Reflective Policy Optimization

Yaozhong Gan [*] [1] Renye Yan [*] [1] Zhe Wu [1] Junliang Xing [1]

Abstract

On-policy reinforcement learning methods, like Trust Region Policy Optimization (TRPO) and Proximal Policy Optimization (PPO), often demand extensive data per update, leading to sample inefficiency. This paper introduces Reflective Policy Optimization (RPO), a novel on-policy extension that amalgamates past and future state-action information for policy optimization. This approach empowers the agent for introspection, allowing modifications to its actions within the current state. Theoretical analysis confirms that policy performance is monotonically improved and contracts the solution space, consequently expediting the convergence procedure. Empirical results demonstrate RPO's feasibility and efficacy in two reinforcement learning benchmarks, culminating in superior sample efficiency. The source code of this work is available at `https://github.com/Edgargan/RPO`.

1. Introduction

On-policy reinforcement learning (RL) aims to learn an optimal mapping from a sequence of states to actions based on rewarding criteria acquired through trajectories generated by interacting with the underlying environment. Proximal Policy Optimization (PPO) (Schulman et al., 2017) is one of the most typical algorithms in this category, owing to its simplicity and effectiveness. It has been successfully applied in various domains, including Atari games (Mnih et al., 2015), continuous control tasks (Dhariwal et al., 2017), and robot control (Lillicrap et al., 2016). However, existing algorithms optimize the policy based on a state-action pair and do not directly consider the impact of subsequent states and actions in the trajectory. This limitation inevitably gives rise to the sample inefficiency problem.

In prior studies (Mnih et al., 2015; van Hasselt et al., 2016; Schulman et al., 2015; 2017; Haarnoja et al., 2018; Silver et al., 2014; Fujimoto et al., 2018), the prevalent approach involves optimizing the policy using the value function of the current state. The value function potentially contains information about the subsequent data. However, a pertinent question emerges: Is optimizing a policy solely based on the value function the fastest (optimal) path to convergence? The answer is no, as this approach may overlook other crucial factors.

To illustrate this answer, consider an environment with a "cliff". If an agent takes an action leading to falling off the cliff under a state, it must learn to avoid the action and the associated state. Returning to this state is perilous and could potentially trigger the same action. Therefore, the agent must actively avoid this state to enhance safety. The preceding action leading to this state must also be avoided, anticipating the possibility of re-entering that state again.

A similar scenario unfolds in a "treasure" environment, where an agent performs an action resulting in a large reward. Subsequent data imparts positive and negative insights into previous states and actions. Therefore, optimizing the previous action should incorporate information from subsequent state-action pairs, not relying solely on the value function. Intuitively, leveraging subsequent data directly can expedite algorithm convergence and enhance sample efficiency. Unfortunately, most existing algorithms lack this capability, which directly exploits the relationship between pairs of trajectory data for policy optimization.

To address the above issues with better sample efficiency, we introduce a new on-policy algorithm that optimizes the current policy by explicitly considering the relationship between the previous and subsequent state-action pairs in the sampled trajectories. Specifically, the proposed algorithm evaluates the current state-action pair and the impact of the subsequent pair of trajectories. It provides a more comprehensive perspective than traditional value function-based policy optimization. This approach enables the optimized policy to adjust its actions based on positive and negative information from subsequent states, thereby effectively reflecting the policy. We thus name the proposed algorithm as Reflective Policy Optimization (RPO).

The RPO algorithm, as proposed, directly focuses on policy optimization rather than solely on evaluating the value

[*]Equal contribution [1]QiYuan Lab. Correspondence to: Junliang Xing <xingjunliang@qiyuanlab.com>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

function. This distinction separates it from multi-step reinforcement learning methods (De Asis et al., 2018; Duan & Wainwright, 2023; Hernandez-Garcia & Sutton, 2019). Our proposed algorithm takes a direct approach by incorporating previous and subsequent trajectory information for policy optimization, establishing more clearly theoretical properties. We also derive a novel policy improvement lower bound, illustrating that, in addition to ensuring the desirable property of monotonic performance improvement, our method effectively reduces the solution space of the optimized policy, thus significantly accelerating the algorithm's convergence procedure. Furthermore, our method improves sample efficiency.

We incorporate our proposed algorithm with the PPO's clipping mechanism (Schulman et al., 2017) to provide a practical implementation. Following standard settings, we validate the effectiveness of our algorithm by utilizing an illustrative toy example, shedding light on the underlying working mechanism of RPO. Additionally, we showcase superior performance on widely recognized RL benchmarks, such as MuJoCo (Todorov et al., 2012) and Atari games (Brockman et al., 2016).

2. Preliminaries

2.1. Markov Decision Process

Commonly, the reinforcement learning problem can be modeled as a Markov Decision Process (MDP), which is described by the tuple $\langle \mathcal{S}, \mathcal{A}, P, R, \gamma \rangle$ (Sutton & Barto, 1998). $\mathcal{S}$ and $\mathcal{A}$ are the state space and action space respectively. The function $P(s'|s,a) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \longmapsto [0,1]$ is the transition probability function from state s to state s' under action a. The function $R(s,a) : \mathcal{S} \times \mathcal{A} \longmapsto \mathbb{R}$ is the reward function. And $\gamma \in [0,1)$ is the discount factor for long-horizon returns. In a state s, the agent performs an action a according to a stochastic policy $\pi : \mathcal{S} \times \mathcal{A} \longmapsto [0,1]$ (satisfies $\sum_a \pi(a|s) = 1$). The environment returns a reward $R(s,a)$ and a new state s' according to the transition function $P(s'|s,a)$. The agent interacts with the MDP to give a trajectory τ of states, actions, and rewards: $s_0, a_0, R(s_0,a_0), \cdots, s_t, a_t, R(s_t,a_t), \cdots$ over $\mathcal{S} \times \mathcal{A} \times \mathbb{R}$ (Silver et al., 2014). Given a policy π, under a state s_t and a action a_t, the state-action value function and state-value function are defined as

$$Q^\pi(s_t,a_t) = \mathbb{E}_{\tau\sim\pi}[G_t|s_t,a_t], V^\pi(s_t) = \mathbb{E}_{\tau\sim\pi}[G_t|s_t],$$

where $G_t = \sum_{i=0}^{\infty} \gamma^i R_{t+i}$ is the discount return, and $R_t = R(s_t,a_t)$.

It is clear that $V^\pi(s_t) = \mathbb{E}_{a_t\sim\pi} Q^\pi(s_t,a_t)$. Correspondingly, advantage function can be represented $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$. We know that $\sum_a \pi(a|s) A^\pi(s,a) = 0$.

Let ρ^π be a normalized discount state visitation distribution, defined

$$\rho^\pi(s) = (1-\gamma)\sum_{t=0}^{\infty} \gamma^t \mathbb{P}(s_t = s|\rho_0, \pi),$$

where ρ_0 is the initial state distribution (Kakade & Langford, 2002). Similarly, $\rho^\pi(\cdot|s,a)$ can be defined and denotes the conditional visitation distribution under state s and action a. And the normalized discount state-action visitation distribution can be represented $\rho^\pi(s,a) = \rho^\pi(s)\pi(a|s)$. We make it clear from the context whether ρ^π refers to the state or state-action distribution.

The goal is to learn a policy that maximizes the expected total discounted reward $\eta(\pi)$, defined

$$\eta(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{i=0}^{\infty} \gamma^i R(s_i,a_i)\right].$$

The following identity indicates that the distance between the policy performance of π and $\hat{\pi}$ is related to the advantage over π (Kakade & Langford, 2002):

$$\eta(\pi) = \eta(\hat{\pi}) + \frac{1}{1-\gamma}\mathbb{E}_{s,a\sim\rho^\pi}\left[A^{\hat{\pi}}(s,a)\right]. \tag{1}$$

Some admirable algorithms obtain good properties by modifying the right-hand side of Eqn. (1), for example, Trust Region Policy Optimization (TRPO) algorithm (Schulman et al., 2015) optimizes the lower bound of policy improvement by replacing ρ^π with $\rho^{\hat{\pi}}$ under state s, and offers better theoretical properties, *i.e.* monotonic improvement of policy improvement.

3. The Generalized Surrogate Function

In this section, we establish a recurrence form by providing the equation relationships before and after the replacement of TRPO. Further, we reach general conclusions by extending TRPO with subsequent state-action pairs.

Lemma 3.1. *Consider a current policy $\hat{\pi}$, and any policies π, we have*

$$\begin{aligned}&\mathbb{E}_{s,a\sim\rho^\pi} A^{\hat{\pi}}(s,a) - \mathbb{E}_{s\sim\rho^{\hat{\pi}},a\sim\pi} A^{\hat{\pi}}(s,a)\\ =&\frac{\gamma}{1-\gamma}\mathbb{E}_{s,a\sim\rho^{\hat{\pi}}}[\frac{\pi(a|s)}{\hat{\pi}(a|s)} - 1]\mathbb{E}_{s',a'\sim\rho^\pi(\cdot|s,a)} A^{\hat{\pi}}(s',a').\end{aligned}$$

The proof of this lemma is given in Appendix A.4.

Note that from this lemma, the difference between the original formula and the replaced one is relevant to the normalized discount subsequent state-action visitation distribution $\rho^\pi(\cdot|s,a)$. By the boundary of the right-hand side of the equation, it is easy to obtain Theorem 1 of the paper (Schulman et al., 2015) and Theorem 1 of the paper (Achiam et al., 2017). From this lemma, we constructed a relationship between the current visitation distributions $(s,a) \sim \rho^\pi(\cdot)$ and the next $(s',a') \sim \rho^\pi(\cdot|s,a)$.

Theorem 3.2. *Consider a current policy $\hat{\pi}$, and any policies π, we have*

$$\eta(\pi) = \eta(\hat{\pi}) + \sum_{i=0}^{k-1} \alpha_i L_i(\pi, \hat{\pi}) + \beta_k G_k(\pi, \hat{\pi}), \quad (2)$$

where

$$L_i(\pi, \hat{\pi}) = \mathop{\mathbb{E}}_{\substack{s_0, a_0 \sim \rho^{\hat{\pi}}(\cdot) \\ \cdots \\ s_{i-1}, a_{i-1} \sim \rho^{\hat{\pi}}(\cdot | s_{i-2}, a_{i-2})}} \prod_{t=0}^{i-1} (r_t - 1) \cdot l_i(\pi, \hat{\pi}),$$

$$G_k(\pi, \hat{\pi}) = \mathop{\mathbb{E}}_{\substack{s_0, a_0 \sim \rho^{\hat{\pi}}(\cdot) \\ \cdots \\ s_{k-1}, a_{k-1} \sim \rho^{\hat{\pi}}(\cdot | s_{k-2}, a_{k-2})}} \prod_{t=0}^{k-1} (r_t - 1) \cdot g_k(\pi, \hat{\pi}),$$

$$l_i(\pi, \hat{\pi}) = \mathbb{E}_{s_i \sim \rho^{\hat{\pi}}(\cdot | s_{i-1}, a_{i-1}), a_i \sim \pi(\cdot | s_i)} A^{\hat{\pi}}(s_i, a_i),$$

$$g_k(\pi, \hat{\pi}) = \mathbb{E}_{s_k, a_k \sim \rho^{\pi}(\cdot | s_{k-1}, a_{k-1})} A^{\hat{\pi}}(s_k, a_k),$$

and

$$r_t = \frac{\pi(a_t | s_t)}{\hat{\pi}(a_t | s_t)}, \ \alpha_i = \frac{\gamma^i}{(1-\gamma)^{i+1}}, \ \beta_k = \frac{\gamma^k}{(1-\gamma)^{k+1}}.$$

We define that $L_0(\pi, \hat{\pi}) = \mathbb{E}_{s_0, a_0 \sim \rho^{\hat{\pi}}(\cdot)} r_0 A^{\hat{\pi}}(s_0, a_0)$, $G_1(\pi, \hat{\pi}) = \mathbb{E}_{s_0, a_0 \sim \rho^{\hat{\pi}}(\cdot); s_1, a_1 \sim \rho^{\pi}(\cdot | s_0, a_0)} (r_0 - 1) A^{\hat{\pi}}(s_1, a_1)$ *and* $r_0 = \frac{\pi(a_0 | s_0)}{\hat{\pi}(a_0 | s_0)}$.

The proof of this theorem is given in Appendix A.5.

This theorem gives a general form for the difference between the policy performance of π and $\hat{\pi}$ by finite sums. This equation accurately represents the general gap between the performance of π and $\hat{\pi}$ from the trajectory-based. It portrays that subsequent state-action pairs can also directly impact optimizing the current policy. We refer to $\sum_{i=0}^{k-1} \alpha_i L_i(\pi, \hat{\pi})$ as the generalized surrogate objective function.

A slight problem may exist if the generalized surrogate objective function is directly optimized. Consider $L_1(\pi, \hat{\pi})$ in Eqn. (2) as an example. We consider this function without delving into the specific form of the parameters. When the environment is unknown, it can only be optimized by sampling. Considering a special case, the function $L_1(\pi, \hat{\pi})$ is optimized by using a sample (s_0, a_0, s_1, a_1), *i.e.*, $L_1(\pi, \hat{\pi}) \approx (r_0 - 1) r_1 A^{\hat{\pi}}(s_1, a_1)$. If $A^{\hat{\pi}}(s_1, a_1) < 0$ and $r_0 - 1 < 0$, it follows that $(r_0 - 1) r_1 A^{\hat{\pi}}(s_1, a_1) = [(r_0 - 1) A^{\hat{\pi}}(s_1, a_1)] r_1 > 0$. This implies an increase in the probability of a_1. However, when $A^{\hat{\pi}}(s_1, a_1) < 0$, we should decrease the probability of a_1. It's a contradiction. Thus, the term "1" in $r_0 - 1$ may be incorrectly misleading for policy optimization despite the soundness of the theory. This situation exists when the environment is unknown.

Next, we measure the gap between the policy performance $\eta(\pi)$ and $\sum_{i=0}^{k-1} \alpha_i L_i(\pi, \hat{\pi})$.

Corollary 3.3. *According to the definition of G_k, we have*

$$|\beta_k G_k(\pi, \hat{\pi})| \le \frac{\gamma^k}{(1-\gamma)^{k+2}} \epsilon^{k+1} R_{\max},$$

where $\epsilon \triangleq \|\pi - \hat{\pi}\|_1 = \max_s \sum_a |\pi(a|s) - \hat{\pi}(a|s)|$ *and* $R_{\max} \triangleq \max_{s,a} |R(s,a)|$.

The proof of the theorem is given in the Appendix A.6.

Based on Theorem 3.2 and Corollary 3.3, a general lower bound exists for the policy performance of π. This theory makes good theoretical sense, which helps to understand the generalized surrogate function. For the case where $k = 1$, the l_1 norm constraints are replaced by KL constraints. This outcome aligns with the lower bound observed in TRPO.

4. Reflective Policy Optimization

Theoretically, the previous section gave a lower bound for the policy performance of π. Although the generalized surrogate function incorporates the current and subsequent state-action pairs of the trajectory, the inclusion of the term "1" in the function $L_i(\pi, \hat{\pi})$ introduces ambiguity regarding how the subsequent pairs influence the behavior of the policy at the current state, potentially yielding positive or adverse effects. In this section, we have made slight modifications to the generalized surrogate function $L_i(\pi, \hat{\pi})$ in Eqn. (2), defined

$$\hat{L}_i(\pi, \hat{\pi}) = \mathop{\mathbb{E}}_{\substack{s_0, a_0 \sim \rho^{\hat{\pi}}(\cdot) \\ \cdots \\ s_i, a_i \sim \rho^{\hat{\pi}}(\cdot | s_{i-1}, a_{i-1})}} \prod_{t=0}^{i} r_t \cdot A^{\hat{\pi}}(s_i, a_i). \quad (3)$$

It is a natural modification that avoids the problems caused by the "1" item. The following theorem measures the relationship between the function $\hat{L}_i(\pi, \hat{\pi})$ and $\eta(\pi)$.

Theorem 4.1. *Consider a current policy $\hat{\pi}$, and any policies π, we have*

$$\eta(\pi) - \eta(\hat{\pi}) \ge \sum_{i=0}^{k-1} \alpha_i \hat{L}_i(\pi, \hat{\pi}) - \hat{C}_k(\pi, \hat{\pi}), \quad (4)$$

where

$$\hat{C}_k(\pi, \hat{\pi}) = \frac{\gamma R_{\max} \|\pi - \hat{\pi}\|_1}{1-\gamma} \sum_{i=1}^{k-1} \alpha_i + \frac{\gamma^k R_{\max}}{(1-\gamma)^{k+2}} \|\pi - \hat{\pi}\|_1^2,$$

$\hat{L}_i(\pi, \hat{\pi})$ is defined in Eqn. (3) and $\alpha_i = \frac{\gamma^i}{(1-\gamma)^{i+1}}$. *We define that* $\sum_{i=1}^{0} \alpha_i = 0$ *for* $k = 1$.

The proof of this theorem is given in Appendix A.8.

From Theorem 4.1, the first term of the generalized lower bound is referred to as the new generalized surrogate function, while the second term is known as the penalty term.

It is worth noting that TRPO (Schulman et al., 2015) is a special case of the generalized lower bound for $k = 1$. By optimizing the generalized lower bound, we can get a monotonically improving sequence of policies $\{\pi_i\}_{i=0}^{\infty}$, satisfy $\eta(\pi_0) \leq \eta(\pi_1) \leq \cdots$. Next, we intuitively analyze the new generalized surrogate function. The difference between the function $L_i(\pi, \hat{\pi})$ and $\hat{L}_i(\pi, \hat{\pi})$ is very small, involving the removal of the number 1 from the ratios' product. However, their intended meanings are quite distinct. The function $\hat{L}_i(\pi, \hat{\pi})$ can directly utilize the information between the current and subsequent state-action pairs to optimize the current policy. Optimizing this function does not encounter the issues discussed in Section 3.

With $k = 2$, we will explain the optimization procedure in detail. The function $\hat{L}_1(\pi, \hat{\pi})$ contains the ratio of the pair (s, a) and (s', a'). If $A^{\hat{\pi}}(s', a') > 0$, it indicates that the action a' is deemed favorable, and its probability will be increased through algorithm optimization. Simultaneously, the state s' is likely fine as well. To return to this state, we should increase the probability of the action a under state s. In contrast, if $A^{\hat{\pi}}(s', a') < 0$, similar results will be obtained. The agent can reflect on its current behavior based on subsequent information. For $\hat{L}_0(\pi, \hat{\pi})$, the action's a probability can be optimizing using the advantage function $A^{\hat{\pi}}(s, a)$. Therefore, optimizing the current action a will be influenced by both the current and subsequent advantage functions $A^{\hat{\pi}}$, taking them into account.

In this way, the optimized policy will likely foster the agent's reflection, and we observe that optimizing the generalized surrogate function lacks this ability. By utilizing the same trajectory, the agent can acquire more information. Figure 1 depicts an experiment conducted in the CliffWalking environment. The results in Figure 1 indicate that optimizing the new surrogate function reduces the number of falling off the Cliff and faster after reaching the goal G. In the experimental section, we explain this phenomenon in detail.

Theorem 4.1 demonstrates that the generalized lower bound is optimized for any k. As k increases, the generalized lower bound is optimized using subsequent samples to learn implicit relationships of the current and subsequent states and actions data. However, is it suitable when k takes a large value? The answer is no. Let's look at the $\hat{L}_k(\pi, \hat{\pi})$ function individually. This objective function comprises the product of the k ratios and an advantage function. If the ratio is too high, it encounters the problem of high variance (Munos et al., 2016), which, in turn, impacts the algorithm's stability. Given this weakness, a very large value of k may not be practical. In the experimental section 5.1, we discuss the values of k and observe that as long as the agent leverages the relationship between previous and subsequent state-action pairs, it enables the agent to fall into the Cliff less often and to reach the goal G faster.

The experimental results show similarity whether $k = 2$ or $k = 3$. Therefore, the main part of the following discussion is framed in terms of $k = 2$. The following theorem shows that the modified generalized surrogate function has another nice property except for the monotonicity.

Theorem 4.2. *For $k = 2$, defined two sets*

$$\Psi_1 = \left\{\mu \mid \alpha_0 \hat{L}_0(\mu, \hat{\pi}) - \hat{C}_1(\mu, \hat{\pi}) \geq 0, \|\mu - \hat{\pi}\|_1 \leq \frac{1}{2}\right\},$$

$$\Psi_2 = \left\{\mu \mid \alpha_0 \hat{L}_0(\mu, \hat{\pi}) + \alpha_1 \hat{L}_1(\mu, \hat{\pi}) - \hat{C}_2(\mu, \hat{\pi}) \geq 0, \right.$$
$$\left. \|\mu - \hat{\pi}\|_1 \leq \frac{1}{2}\right\},$$

then we have

$$\Psi_2 \subseteq \Psi_1.$$

The proof of the theorem is given in Appendix A.9.

Note that when the old and new policies do not change much, the set Ψ_1 is a solution space of TRPO, and the set Ψ_k corresponds to the solution space of the k-th generalized lower bound. The theorem 4.2 shows that the scale of the solution space of the policy is reduced when $k = 2$. Under certain conditions, the optimal policy $\pi^\star$ belongs to the set Ψ_1 as well as to Ψ_2. Reducing the solution space is possibly more efficient in finding a good policy, and therefore, it is intuitive that the algorithm's convergence procedure can be accelerated. Furthermore, it can improve sample efficiency. Similarly, we can define the solution space of $k = 3, 4, \cdots$ and use the same way to get $\Psi_1 \supseteq \Psi_2 \supseteq \Psi_3 \supseteq \Psi_4 \supseteq \cdots$. Note that $\pi^\star$ is in those sets. It reveals the benefits of using current and subsequent states and actions of trajectory data to optimize the policy. This result provides a promising theoretical basis for our algorithm.

4.1. The Clipped Generalized Surrogate Objection

In the previous subsection discussion, the generalized lower bound function contained the generalized surrogate function and a penalty term. The optimization approach for this lower bound is similar to TRPO, utilizing a linear approximation for the surrogate objective and a quadratic approximation for the penalty term. However, it requires computing the inverse matrix of the quadratic approximation of the penalty term. In particular, the generalized lower bound function also includes the relationship between before and after state-action pairs. It is, therefore, impractical to solve this. Inspired by the PPO (Schulman et al., 2017) algorithm, we propose a new clipped surrogate objection according to Eqn. (4).

When $k = 1$, for $\hat{L}_0(\pi, \hat{\pi})$, we use the PPO's objective function:

$$\begin{aligned}\hat{L}_0^{\text{clip}}(\pi, \hat{\pi}) =& \mathbb{E}_{(s,a)} \min \left[r(a|s) A^{\hat{\pi}}(s, a)\right., \\ & \left.\text{clip}\left(r(a|s), 1 - \epsilon, 1 + \epsilon\right) A^{\hat{\pi}}(s, a)\right],\end{aligned} \tag{5}$$

Algorithm 1 Reflective Policy Optimization (RPO)

```
Environment $E$, discount factor $\gamma$, batch size $n$, clipping parameter $\epsilon$ and $\epsilon_1$, learning rate $\alpha$ and the weighted parameter $\beta$. Initialize policy network parameter $\theta$.
for $t = 0, 1, 2, \ldots$ do
    Collect data:
    Collect $n$ samples with $\pi_t$ on environment $E$.
    Estimate policy objective:
    Samples a policy data $\pi_t$, estimate on-policy advantage $A^{\pi_t}$ using GAE method, approximately estimate maximize the empirical objective $\hat{L}_0^{\text{clip}}(\pi_\theta, \pi_t)$ and $\hat{L}_1^{\text{clip}}(\pi_\theta, \pi_t)$ from Eqn.(5) and Eqn.(6). The full objective: $\hat{L}(\pi_\theta) \leftarrow \hat{L}_0^{\text{clip}}(\pi_\theta, \pi_t) + \beta \hat{L}_1^{\text{clip}}(\pi_\theta, \pi_t)$.
    Update policy network:
    Update gradient: $\theta \leftarrow \theta + \alpha \nabla_\theta \hat{L}(\pi_\theta)$.
end for
```

where $r(a|s) = \frac{\pi(a|s)}{\hat{\pi}(a|s)}$, ϵ is the hyperparameter, and we ignore the distribution of random variables (s, a).

When $k = 2$, for $\hat{L}_1(\pi, \hat{\pi})$, we simply modify the clipping mechanism:

$$\begin{aligned}\hat{L}_1^{\text{clip}}(\pi, \hat{\pi}) =& \mathbb{E}_{(s,a,s',a')} \min \left[r(a|s) r(a'|s') A^{\hat{\pi}}(s', a') \right. , \\ & \left. C(r, r') A^{\hat{\pi}}(s', a') \right] ,\end{aligned} \tag{6}$$

where $C(r, r') = \text{clip}\left[r(a|s), 1-\epsilon, 1+\epsilon\right] \cdot \text{clip}\left[r(a'|s'),\right.$ $\left.1-\epsilon_1, 1+\epsilon_1\right]$, $r(a|s) = \frac{\pi(a|s)}{\hat{\pi}(a|s)}$, $r(a'|s') = \frac{\pi(a'|s')}{\hat{\pi}(a'|s')}$, ϵ and ϵ_1 are the hyperparameter, we ignore the distribution of random variables (s, a, s', a').

From the Eqn. (6), we implement the clipping mechanism for each ratio, not altogether. If the ratio $r(a|s)$ is large and the ratio $r(a'|s')$ is small, the product of $r(a|s)$ and $r(a'|s')$ may fall between $1-\epsilon$ and $1+\epsilon$. If their product is clipped, the policy will continue to be optimized, and the result may improve or worsen. We have no control over this phenomenon. Therefore, using the separate clipping mechanism will be considered a reasonable situation. The clipping mechanism constrains the ratio variance, making the algorithm's training procedure more stable. In practice, we find that the parameter ϵ_1 cannot be too large, and it's better to be a little smaller than or equal to the ϵ. We want to use the subsequent state-action information to subsidiarily optimize the current policy while avoiding abrupt changes between old and new policies. In this way, the training procedure can be more stable.

Additionally, $k > 2$, the function $\hat{L}_k(\pi, \hat{\pi})$ can be clipped using the same mechanism. Therefore, for the generalized lower bound function, we present a more practical version of the algorithm.

Combining Eqn. (5) and Eqn. (6), we present the Reflective Policy Optimization algorithm (RPO), a practical variant for the generalized surrogate objective function:

$$\hat{L}(\pi, \pi_t) = \hat{L}_0^{\text{clip}}(\pi, \pi_t) + \beta \hat{L}_1^{\text{clip}}(\pi, \pi_t), \tag{7}$$

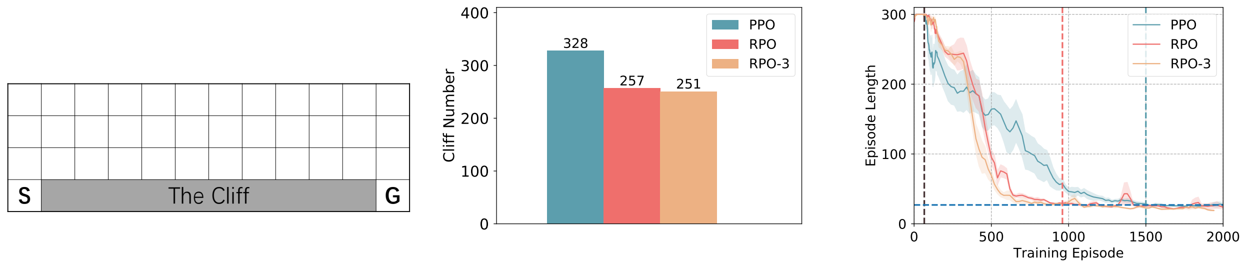
where $\hat{L}_0^{\text{clip}}(\pi, \pi_t)$ is defined in Eqn.(5), $\hat{L}_1^{\text{clip}}(\pi, \pi_t)$ is defined in Eqn.(6), and $\beta > 0$. By choosing the parameter β, this parameter plays a role in weighting the use of subsequent state-action pair information. Eqn. (7) is the optimization objective function for the t-th update. This paper's optimization of the value function is the same as that of PPO. Algorithm 1 shows the detailed implementation pipeline. The RPO algorithm is divided into three steps in each iteration: collect samples, estimate policy objectives, and update the network. It can be seen that our proposed method is also an on-policy algorithm.

Discuss with multi-step RL Multi-step reinforcement learning (RL) is a set of methods that aim to adjust the trade-off of utilization between the knowledge of the current and future return. Recent advances in multi-step RL have demonstrated remarkable empirical success (Wu et al., 2023; Tang et al., 2022). This approach does not directly optimize the current policy but is based on the value function estimated in multiple steps. It is difficult to see directly what role multi-step RL plays in the policy optimization procedure. However, the approach proposed in this paper is viewed from a multi-step perspective: subsequent state-action pairs directly affect policy optimization. It has a direct effect on the actions of the agent and has better theoretical properties. Therefore, our proposed method is fundamentally different from traditional multi-step RL.

Discuss with TayPO The surrogate objective function of the TayPO algorithm (Tang et al., 2020) is denoted as $L_i(\pi, \hat{\pi})$ in Eqn. (2). As discussed in Section 3, their algorithm includes a "1" term, which encounters the same problem. We have observed experimentally (refer to Figure 5 of the appendix) that the "1" term will severely damage the performance of their algorithm. Furthermore, we establish an equality relationship between $\eta(\pi)$ and the generalized surrogate function. Compared to their method, we give a tighter lower bound (please refer to Appendix A.7).

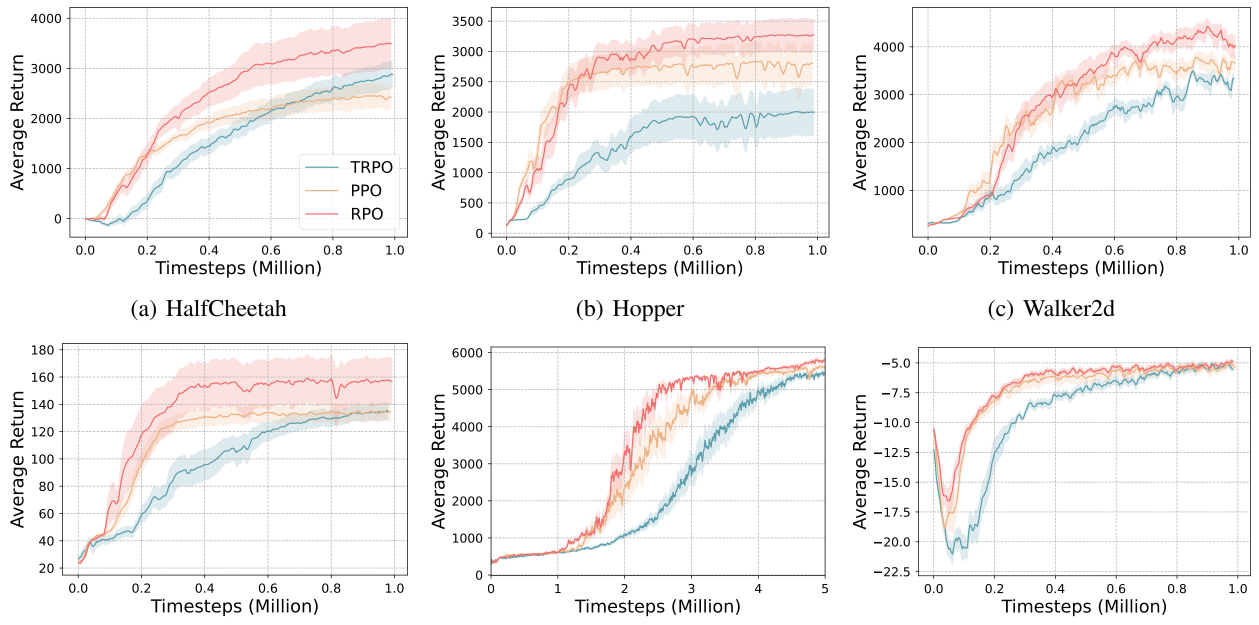
5. Experiments

To verify the effectiveness of the proposed RPO algorithm, we utilize several continuous and discrete environments from the MuJoCo (Todorov et al., 2012) and Atari games in OpenAI Gym (Brockman et al., 2016) extensively adopted in previous works. We conducted experiments with the Mujoco environment based on code from (Queeney et al., 2021) and Atari games based on code from (Zhang, 2018).



(a) CliffWalking (b) Number (c) Length

Figure 1. (a) is a CliffWalking environment. (b) represents the total number of times the agent fell into the Cliff during the training procedure. (c) represents the agent's steps to reach the goal G during the training procedure. RPO-3 means that when $k = 3$, the algorithm uses three ratios.



(a) HalfCheetah (b) Hopper (c) Walker2d

(d) Swimmer (e) Humanoid (f) Reacher

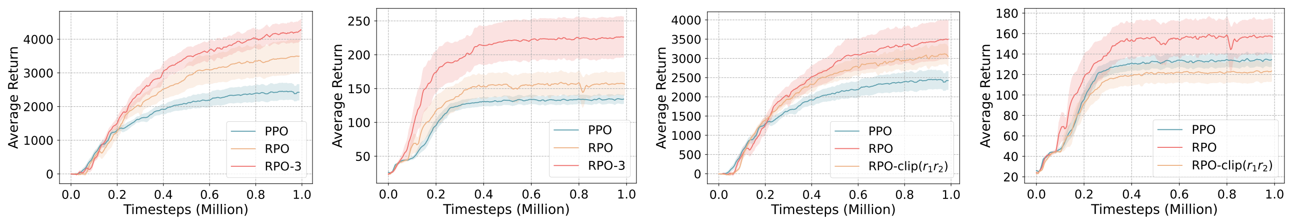
Figure 2. Learning curves on the Gym environments. Performance of RPO vs. PPO. The shaded region indicates the standard deviation of ten random seeds. The X-axis represents the timesteps in the environment. The Y-axis represents the average return.

5.1. Visual Validation Experiment

To demonstrate the effectiveness of the "Reflective Mechanism" of RPO, we conducted visual validation experiments in the CliffWalking environment. CliffWalking is a classic setting widely used for visualizing the performance of reinforcement learning algorithms. From Figure 1 (a), it is characterized by a gird environment in which the agent starts from S and moves through several girds to reach the goal G while avoiding falling into the cliff.

Figure 1 illustrates the overall performance of RPO and its baseline algorithm during the training processing, mainly focusing on the frequency of falling off the cliff and the interaction step overhead, assisting in validating the advantages of RPO's "Reflective Mechanism".

As shown in Figure 1 (b), RPO significantly reduces the frequency of falling off the cliff under equal iteration conditions. This data attests to the significant efficiency of RPO's "Reflective Mechanism". It capitalizes on previous interaction experiences, substantially reducing the occurrence rate of poor decisions. Figure 1 (c) reveals that as the number of interactions increases, RPO markedly cuts down the interaction step overhead per episode, which further confirms the benefits of directly utilizing the subsequent data, *i.e.* the ability to reflect on the action under state and gain the greater rewards. In addition, the selection of k is further discussed. When $k = 3$, using a similar clipping mechanism as in Eqn. (6), we construct the generalized surrogate objective function. From Figure 1, the performance of $k = 3$ is only a little better than $k = 2$. Considering the simplicity and the



(a) HalfCheetah (b) Swimmer (c) HalfCheetah (d) Swimmer

Figure 3. The figure (a) and (b) represent the performance of RPO vs. RPO-3 (means that when $k = 3$, the algorithm uses three ratios), and the figure (c) and (d) represent the performance of RPO vs. RPO-clip($r_1 r_2$) (means that the two ratios are clipped together.).

Table 1. Mean of return of ten random seeds for different methods in Mujoco. The best results are highlighted in **bold**.

	Half-Cheetah	Hopper	Reacher	Walker2d	Swimmer	Humanoid
TRPO	2862	1996	-5.3	3198	135	5425
PPO	2408	2795	-5.2	3643	134	5608
ISPO	1205	1984	-6.7	2275	71	2736
TayPO	-137	29	-66.5	5.7	-22	3717
GePPO	3153	3170	-8.1	3690	**157**	4100
OTRPO	3075	1597	-10.7	2435	122	1768
RPO	**3495**	**3262**	**-4.9**	**4025**	**157**	**5793**

complexity of the policy optimization, the case of $k = 2$ is considered in the main experiment later.

The successful implementation of this RPO mechanism is attributed to its unique approach to comprehensive analysis of state-action pairs. Interestingly, RPO distinguishes itself from most existing algorithms by integrating current and subsequent data strengths. Unlike other algorithms, RPO efficiently utilizes "good" experiences and adjusts based on "bad" experiences. It can more accurately incorporate future states' development, a comprehensive feature that current peer algorithms do not have.

5.2. Main Experimental Analysis

Since the CliffWalking environment is especially conducive to showcasing RPO's "Reflective Mechanism," we performed auxiliary experiments in this setting. In this subsection, we conducted experiments in continuous and discrete action space environments to validate RPO's extensive effectiveness and universal adaptability in reinforcement learning scenarios.

We conducted a detailed comparative analysis with six related algorithms in the field (TRPO (Schulman et al., 2015), PPO (Schulman et al., 2017), GePPO (Queeney et al., 2021), OTRPO (Meng et al., 2022), TayPo (Tang et al., 2020) and ISPO (Tomczak et al., 2019)) in six major experimental environments of MuJoCo (Todorov et al., 2012). The results (as shown in Table 1 and Figure 5 of the appendix) indicate that RPO consistently outperforms in all environments.

In the continuous environments, we use the same hyperparameters $\epsilon_1 = 0.1$ and $\beta = 0.3$ in all environments. From Figure 2 and Figure 5 of the appendix, RPO surpasses classic on-policy reinforcement learning algorithms PPO and TRPO not only in terms of average return but also in convergence speed. This improvement is attributed to RPO's incorporation of current and subsequent data strengths. RPO also exhibits significant advantages compared to the related off-policy algorithms OTRPO and GePPO. This shows that even if off-policy data are not used, better performance can be achieved by using only the data of the current policy itself. The TayPO's performance is not good because this method faces the problems discussed in Section 3.

In the discrete Atari environments, the results are averaged over three seeds during 50M timesteps. We run our experiments across three seeds with fair evaluation metrics. We use the same hyperparameters $\epsilon_1 = 0.1$ and $\beta = 3.0$ in all environments. The normalized improvement (*N.I.*) of the RPO final score a *w.r.t.* the final score of a baseline PPO b is $\frac{a-b}{b-r}$, r is the random score (Vieillard et al., 2020). From Figure 4, we compute the *N.I./PPO* in each environment[1]. It demonstrates that RPO performs better than the PPO algorithm in most environments. We also calculated an average performance improvement of at least 70%. Please refer to the appendix for a detailed curve (refer to Figure 7) and table results (refer to Table 2).

The exceptional performance of RPO is rooted in its unique reflective mechanism that facilitates the efficient utilization of both positive and negative experiences. By employing short trajectories composed of two consecutive states for learning and decision-making, a more profound reflection

[1]For the Venture environment, since the final performance of PPO and random are both zero, whereas our method attains a final performance of 659 points. The calculated $N.I. = \infty$, and we clip the $N.I.$ to 12 for graphical representation.

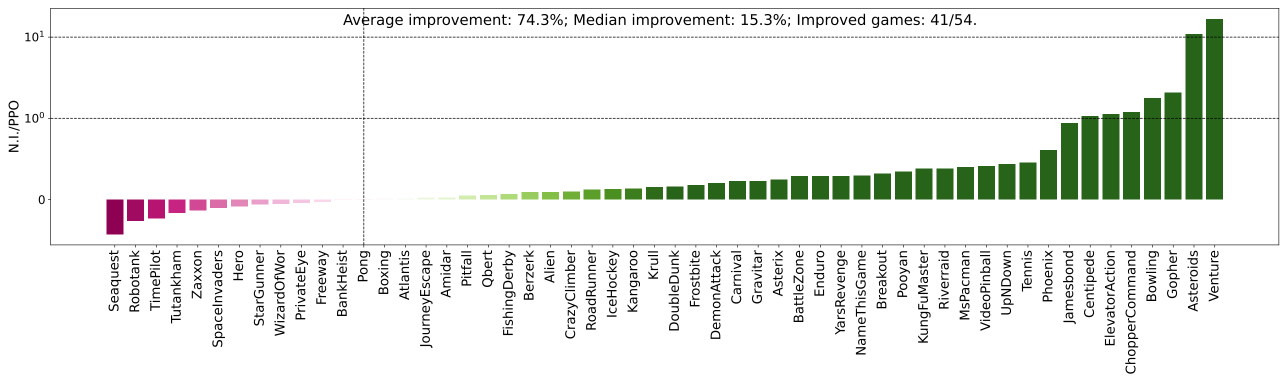


Figure 4. Normalized Improvement of RPO vs. PPO in 54 Atari 2600 games.

and utilization of experience is achieved. This approach has the following benefits: it enables the effective use of interaction experiences from adjacent states. Adopting this pair-wise state combination for short trajectory inputs minimizes the method's computational and storage overhead while maximizing the retention of relevance , promoting a deeper utilization of experience and Reflective mechanism.

The analysis above shows that RPO exhibits significant advantages in various aspects, especially in convergence speed and average return, compared to other algorithms. These empirical findings underscore the efficiency and applicability of the RPO algorithm in both complex continuous and discrete action space environments.

5.3. Ablation Studies

We conduct ablation experiments to evaluate several principal elements that might influence the RPO's performance.

Initially, we conducted an ablation study on the number of states-action pairs, as shown in figures (a) and (b) of Figure 3. When k = 3, we fine-tuned the hyperparameters of the algorithm RPO-3 (three ratios). The results reveal that RPO-3's performance is slightly better than RPO's (two ratios), but the cost of fine-tuning the hyperparameters increases. Considering the number of parameters and the algorithm's simplicity, we think the ratios in RPO need not exceed two, as two states suffice for effective reflection.

Secondly, we conducted ablation experiments on whether the two ratios were clipped together or not, and by comparing it with the RPO-clip($r_1 r_2$) algorithm (this means that the two ratios are clipped together), as shown in figure (c) and (d) of Figure 3. The RPO-clip($r_1 r_2$) may not achieve better performance in some environments, and it may be that two ratios clipped together will face instability. And the RPO method exhibits greater performance, as indicated by the results in Figure 3.

Finally, we conducted ablation experiments on the clipping and weighting coefficients (see Figure 6 in the appendix). Adjusting these two clipping parameters aims to maintain policy stability and address the risk associated with current methods that solely apply clipping to the product. This approach can avoid abrupt changes between old and new policies. We observed that the smaller clipping values yield more significant improvements under a fixed weighting coefficient. Furthermore, our findings suggest that reducing the weighting coefficient alongside certain clipping levels positively influences the results within an acceptable range. It facilitates deep experiential learning from new short trajectories formed by preceding and succeeding states, promoting stable and enhanced performance and accelerating the algorithm's convergence rate.

6. Conclusive Remarks

This paper proposes a simple on-policy algorithm called Reflective Policy Optimization (RPO). This method aims to combine before and after state and action information of the trajectory data to optimize the current policy, thus allowing the agent to reflect on and modify the action of the current state to some extent. Furthermore, theoretical analyses show that our proposed method, in addition to satisfying the desirable property of the monotonic improvement of policy performance, can effectively reduce the optimized policy's solution space, speeding up the algorithm's convergence procedure. We verify the feasibility and effectiveness of the proposed method by a toy example and achieve better performance on RL benchmarks.

In future work, since RPO optimizes the policy by leveraging information from past and future pairs, it is straightforward to integrate it into other Actor-Critic algorithms or maximum entropy methods, further exploiting its advantages. We anticipate that RPO will play a good role in developing new reinforcement learning algorithms.

Acknowledgments

We thank the anonymous ICML reviewers for their insightful and constructive comments. Dr. Junliang Xing was partly supported by the Natural Science Foundation of China under Grant No. 62222606 and 62076238.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Achiam, J., Held, D., Tamar, A., and Abbeel, P. Constrained policy optimization. In *International conference on machine learning*, pp. 22–31, 2017.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. OpenAI Gym. *arXiv preprint arXiv:1606.01540*, 2016.

De Asis, K., Hernandez-Garcia, J., Holland, G., and Sutton, R. Multi-step reinforcement learning: A unifying algorithm. In *AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., Wu, Y., and Zhokhov, P. OpenAI baselines. https://github.com/openai/baselines, 2017.

Duan, Y. and Wainwright, M. J. A finite-sample analysis of multi-step temporal difference estimates. In *Learning for Dynamics and Control Conference*, pp. 612–624, 2023.

Finner, H. A generalization of holder's inequality and some probability inequalities. *The Annals of Probability*, pp. 1893–1901, 1992.

Fujimoto, S., van Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pp. 1582–1591, 2018.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pp. 1856–1865, 2018.

Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., and Meger, D. Deep reinforcement learning that matters. In *AAAI conference on artificial intelligence*, pp. 3207–3214, 2018.

Hernandez-Garcia, J. F. and Sutton, R. S. Understanding multi-step deep reinforcement learning: A systematic study of the dqn target. *arXiv preprint arXiv:1901.07510*, 2019.

Kakade, S. M. and Langford, J. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning*, pp. 267–274, 2002.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*, 2016.

Meng, W., Zheng, Q., Shi, Y., and Pan, G. An off-policy trust region policy optimization method with monotonic improvement guarantee for deep reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 33(5):2223–2235, 2022.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M. A., Fidjeland, A., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. Human-level control through deep reinforcement learning. volume 518, pp. 529–533, 2015.

Munos, R., Stepleton, T., Harutyunyan, A., and Bellemare, M. G. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 1046–1054, 2016.

Queeney, J., Paschalidis, Y., and Cassandras, C. G. Generalized proximal policy optimization with sample reuse. In *Advances in Neural Information Processing Systems*, pp. 11909–11919, 2021.

Schulman, J., Levine, S., Abbeel, P., Jordan, M. I., and Moritz, P. Trust region policy optimization. In *International Conference on Machine Learning*, pp. 1889–1897, 2015.

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

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. A. Deterministic policy gradient algorithms. In *International Conference on Machine Learning*, pp. 387–395, 2014.

Sutton, R. S. and Barto, A. G. Reinforcement learning: an introduction. 1998.

Tang, Y., Valko, M., and Munos, R. Taylor expansion policy optimization. In *International Conference on Machine Learning*, pp. 9397–9406, 2020.

Tang, Y., Munos, R., Rowland, M., Pires, B. Á., Dabney, W., and Bellemare, M. G. The nature of temporal difference errors in multi-step distributional reinforcement learning. In *Advances in Neural Information Processing Systems*, 2022.

Todorov, E., Erez, T., and Tassa, Y. MuJoCo: A physics engine for model-based control. In *International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012.

Tomczak, M. B., Kim, D., Vrancx, P., and Kim, K.-E. Policy optimization through approximate importance sampling. *arXiv preprint arXiv:1910.03857*, 2019.

van Hasselt, H., Guez, A., and Silver, D. Deep reinforcement learning with double q-learning. In *AAAI Conference on Artificial Intelligence*, pp. 2094–2100, 2016.

Vieillard, N., Pietquin, O., and Geist, M. Munchausen reinforcement learning. In *Advances in Neural Information Processing Systems*, 2020.

Wu, Z., Yu, C., Chen, C., Hao, J., and Zhuo, H. H. Models as agents: Optimizing multi-step predictions of interactive local models in model-based multi-agent reinforcement learning. In *AAAI Conference on Artificial Intelligence*, pp. 10435–10443, 2023.

Zhang, S. Modularized implementation of deep rl algorithms in pytorch. https://github.com/ShangtongZhang/DeepRL, 2018.

A. Proof

Let's start with some useful lemmas.

Lemma A.1. *(Kakade & Langford, 2002) Consider any two policies $\hat{\pi}$ and π, we have*

$$\eta(\pi) - \eta(\hat{\pi}) = \frac{1}{1-\gamma}\mathbb{E}_{s,a\sim\rho^{\pi}} A^{\hat{\pi}}(s,a).$$

Corollary A.2. *Consider any two policies $\hat{\pi}$ and π, we have*

- $V^{\pi}(s_0) - V^{\hat{\pi}}(s_0) = \frac{1}{1-\gamma}\mathbb{E}_{s,a\sim\rho^{\pi}(\cdot|s_0)} A^{\hat{\pi}}(s,a).$
- $Q^{\pi}(s_0,a_0) - Q^{\hat{\pi}}(s_0,a_0) = \frac{\gamma}{1-\gamma}\mathbb{E}_{s,a\sim\rho^{\pi}(\cdot|s_0,a_0)} A^{\hat{\pi}}(s,a).$

Proof. The first formula is simple, due to $\eta(\pi) = \mathbb{E}_{s_0\sim\rho_0} V^{\pi}(s_0)$.

Let's prove the second formula.

$$\begin{aligned}
&Q^{\pi}(s_0,a_0) - Q^{\hat{\pi}}(s_0,a_0)\\
=&\gamma\mathbb{E}_{s'\sim P(s'|s_0,a_0)}\left[V^{\pi}(s') - V^{\hat{\pi}}(s')\right]\\
=&\frac{\gamma}{1-\gamma}\mathbb{E}_{s'\sim P(s'|s_0,a_0)}\mathbb{E}_{s,a\sim\rho^{\pi}(\cdot|s')} A^{\hat{\pi}}(s,a)\\
=&\frac{\gamma}{1-\gamma}\mathbb{E}_{s,a\sim\rho^{\pi}(\cdot|s_0,a_0)} A^{\hat{\pi}}(s,a).
\end{aligned}$$

□

Lemma A.3. *(Tomczak et al., 2019) Consider any two policies $\hat{\pi}$ and π, we have*

$$\eta(\pi) - \eta(\hat{\pi}) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim\rho^{\hat{\pi}},a\sim\pi} A^{\hat{\pi}}(s,a) + \frac{1}{1-\gamma}\mathbb{E}_{s,a\sim\rho^{\hat{\pi}}}\left[\frac{\pi(a|s)}{\hat{\pi}(a|s)} - 1\right]\left[Q^{\pi}(s,a) - Q^{\hat{\pi}}(s,a)\right].$$

Lemma A.4. *Consider a current policy $\hat{\pi}$, and any policies π, we have*

$$\begin{aligned}
&\mathbb{E}_{s,a\sim\rho^{\pi}(\cdot)} A^{\hat{\pi}}(s,a) - \mathbb{E}_{s\sim\rho^{\hat{\pi}},a\sim\pi} A^{\hat{\pi}}(s,a)\\
=&\frac{\gamma}{1-\gamma}\mathop{\mathbb{E}}_{\substack{s,a\sim\rho^{\hat{\pi}}(\cdot)\\ s',a'\sim\rho^{\pi}(\cdot|s,a)}}[\frac{\pi(a|s)}{\hat{\pi}(a|s)} - 1]A^{\hat{\pi}}(s',a').
\end{aligned}$$

Proof. From Lemma A.1 and A.3, we have

$$\begin{aligned}
&\mathbb{E}_{s,a\sim\rho^{\pi}} A^{\hat{\pi}}(s,a) - \mathbb{E}_{s\sim\rho^{\hat{\pi}},a\sim\pi} A^{\hat{\pi}}(s,a)\\
=&\mathbb{E}_{s,a\sim\rho^{\hat{\pi}}}\left[\frac{\pi(a|s)}{\hat{\pi}(a|s)} - 1\right]\left[Q^{\pi}(s,a) - Q^{\hat{\pi}}(s,a)\right].
\end{aligned}$$

According to Corollary A.2, it is easy to get the conclusion. □

Theorem A.5. *Consider a current policy $\hat{\pi}$, and any policies π, we have*

$$\eta(\pi) = \eta(\hat{\pi}) + \sum_{i=0}^{k-1}\alpha_i L_i(\pi,\hat{\pi}) + \beta_k G_k(\pi,\hat{\pi}),$$

where

$$
\begin{aligned}
L_i(\pi,\hat{\pi}) &= \mathop{\mathbb{E}}_{\substack{s_0,a_0\sim\rho^{\hat{\pi}}(\cdot)\\ \cdots \\ s_{i-1},a_{i-1}\sim\rho^{\hat{\pi}}(\cdot|s_{i-2},a_{i-2})}} \prod_{t=0}^{i-1}(r_t-1)\cdot l_i(\pi,\hat{\pi}),\\
G_k(\pi,\hat{\pi}) &= \mathop{\mathbb{E}}_{\substack{s_0,a_0\sim\rho^{\hat{\pi}}(\cdot)\\ \cdots \\ s_{k-1},a_{k-1}\sim\rho^{\hat{\pi}}(\cdot|s_{k-2},a_{k-2})}} \prod_{t=0}^{k-1}(r_t-1)\cdot g_k(\pi,\hat{\pi}),\\
l_i(\pi,\hat{\pi}) &= \mathbb{E}_{s_i\sim\rho^{\hat{\pi}}(\cdot|s_{i-1},a_{i-1}),a_i\sim\pi(\cdot|s_i)}A^{\hat{\pi}}(s_i,a_i),\\
g_k(\pi,\hat{\pi}) &= \mathbb{E}_{s_k,a_k\sim\rho^{\pi}(\cdot|s_{k-1},a_{k-1})}A^{\hat{\pi}}(s_k,a_k),
\end{aligned}
$$

and

$$
r_t=\frac{\pi(a_t|s_t)}{\hat{\pi}(a_t|s_t)},\ \alpha_i=\frac{\gamma^i}{(1-\gamma)^{i+1}},\ \beta_k=\frac{\gamma^k}{(1-\gamma)^{k+1}}.
$$

Proof. From Lemma A.4, this formula creates a link between $\mathbb{E}_{s,a\sim\rho^{\pi}(\cdot)}A^{\hat{\pi}}(s,a)$ and $\mathbb{E}_{s',a'\sim\rho^{\pi}(\cdot|s,a)}A^{\hat{\pi}}(s',a')$, resulting in a recursive relationship.

According to Lemma A.3, and using recursive relationships, defined

$$
l_i(\pi,\hat{\pi}) = \mathbb{E}_{s_i\sim\rho^{\hat{\pi}}(\cdot|s_{i-1},a_{i-1}),a_i\sim\pi(\cdot|s_i)}A^{\hat{\pi}}(s_i,a_i),
$$

we have

$$
\begin{aligned}
&\eta(\pi)-\eta(\hat{\pi})\\
=&\frac{1}{1-\gamma}\mathbb{E}_{s_0\sim\rho^{\hat{\pi}},a_0\sim\pi}A^{\hat{\pi}}(s_0,a_0)+\frac{\gamma}{(1-\gamma)^2}\mathbb{E}_{s_0,a_0\sim\rho^{\hat{\pi}}}[r_0-1]\mathbb{E}_{s_1,a_1\sim\rho^{\pi}(\cdot|s_0,a_0)}A^{\hat{\pi}}(s_1,a_1)\\
=&\frac{1}{1-\gamma}l_0(\pi,\hat{\pi})\\
&+\frac{\gamma}{(1-\gamma)^2}\mathbb{E}_{s_0,a_0\sim\rho^{\hat{\pi}}}[r_0-1]\left(l_1(\pi,\hat{\pi})+\frac{\gamma}{1-\gamma}\mathbb{E}_{s_1,a_1\sim\rho^{\hat{\pi}}(\cdot|s_0,a_0)}[r_1-1]\mathbb{E}_{s_2,a_2\sim\rho^{\pi}(\cdot|s_1,a_1)}A^{\hat{\pi}}(s_2,a_2)\right)\\
=&\frac{1}{1-\gamma}l_0(\pi,\hat{\pi})+\frac{\gamma}{(1-\gamma)^2}\mathbb{E}_{s_0,a_0\sim\rho^{\hat{\pi}}}[r_0-1]l_1(\pi,\hat{\pi})\\
&+\frac{\gamma^2}{(1-\gamma)^3}\mathop{\mathbb{E}}_{\substack{s_0,a_0\sim\rho^{\hat{\pi}}(\cdot)\\ s_1,a_1\sim\rho^{\hat{\pi}}(\cdot|s_0,a_0)\\ s_2,a_2\sim\rho^{\pi}(\cdot|s_1,a_1)}}[r_0-1][r_1-1]A^{\hat{\pi}}(s_2,a_2)\\
&\cdots\\
=&\sum_{i=0}^{k-1}\alpha_i L_i(\pi,\hat{\pi})+\beta_k G_k(\pi,\hat{\pi}).
\end{aligned}
$$

□

Corollary A.6. *According to the definition of G_k, we have*

$$
|\beta_k G_k(\pi,\hat{\pi})|\le\frac{\gamma^k}{(1-\gamma)^{k+2}}\epsilon^{k+1}R_{\max},
$$

where $\epsilon\triangleq\|\pi-\hat{\pi}\|_1=\max_s\sum_a|\pi(a|s)-\hat{\pi}(a|s)|$ and $R_{\max}\triangleq\max_{s,a}|R(s,a)|$.

Proof. According to the definition of $G_k(\pi, \hat{\pi})$, and defined $\epsilon \triangleq \|\pi - \hat{\pi}\|_1$, we have

$$\begin{aligned} |G_k(\pi, \hat{\pi})| &\le \epsilon^k \cdot |\mathbb{E}_{s_k, a_k \sim \rho^{\pi}(\cdot|s_{k-1}, a_{k-1})} A^{\hat{\pi}}(s_k, a_k)| \\ &\le \epsilon^k \cdot | \int_a (\pi - \hat{\pi}) Q^{\hat{\pi}}(s, a) da| \\ &\le \frac{R_{\max}}{1-\gamma} \epsilon^{k+1}. \end{aligned}$$

Combining with β_k, we can get this conclusion. □

Corollary A.7. *Compared with Theorem 2 of the paper (Tang et al., 2020), we give a tighter lower bound.*

Proof. From the paper (Tang et al., 2020), they give the gap between the policy performance of π and the general surrogate object

$$\hat{G}_k = \frac{1}{\gamma(1-\gamma)} \left(1 - \frac{\gamma}{1-\gamma}\epsilon\right)^{-1} \left(\frac{\gamma\epsilon}{1-\gamma}\right)^{K+1} R_{\max}.$$

Next, from Corollary A.6, we will prove that the following inequality holds

$$\frac{\gamma^k}{(1-\gamma)^{k+2}} \epsilon^{k+1} R_{\max} < \hat{G}_k.$$

That is, we need to prove

$$\frac{\gamma^k}{(1-\gamma)^{k+2}} \epsilon^{k+1} R_{\max} < \frac{1}{\gamma(1-\gamma)} \left(1 - \frac{\gamma}{1-\gamma}\epsilon\right)^{-1} \left(\frac{\gamma\epsilon}{1-\gamma}\right)^{K+1} R_{\max}.$$

After simplification, we get

$$\frac{1}{1-\gamma} < \frac{1}{1-\gamma-\gamma\epsilon}.$$

The inequality obviously holds. So, we give a tighter lower bound. □

Theorem A.8. *Consider a current policy $\hat{\pi}$, and any policies π, we have*

$$\eta(\pi) - \eta(\hat{\pi}) \ge \sum_{i=0}^{k-1} \alpha_i \hat{L}_i(\pi, \hat{\pi}) - \hat{C}_k(\pi, \hat{\pi}),$$

where

$$\hat{L}_i(\pi, \hat{\pi}) = \mathop{\mathbb{E}}_{\substack{s_0, a_0 \sim \rho^{\hat{\pi}}(\cdot) \\ \cdots \\ s_{i-1}, a_{i-1} \sim \rho^{\hat{\pi}}(\cdot|s_{i-2}, a_{i-2}) \\ s_i, a_i \sim \rho^{\hat{\pi}}(\cdot|s_{i-1}, a_{i-1})}} \prod_{t=0}^{i} r_t \cdot A^{\hat{\pi}}(s_i, a_i),$$

$$\hat{C}_k(\pi, \hat{\pi}) = \frac{\gamma R_{\max} \|\pi - \hat{\pi}\|_1}{1-\gamma} \sum_{i=1}^{k-1} \alpha_i + \frac{\gamma^k R_{\max}}{(1-\gamma)^{k+2}} \|\pi - \hat{\pi}\|_1^2,$$

and $\alpha_i = \frac{\gamma^i}{(1-\gamma)^{i+1}}$.

Proof. For the definition of $L_i(\pi, \hat{\pi})$, we have

$$\begin{aligned}
&\eta(\pi) - \eta(\hat{\pi}) \\
=&\frac{1}{1-\gamma}\mathbb{E}_{s_0\sim\rho^{\hat{\pi}},a_0\sim\pi}A^{\hat{\pi}}(s_0,a_0) + \frac{\gamma}{(1-\gamma)^2}\mathbb{E}_{s_0,a_0\sim\rho^{\hat{\pi}}}[r_0-1]\mathbb{E}_{s_1,a_1\sim\rho^{\pi}(\cdot|s_0,a_0)}A^{\hat{\pi}}(s_1,a_1) \\
=&\frac{1}{1-\gamma}l_0(\pi,\hat{\pi}) - \frac{\gamma}{(1-\gamma)^2}\mathbb{E}_{s_0,a_0\sim\rho^{\hat{\pi}}}\mathbb{E}_{s_1,a_1\sim\rho^{\hat{\pi}}(\cdot|s_0,a_0)}A^{\hat{\pi}}(s_1,a_1) \\
&+\frac{\gamma}{(1-\gamma)^2}\mathbb{E}_{s_0,a_0\sim\rho^{\hat{\pi}}}r_0\left(l_1(\pi,\hat{\pi}) + \frac{\gamma}{1-\gamma}\mathbb{E}_{s_1,a_1\sim\rho^{\hat{\pi}}(\cdot|s_0,a_0)}[r_1-1]\mathbb{E}_{s_2,a_2\sim\rho^{\pi}(\cdot|s_1,a_1)}A^{\hat{\pi}}(s_2,a_2)\right) \\
=&\frac{1}{1-\gamma}l_0(\pi,\hat{\pi}) - \frac{\gamma}{(1-\gamma)^2}\mathbb{E}_{s_0,a_0\sim\rho^{\hat{\pi}}}\mathbb{E}_{s_1,a_1\sim\rho^{\hat{\pi}}(\cdot|s_0,a_0)}A^{\hat{\pi}}(s_1,a_1) \\
&+\frac{\gamma}{(1-\gamma)^2}\mathbb{E}_{s_0,a_0\sim\rho^{\hat{\pi}}}r_0 l_1(\pi,\hat{\pi}) - \frac{\gamma^2}{(1-\gamma)^3}\mathbb{E}_{s_0,a_0\sim\rho^{\hat{\pi}}}r_0\mathbb{E}_{s_1,a_1\sim\rho^{\hat{\pi}}(\cdot|s_0,a_0)}\mathbb{E}_{s_2,a_2\sim\rho^{\pi}(\cdot|s_1,a_1)}A^{\hat{\pi}}(s_2,a_2) \\
&+\frac{\gamma^2}{(1-\gamma)^3}\mathbb{E}_{s_0,a_0\sim\rho^{\hat{\pi}}}r_0\mathbb{E}_{s_1,a_1\sim\rho^{\hat{\pi}}(\cdot|s_0,a_0)}r_1\mathbb{E}_{s_2,a_2\sim\rho^{\pi}(\cdot|s_1,a_1)}A^{\hat{\pi}}(s_2,a_2) \\
=&\frac{1}{1-\gamma}l_0(\pi,\hat{\pi}) + \frac{\gamma}{(1-\gamma)^2}\mathbb{E}_{s_0,a_0\sim\rho^{\hat{\pi}}}[r_0-1]l_1(\pi,\hat{\pi}) \\
&+\frac{\gamma^2}{(1-\gamma)^3}\mathop{\mathbb{E}}_{\substack{s_0,a_0\sim\rho^{\hat{\pi}}(\cdot)\\ s_1,a_1\sim\rho^{\hat{\pi}}(\cdot|s_0,a_0)\\ s_2,a_2\sim\rho^{\pi}(\cdot|s_1,a_1)}}[r_0-1][r_1-1]A^{\hat{\pi}}(s_2,a_2) \\
&\dots \\
=&\sum_{i=0}^{k-1}\alpha_i\hat{L}_i(\pi,\hat{\pi}) - \sum_{i=1}^{k-1}\alpha_i\hat{H}_i(\pi,\hat{\pi}) + \beta_k\hat{G}_k(\pi,\hat{\pi}),
\end{aligned}$$

where

$$\hat{L}_i(\pi,\hat{\pi}) = \mathop{\mathbb{E}}_{\substack{s_0,a_0\sim\rho^{\hat{\pi}}(\cdot)\\ \dots\\ s_{i-1},a_{i-1}\sim\rho^{\hat{\pi}}(\cdot|s_{i-2},a_{i-2})\\ s_i,a_i\sim\rho^{\pi}(\cdot|s_{i-1},a_{i-1})}}\prod_{t=0}^{i} r_t\cdot A^{\hat{\pi}}(s_i,a_i),$$

$$\hat{H}_i(\pi,\hat{\pi}) = \mathop{\mathbb{E}}_{\substack{s_0,a_0\sim\rho^{\hat{\pi}}(\cdot)\\ \dots\\ s_{i-1},a_{i-1}\sim\rho^{\hat{\pi}}(\cdot|s_{i-2},a_{i-2})\\ s_i,a_i\sim\rho^{\pi}(\cdot|s_{i-1},a_{i-1})}}\prod_{t=0}^{i-2} r_t\cdot A^{\hat{\pi}}(s_i,a_i),$$

$$\hat{G}_k(\pi,\hat{\pi}) = \mathop{\mathbb{E}}_{\substack{s_0,a_0\sim\rho^{\hat{\pi}}(\cdot)\\ \dots\\ s_{i-1},a_{i-1}\sim\rho^{\hat{\pi}}(\cdot|s_{i-2},a_{i-2})\\ s_i,a_i\sim\rho^{\pi}(\cdot|s_{i-1},a_{i-1})}}\prod_{t=0}^{i-2} r_t\cdot(r_{i-1}-1)\cdot A^{\hat{\pi}}(s_i,a_i),$$

and $\alpha_i = \frac{\gamma^i}{(1-\gamma)^{i+1}},\ \beta_k = \frac{\gamma^k}{(1-\gamma)^{k+1}}$.

It is easy to prove that the following inequality holds

$$\hat{H}_i(\pi,\hat{\pi}) \le \frac{R_{\max}}{1-\gamma}\|\pi-\hat{\pi}\|_1,\ \hat{G}_k(\pi,\hat{\pi}) \le \frac{R_{\max}}{1-\gamma}\|\pi-\hat{\pi}\|_1^2.$$

We have

$$\eta(\pi)-\eta(\hat{\pi}) \ge \sum_{i=0}^{k-1}\alpha_i\hat{L}_i(\pi,\hat{\pi}) - \frac{\gamma R_{\max}\|\pi-\hat{\pi}\|_1}{1-\gamma}\sum_{i=1}^{k-1}\alpha_i - \frac{\gamma^k R_{\max}}{(1-\gamma)^{k+2}}\|\pi-\hat{\pi}\|_1^2.$$

□

Theorem A.9. *Define two sets*

$$\begin{aligned} \Psi_1 &= \left\{ \mu \mid \alpha_0 \hat{L}_0(\mu, \hat{\pi}) - \hat{C}_1(\mu, \hat{\pi}) \geq 0, \|\mu - \hat{\pi}\|_1 \leq \frac{1}{2} \right\}, \\ \Psi_2 &= \left\{ \mu \mid \alpha_0 \hat{L}_0(\mu, \hat{\pi}) + \alpha_1 \hat{L}_1(\mu, \hat{\pi}) - \hat{C}_2(\mu, \hat{\pi}) \geq 0, \|\mu - \hat{\pi}\|_1 \leq \frac{1}{2} \right\}, \end{aligned}$$

then we have

$$\Psi_2 \subseteq \Psi_1.$$

Proof. Let $\mu \in \Psi_1$, we have

$$\hat{L}_0(\mu, \hat{\pi}) - \frac{\gamma R_{\max}}{(1-\gamma)^2} \|\mu - \hat{\pi}\|_1^2 \geq 0. \tag{8}$$

Below, we will show that μ may not be in the set Ψ_2.

For $\hat{L}_1(\pi, \hat{\pi})$, we can get

$$\begin{aligned} \hat{L}_1(\pi, \hat{\pi}) = & \mathop{\mathbb{E}}_{\substack{s_0 \sim \rho^{\hat{\pi}}(\cdot), a_0 \sim \pi(\cdot|s_0) \\ s_1 \sim \rho^{\hat{\pi}}(\cdot|s_0, a_0), a_1 \sim \pi(\cdot|s_1)}} A^{\hat{\pi}}(s_1, a_1) && (9) \\ = & \mathop{\mathbb{E}}_{\substack{s_0 \sim \rho^{\hat{\pi}}(\cdot), a_0 \sim \hat{\pi}(\cdot|s_0) \\ s_1 \sim \rho^{\hat{\pi}}(\cdot|s_0, a_0), a_1 \sim \pi(\cdot|s_1)}} A^{\hat{\pi}}(s_1, a_1) + \left(\mathop{\mathbb{E}}_{\substack{s_0 \sim \rho^{\hat{\pi}}(\cdot), a_0 \sim \pi(\cdot|s_0) \\ s_1 \sim \rho^{\hat{\pi}}(\cdot|s_0, a_0), a_1 \sim \pi(\cdot|s_1)}} - \mathop{\mathbb{E}}_{\substack{s_0 \sim \rho^{\hat{\pi}}(\cdot), a_0 \sim \hat{\pi}(\cdot|s_0) \\ s_1 \sim \rho^{\hat{\pi}}(\cdot|s_0, a_0), a_1 \sim \pi(\cdot|s_1)}} \right) A^{\hat{\pi}}(s_1, a_1) && (10) \\ \geq & \mathop{\mathbb{E}}_{s_1 \sim \rho^{\hat{\pi}}(\cdot), a_1 \sim \pi(\cdot|s_1)} A^{\hat{\pi}}(s_1, a_1) - \frac{R_{\max}}{1-\gamma} \|\pi - \hat{\pi}\|_1^2. && (11) \end{aligned}$$

The last inequality uses $\mathbb{E}_{s_0 \sim \rho^{\hat{\pi}}(\cdot), a_0 \sim \hat{\pi}(\cdot|s_0)} \rho^{\hat{\pi}}(\cdot|s_0, a_0) = \rho^{\hat{\pi}}(\cdot)$ and Hölder's inequality (Finner, 1992).

Combining with $\hat{L}_0(\pi, \hat{\pi})$ and $\hat{C}_2(\pi, \hat{\pi})$, we have

$$\begin{aligned} & \hat{L}_0(\pi, \hat{\pi}) + \frac{\gamma}{1-\gamma} \hat{L}_1(\pi, \hat{\pi}) - \frac{\gamma R_{\max}}{(1-\gamma)^2} \|\pi - \hat{\pi}\|_1 - \frac{\gamma^2 R_{\max}}{(1-\gamma)^3} \|\pi - \hat{\pi}\|_1^2 \\ \geq & \hat{L}_0(\pi, \hat{\pi}) + \frac{\gamma}{1-\gamma} \left(\mathop{\mathbb{E}}_{s_1 \sim \rho^{\hat{\pi}}(\cdot), a_1 \sim \pi(\cdot|s_1)} A^{\hat{\pi}}(s_1, a_1) - \frac{R_{\max}}{1-\gamma} \|\pi - \hat{\pi}\|_1^2 \right) - \frac{\gamma R_{\max}}{(1-\gamma)^2} \|\pi - \hat{\pi}\|_1 - \frac{\gamma^2 R_{\max}}{(1-\gamma)^3} \|\pi - \hat{\pi}\|_1^2 \\ = & \frac{1}{1-\gamma} \hat{L}_0(\pi, \hat{\pi}) - \frac{\gamma R_{\max}}{(1-\gamma)^3} \|\pi - \hat{\pi}\|_1^2 - \frac{\gamma R_{\max}}{(1-\gamma)^2} \|\pi - \hat{\pi}\|_1. \end{aligned}$$

Combining with the inequality (8), we have

$$\hat{L}_0(\mu, \hat{\pi}) + \frac{\gamma}{1-\gamma} \hat{L}_1(\mu, \hat{\pi}) - \frac{\gamma R_{\max}}{(1-\gamma)^2} \|\mu - \hat{\pi}\|_1 - \frac{\gamma^2 R_{\max}}{(1-\gamma)^3} \|\mu - \hat{\pi}\|_1^2 \geq - \frac{\gamma R_{\max}}{(1-\gamma)^2} \|\mu - \hat{\pi}\|_1.$$

From the above inequality, it shows that μ may not be in set Ψ_2.

On the contrary, we will show that if $\mu \in \Psi_2$, then $\mu \in \Psi_1$.
According to the Eqn. (9),we can get the upper bound of $\hat{L}_1(\pi, \hat{\pi})$:

$$\hat{L}_1(\pi, \hat{\pi}) \leq \mathop{\mathbb{E}}_{s_1 \sim \rho^{\hat{\pi}}(\cdot), a_1 \sim \pi(\cdot|s_1)} A^{\hat{\pi}}(s_1, a_1) + \frac{R_{\max}}{1-\gamma} \|\pi - \hat{\pi}\|_1^2.$$

If $\mu \in \Psi_2$, we have

$$\begin{aligned}
0 \leq & \hat{L}_0(\mu, \hat{\pi}) + \frac{\gamma}{1-\gamma} \hat{L}_1(\mu, \hat{\pi}) - \frac{\gamma R_{\max}}{(1-\gamma)^2} \|\mu - \hat{\pi}\|_1 - \frac{\gamma^2 R_{\max}}{(1-\gamma)^3} \|\mu - \hat{\pi}\|_1^2 \\
\leq & \hat{L}_0(\mu, \hat{\pi}) + \frac{\gamma}{1-\gamma} \left(\mathop{\mathbb{E}}_{s_1 \sim \rho^{\hat{\pi}}(\cdot), a_1 \sim \mu(\cdot|s_1)} A^{\hat{\pi}}(s_1, a_1) + \frac{R_{\max}}{1-\gamma} \|\mu - \hat{\pi}\|_1^2 \right) \\
& - \frac{\gamma R_{\max}}{(1-\gamma)^2} \|\mu - \hat{\pi}\|_1 - \frac{\gamma^2 R_{\max}}{(1-\gamma)^3} \|\mu - \hat{\pi}\|_1^2 \\
= & \frac{1}{1-\gamma} \hat{L}_0(\mu, \hat{\pi}) - \frac{\gamma R_{\max}}{(1-\gamma)^2} \|\mu - \hat{\pi}\|_1 + \frac{\gamma(1-2\gamma) R_{\max}}{(1-\gamma)^3} \|\mu - \hat{\pi}\|_1^2 \\
= & \frac{1}{1-\gamma} \hat{L}_0(\mu, \hat{\pi}) - \frac{\gamma R_{\max}}{(1-\gamma)^3} \|\mu - \hat{\pi}\|_1^2 \\
& + \frac{\gamma R_{\max}}{(1-\gamma)^3} \|\mu - \hat{\pi}\|_1^2 - \frac{\gamma R_{\max}}{(1-\gamma)^2} \|\mu - \hat{\pi}\|_1 + \frac{\gamma(1-2\gamma) R_{\max}}{(1-\gamma)^3} \|\mu - \hat{\pi}\|_1^2.
\end{aligned} \tag{12}$$

Next, we will show that the second line of the last equation in the above formula is less than or equal to 0. We have

$$\begin{aligned}
& \frac{\gamma R_{\max}}{(1-\gamma)^3} \|\mu - \hat{\pi}\|_1^2 - \frac{\gamma R_{\max}}{(1-\gamma)^2} \|\mu - \hat{\pi}\|_1 + \frac{\gamma(1-2\gamma) R_{\max}}{(1-\gamma)^3} \|\mu - \hat{\pi}\|_1^2 \\
= & \frac{\gamma R_{\max}}{(1-\gamma)^2} \left(\frac{1}{1-\gamma} \|\mu - \hat{\pi}\|_1^2 - \|\mu - \hat{\pi}\|_1 + \frac{1-2\gamma}{1-\gamma} \|\mu - \hat{\pi}\|_1^2 \right) \\
= & \frac{\gamma R_{\max}}{(1-\gamma)^2} \left(2\|\mu - \hat{\pi}\|_1^2 - \|\mu - \hat{\pi}\|_1 \right) \\
\leq & 0.
\end{aligned}$$

The last inequality holds because $\|\mu - \hat{\pi}\|_1 \leq \frac{1}{2}$. So, from the Eqn.(12), we have

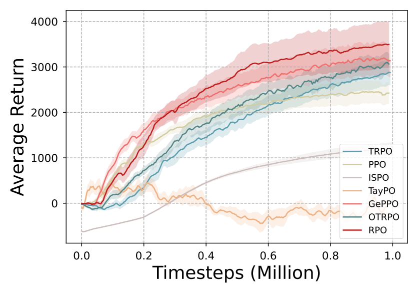
$$0 \leq \frac{1}{1-\gamma} \hat{L}_0(\mu, \hat{\pi}) - \frac{\gamma R_{\max}}{(1-\gamma)^3} \|\mu - \hat{\pi}\|_1^2.$$

In summary, we have that $\Psi_2 \subseteq \Psi_1$ holds. □

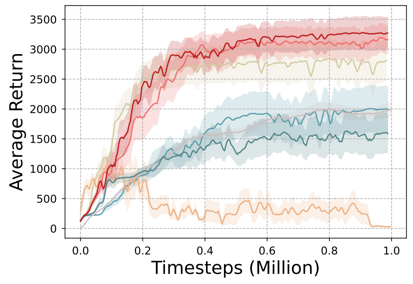
B. Additional experimental results

To verify the effectiveness of the proposed RPO method, we select six continuous control tasks from the MuJoCo environments (Todorov et al., 2012) in OpenAI Gym (Brockman et al., 2016). We conduct all the experiments mainly based on the code from (Queeney et al., 2021). The test procedures are averaged over ten test episodes across ten independent runs. For the experimental parameters, we use the default parameters from (Dhariwal et al., 2017; Henderson et al., 2018), for example, the discount factor is $\gamma = 0.995$, and we use the Adam optimizer (Kingma & Ba, 2015) throughout the training progress. The learning rate $\phi = 3e-4$ except for Humanoid which is $1e-5$. For RPO, the clipping parameters are $\epsilon = 0.2$ and $\epsilon_1 = 0.1$, and the weighted parameter $\beta = 0.3$ on MuJoCo environments and do not fine-tune them. The RPO algorithm involves hyperparameters, but we have yet to extensively fine-tun them, from Table 3, we supplemented some experiments for $k = 4$.

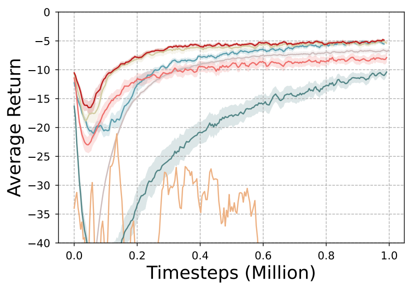
To verify the effectiveness of the proposed RPO method in discrete Atari environments, the code is based on (Zhang, 2018). We run our experiments across three seeds with fair evaluation metrics. We use the same hyperparameters $\epsilon_1 = 0.1$ and $\beta = 3.0$ in all environments and do not fine-tune them. Firstly, in the discrete state-action space of Atari, rewards are sparser compared to the continuous state-action space of MuJoCo. Therefore, to better utilize information from subsequent states, β should be larger than the setting in MuJoCo. Secondly, we did not specifically select the hyperparameters but set them arbitrarily. Supplementary experiments were conducted on six randomly selected Atari games to demonstrate this. In Table 4, experimental results show that the performance of the RPO algorithm with different parameters is better than that of PPO, demonstrating better robustness.



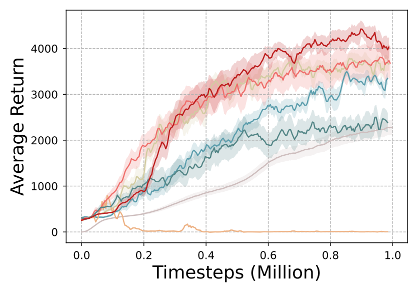
(a) HalfCheetah



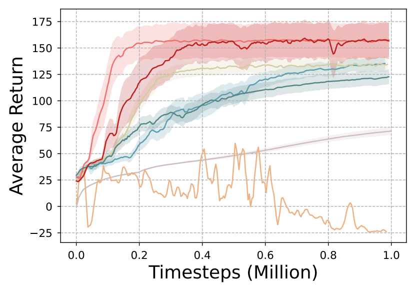
(b) Hopper



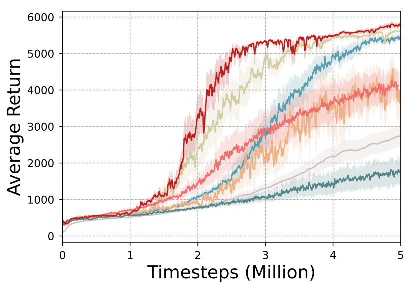
(c) Reacher



(d) Walker2d

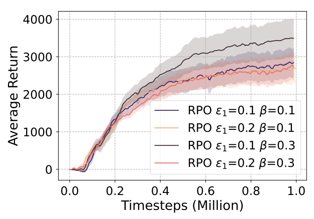


(e) Swimmer

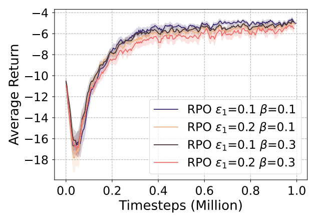


(f) Humanoid

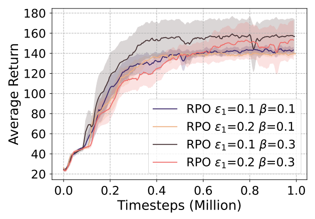
Figure 5. Learning curves on the Gym environments. Performance of RPO vs. PPO, TRPO, OTRPO, GePPO, ISPO and TayPO.



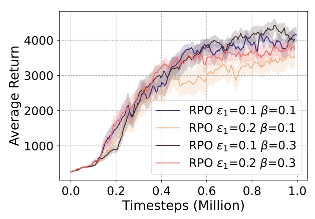
(a) HalfCheetah



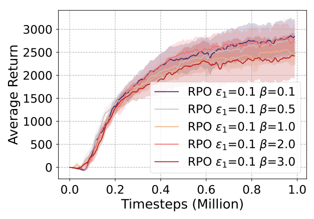
(b) Reacher



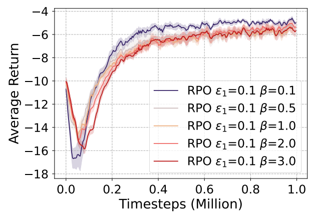
(c) Swimmer



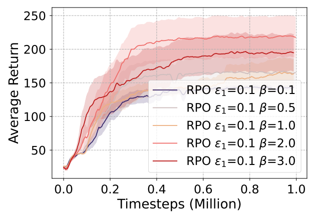
(d) Walker2d



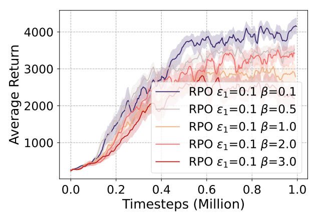
(e) HalfCheetah



(f) Reacher



(g) Swimmer



(h) Walker2d

Figure 6. The top line represents the performance under the condition of β fixed, and the bottom line represents the performance under the condition of ϵ_1 fixed.

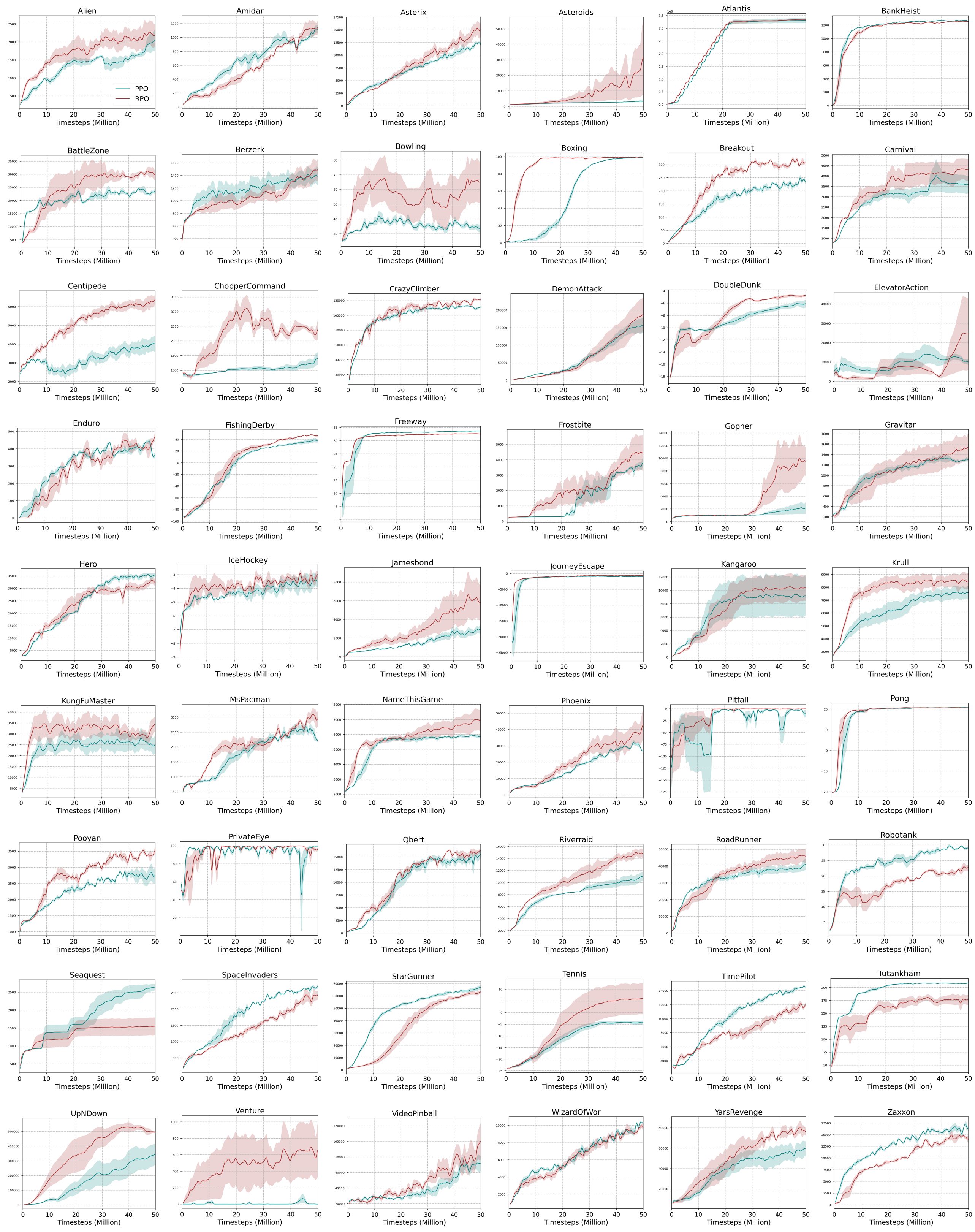


Figure 7. Learning curves on the Atari games. Performance of RPO vs. PPO.

Table 2. Mean of returns of three random seeds of last 100 episodes for different methods in the Atari environments. The best results are highlighted in **bold**. The bottom row counts the number of games where the algorithm or human performed best.

Algorithm	Random	Human	PPO	RPO
Alien	228	**7128**	2037	2201
Amidar	6	**1720**	1121	1148
Asterix	210	8503	12048	**14904**
Asteroids	719	**47389**	3409	31230
Atlantis	12850	29028	3311130	**3347654**
BankHeist	14	753	**1264**	1257
BattleZone	2360	**37188**	23629	29674
Berzerk	124	**2630**	1390	1500
Bowling	23	**161**	33	65
Boxing	0	12	98	**99**
Breakout	2	30	233	**306**
Carnival	380	4000	3578	**4289**
Centipede	2091	**12017**	4007	6361
ChopperCommand	811	**7388**	1407	2394
CrazyClimber	10780	35829	111516	**121374**
DemonAttack	152	1971	157620	**188806**
DoubleDunk	-19	-16	-6	**-4**
ElevatorAction	0	3000	10085	**24473**
Enduro	0	**860**	366	470
FishingDerby	-92	-39	38	**46**
Freeway	0	30	**33**	32
Frostbite	65	4335	3808	**4455**
Gopher	258	2412	2165	**9489**
Gravitar	173	**3351**	1301	1551
Hero	1027	30826	**35477**	32507
IceHockey	-11	**1**	-3	-2
Jamesbond	29	303	2979	**5751**
JourneyEscape	-18000	-1000	-1008	**-609**
Kangaroo	52	3035	9173	**10383**
Krull	1598	2666	7593	**8507**
KungFuMaster	258	22736	25088	**34463**
MsPacman	307	**6952**	2215	2969
NameThisGame	2292	**8049**	5872	6925
Phoenix	761	7243	27074	**43073**
Pitfall	-229	**6464**	-10	0
Pong	-21	15	**21**	**21**
Pooyan	500	1000	2765	**3542**
PrivateEye	25	**69571**	99	96
Qbert	164	13455	15410	**16200**
Riverraid	1338	**17118**	11093	14785
RoadRunner	12	7845	41111	**45982**
Robotank	2	12	**29**	22
Seaquest	68	**42055**	2651	1548
SpaceInvaders	148	1669	**2706**	2445
StarGunner	664	10250	**66997**	63297
Tennis	-24	-8	-4	**5**
TimePilot	3568	5229	**14527**	12012
Tutankham	11	168	**208**	176
UpNDown	533	11693	344972	**494588**
Venture	0	**1188**	0	659
VideoPinball	0	17668	71406	**100495**
WizardOfWor	564	4756	**10347**	9805
YarsRevenge	3093	54577	59645	**75887**
Zaxxon	32	9173	**16258**	14066
Best	0	18	11	26

Table 3. Mean of return for different k in Mujoco.

	PPO	RPO ($k = 2$)	RPO ($k = 3$)	RPO ($k = 4$)
HalfCheetah	2408	3495	4239	4381
Swimmer	134	157	226	227

Table 4. Mean of return of three random seeds for different β in some Atari. The best results are highlighted in **bold**.

	PPO	RPO ($\beta = 1$)	RPO ($\beta = 2$)	RPO ($\beta = 3$)	RPO ($\beta = 4$)	RPO ($\beta = 5$)
Asterix	12048	15198	**17138**	14904	16054	13806
Breakout	233	272	303	306	357	**360**
CrazyClimber	111516	112760	118933	121374	114350	**123890**
Kangaroo	9173	**12434**	10611	10383	10239	11590
Phoenix	27074	28240	34910	**43073**	42200	42082
Qbert	15410	16634	16213	16200	16137	**16958**

Table 5. Hyperparameters for RPO on Mujoco tasks.

Hyperparameter	Value
Discount rate γ	0.995
GAE parameter	0.97
Minibatches per epoch	32
Epochs per update	10
Optimizer	Adam
Learning rate ϕ	3e-4
Minimum batch size (n)	2048
First clipping parameter ϵ	0.2
Second clipping parameter ϵ_1	0.1
Weighting parameter β	0.3

Table 6. Hyperparameters for RPO on Atari tasks.

Hyperparameter	Value
Discount rate γ	0.99
GAE parameter	0.95
Number Workers	16
Epochs per update	4
Optimizer	Adam
Learning rate ϕ	2.5e-4
Rollout Length	256
First clipping parameter ϵ	0.1
Second clipping parameter ϵ_1	0.1
Weighting parameter β	3.0