{\pi}_{\beta_i})(o_i) = \frac{1}{\hat{\pi}_{\beta_i}(\pi_i(o_i)|o_i)} - 1$, and $d^{\pi_i}(o_i)$ is the marginal discounted distribution of observations of policy π_i.*

Proof. For OMAR, we have the following iterative update for agent i:

$$\begin{aligned}\hat{Q}_i^{k+1} \leftarrow \arg\min_{Q_i} \alpha\mathbb{E}_{o_i\sim\mathcal{D}_i}&\left[\mathbb{E}_{a_i\sim\tilde{\pi}_i(a_i|o_i)}\left[Q_i(o_i,a_i)\right] - \mathbb{E}_{a_i\sim\hat{\pi}_{\beta_i}(a_i|o_i)}\left[Q_i(o_i,a_i)\right]\right] \\ &+ \frac{1}{2}\mathbb{E}_{o_i,a_i,o_i'\sim\mathcal{D}}\left[\left(Q_i(o_i,a_i) - \hat{\mathcal{B}}^{\pi_i}\hat{Q}_i^k(o_i,a_i)\right)^2\right],\end{aligned} \tag{4}$$

where $\tilde{\pi}_i(a_i|o_i) = 1$ if and only if $a_i = \pi_i(o_i)$.

Let $\hat{Q}_i^{k+1}$ be the fixed point of solving Equation (4) by setting the derivative of Eq. (4) with respect to Q_i to be 0, then we have that

$$\hat{Q}_i^{k+1}(o_i,a_i) = \hat{\mathcal{B}}^{\pi_i}\hat{Q}_i^k(o_i,a_i) - \alpha\left(\frac{I_{a_i=\pi_i(o_i)}}{\hat{\pi}_{\beta_i}(a_i|o_i)} - 1\right), \tag{5}$$

where I is the indicator function.

Denote $D(\pi_i, \hat{\pi}_{\beta_i})(o_i) = \frac{1}{\hat{\pi}_{\beta_i}(\pi_i(o_i)|o_i)} - 1$, and we obtain the difference between the value function $\hat{V}_i(o_i)$ and the original value function as:

$$\hat{V}_i(o_i) = V_i(o_i) - \alpha D(\pi_i, \hat{\pi}_{\beta_i})(o_i), \tag{6}$$

Then, the policy that minimizes the loss function defined in Eq. (2) is equivalently obtained by maximizing

$$(1-\tau)\left(J(\pi_i) - \alpha\frac{1}{1-\gamma}\mathbb{E}_{o_i\sim d_{\hat{M}_i}^{\pi_i}(o_i)}\left[D(\pi_i, \hat{\pi}_{\beta_i})(o_i)\right]\right) - \tau\mathbb{E}_{o_i\sim d_{\hat{M}_i}^{\pi_i}(o_i)}\left[(\pi_i(o_i) - \hat{a}_i)^2\right]. \tag{7}$$

Therefore, we obtain that

$$\begin{aligned}&(1-\tau)\left(J(\pi_i^*) - \alpha\frac{1}{1-\gamma}\mathbb{E}_{o_i\sim d_{\hat{M}_i}^{\pi_i^*}(o_i)}\left[D(\pi_i^*, \hat{\pi}_{\beta_i})(o_i)\right]\right) - \tau\mathbb{E}_{o_i\sim d_{\hat{M}_i}^{\pi_i^*}(o_i)}\left[(\pi_i^*(o_i) - \hat{a}_i)^2\right] \\ \geq&(1-\tau)J(\hat{\pi}_{\beta_i}) - \tau\mathbb{E}_{o_i\sim d_{\hat{M}_i}^{\hat{\pi}_{\beta_i}}(o_i), a_i\sim\hat{\pi}_{\beta_i}(a_i|o_i)}\left[(a_i - \hat{a}_i)^2\right].\end{aligned} \tag{8}$$

Then, from Eq. (8) we obtain the result. □

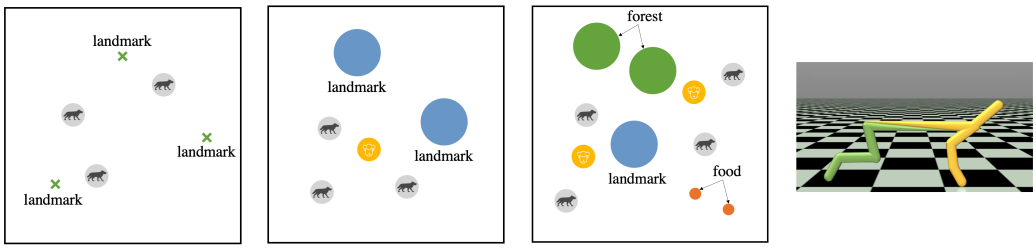
B MORE DETAILS OF THE EXPERIMENTS

B.1 EXPERIMENTAL SETUP

Tasks. We adopt the open-source implementations for multi-agent particle environments[2] from (Lowe et al., 2017) and Multi-Agent MuJoCo[3] from (Peng et al., 2020). Figure 5 illustrates the tasks.

[2] https://github.com/openai/multiagent-particle-envs

[3] https://github.com/schroederdewitt/multiagent_mujoco



(a) Cooperative navigation. (b) Predator-prey. (c) World. (d) Two-agent HalfCheetah.

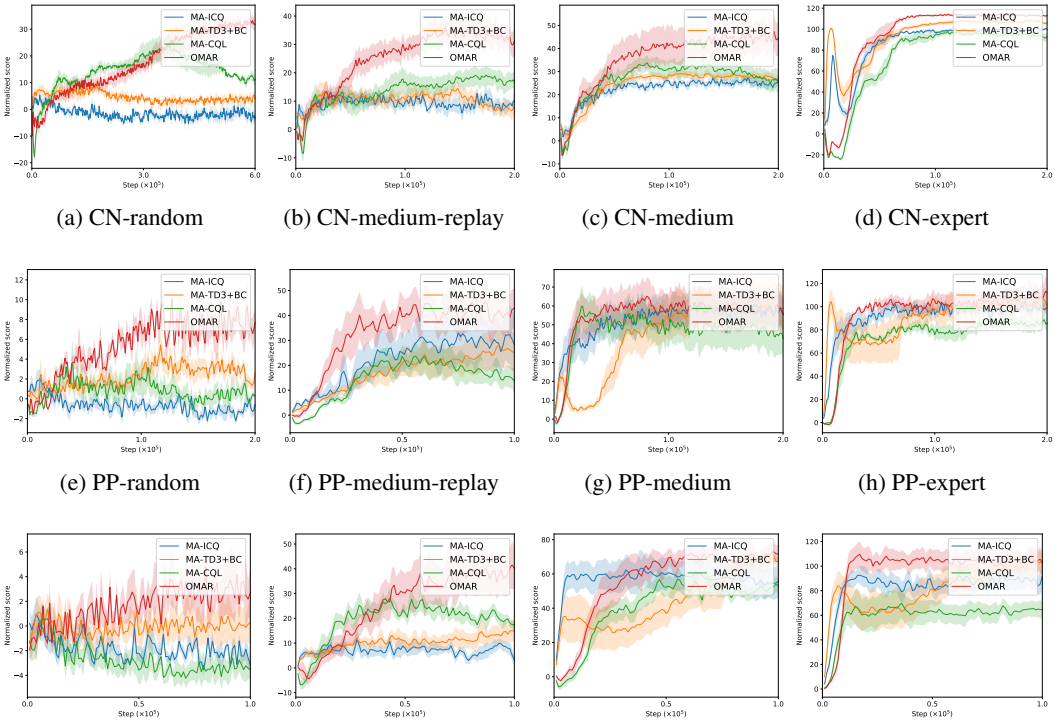
Figure 5: Multi-agent particle environments and Multi-Agent HalfCheetah.

Baselines. All baseline methods are implemented based on an open-source implementation[4] from (Iqbal & Sha, 2019), where we implement MA-TD3+BC[5], MA-CQL[6], and MA-ICQ[7] based on authors' open-source implementations with fine-tuned hyperparameters. For MA-CQL, we tune a best critic regularization coefficient from $\{0.1, 0.5, 1.0, 5.0\}$ following (Kumar et al., 2020) for each task. Specifically, we use the discount factor γ of 0.99. We sample a minibatch of 1024 samples from the dataset for updating each agent's actor and critic using the Adam (Kingma & Ba, 2014) optimizer with the learning rate to be 0.01. The target networks for the actor and critic are soft updated with the update rate to be 0.01. Both the actor and critic networks are feedforward networks consisting of two hidden layers with 64 neurons per layer using ReLU activation. For OMAR, the only hyperparameter that requires tuning is the regularization coefficient λ, where we use a smaller value for datasets with more diverse data distribution in random and medium-replay with a value of 0.5, while we use a larger value for datasets with more narrow data distribution in medium and expert with values of 0.7 and 0.9 respectively. As OMAR is insensitive to the hyperparameters of the sampling mechanism, we set them to a fixed set of values for all types of datasets in all tasks, where the number of iteration is 3, the number of samples is 10, the mean is 0.0, and the standard deviation is 2.0. The code will be released upon publication of the paper.

[4] https://github.com/shariqiqbal2810/maddpg-pytorch
[5] https://github.com/sfujim/TD3_BC
[6] https://github.com/aviralkumar2907/CQL
[7] https://github.com/YiqinYang/ICQ

B.2 LEARNING CURVES

Figure 6 demonstrates the learning curves of MA-ICQ, MA-TD3+BC, MA-CQL and OMAR in different types of datasets in multi-agent particle environments, where the solid line and shaded region represent mean and standard deviation, respectively.



(a) CN-random (b) CN-medium-replay (c) CN-medium (d) CN-expert

(e) PP-random (f) PP-medium-replay (g) PP-medium (h) PP-expert

(i) W-random (j) W-medium-replay (k) W-medium (l) W-expert

Figure 6: Learning curves of MA-ICQ, MA-TD3+BC, MA-CQL, and OMAR in multi-agent particle environments (CN, PP, and W is abbreviated for cooperative navigation, predator-prey, and world respectively).