BEHAVIOR PROXIMAL POLICY OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Offline reinforcement learning (RL) is a challenging setting where existing off-policy actor-critic methods perform poorly due to the overestimation of out-of-distribution actions. Thus, various additional augmentations are proposed to keep the learned policy close to the offline dataset (or behavior policy). In this work, starting from the analysis of offline monotonic policy improvement, we get a surprising finding that *some online on-policy algorithms are naturally able to solve offline RL*. Specifically, the inherent conservatism of these on-policy algorithms is exactly what the offline RL method needs to accomplish the closeness. Based on this, we design an algorithm called **B**ehavior **P**roximal **P**olicy **O**ptimization (BPPO), which successfully solves offline RL without any extra constraint or regularization introduced. Extensive experiments on the D4RL benchmark indicate this extremely succinct method outperforms state-of-the-art offline RL algorithms.

1 INTRODUCTION

Typically, reinforcement learning (RL) is thought of as a paradigm for online learning, where the agent interacts with the environment to collect experience and then uses that to improve itself (Sutton et al., 1998). This online process poses the biggest obstacles to real-world applications because of expensive or even risky data collection in some fields (such as navigation (Mirowski et al., 2018) and healthcare (Yu et al., 2021a)). As an alternative, offline reinforcement learning eliminates the online interaction and learns from a fixed dataset, collected by some arbitrary and possibly unknown process (Lange et al., 2012; Fu et al., 2020). The prospect of this data-driven mode (Levine et al., 2020) is pretty encouraging and has been placed with great expectations for solving real-world applications.

Unfortunately, the major superiority of offline RL, the lack of online interaction, also raises another challenge. The classical off-policy iterative algorithms should be applicable to the offline setting since it is sound to regard offline RL as a more severe off-policy case. But all of them tend to underperform due to the overestimation of out-of-distribution (shorted as OOD) actions. In policy evaluation, the Q-function will poorly estimate the value of OOD state-action pairs. This in turn affects the policy improvement, where agents take the OOD actions with erroneously estimated high values, resulting in low-performance (Fujimoto et al., 2019). Thus, most solutions keep the learned policy close to the behavior policy to overcome the overestimation (Fujimoto et al., 2019; Wu et al., 2019).

Although it has been emphasized that offline RL does not interact with the environment and directly learns from the dataset, in fact, additional online interaction is unavoidable. This is because offline hyperparameter selection is always an open problem lacking satisfactory solutions (Paine et al., 2020; Zhang & Jiang, 2021) and most algorithms have to adapt online interactions to select hyperparameters, which is potentially risky in certain areas (Mirowski et al., 2018; Yu et al., 2021a). If the policy can guarantee better performance than the behavior policy, the risk during online interactions will be greatly reduced. This inspires us to focus on offline monotonic policy improvement, specifically, utilizing offline datasets to improve behavior policy with a monotonic performance guarantee.

To theoretically analyze offline monotonic policy improvement, we introduce the Performance Difference Theorem (Kakade & Langford, 2002). During analysis, we find that the offline setting makes the monotonic policy improvement more complicated to implement, but the essence remains unchanged. This indicates the algorithms derived from *online* monotonic policy improvement (such as Proximal Policy Optimization) can also achieve *offline* monotonic policy improvement under certain conditions. Based on this surprising discovery, we proposed **B**ehavior **P**roximal **P**olicy **O**ptimization (BPPO), an offline algorithm monotonically improves behavior policy in the manner of Proximal

Policy Optimization (PPO). Owing to the inherent conservatism of PPO, our method restricts the ratio of new policy and behavior policy within a certain range, corresponding to offline RL mainstream methods which make the learned policy close to the behavior policy. As offline algorithms are becoming more and more sophisticated, TD3+BC (Fujimoto & Gu, 2021), simply augmenting TD3 (Fujimoto et al., 2018) with behavior cloning (Pomerleau, 1988), reminds us to revisit the simple alternatives with potentially good performance. BPPO is such a "most simple" alternative without introducing any extra constraint or regularization. Extensive experiments on the D4RL benchmark (Fu et al., 2020) indicate BPPO has outperformed state-of-the-art offline RL algorithms.

2 Preliminaries

2.1 Reinforcement Learning

Reinforcement Learning (RL) is a framework of sequential decision. Typically, this problem is formulated by a Markov decision process (MDP) $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, r, p, d_0, \gamma\}$, with state space $\mathcal{S}$, action space $\mathcal{A}$, scalar reward function r, transition dynamics p, initial state distribution $d_0(s_0)$ and discount factor γ (Sutton et al., 1998). The objective of RL is to learn a policy, which defines a distribution over action conditioned on states $\pi\left(a_t|s_t\right)$ at timestep t, where $a_t \in \mathcal{A}, s_t \in \mathcal{S}$. Given this definition, the trajectory $\tau = (s_0, a_0, \cdots, s_H, a_H)$ generated by the agent's interaction with environment $\mathcal{M}$ can be described as a distribution $P_\pi\left(\tau\right) = d_0(s_0) \prod_{t=0}^{H} \pi\left(a_t|s_t\right) p\left(s_{t+1}|s_t, a_t\right)$, where H is the length of the trajectory, and it can be infinite. Then, the goal of RL can be written as an expectation under the trajectory distribution $J\left(\pi\right) = \mathbb{E}_{\tau\sim P_\pi(\tau)}\left[\sum_{t=0}^{H} \gamma^t r(s_t, a_t)\right]$. This objective can also be measured by a state-action value function $Q_\pi\left(s, a\right)$, the expected discounted return given action a in state s: $Q_\pi\left(s, a\right) = \mathbb{E}_{\tau\sim P_\pi(\tau|s,a)}\left[\sum_{t=0}^{H} \gamma^t r(s_t, a_t)|s_0 = s, a_0 = a\right]$. Similarly, the value function $V_\pi\left(s\right)$ is the expected discounted return of certain state s: $V_\pi\left(s\right) = \mathbb{E}_{\tau\sim P_\pi(\tau|s)}\left[\sum_{t=0}^{H} \gamma^t r(s_t, a_t)|s_0 = s\right]$. Then, we can define the advantage function: $A_\pi\left(s, a\right) = Q_\pi\left(s, a\right) - V_\pi\left(s\right)$.

2.2 Offline Reinforcement Learning

In offline RL, the agent only has access to a fixed dataset with transitions $\mathcal{D} = \left\{(s_t, a_t, s_{t+1}, r_t)_{t=1}^{N}\right\}$ collected by the behavior policy π_β. Without interacting with environment $\mathcal{M}$, offline RL expected the agent can infer good behavior from the dataset. Behavior cloning (BC) (Pomerleau, 1988), an approach of imitation learning, can directly imitate the actions of each state with supervised learning:

$$\hat{\pi}_\beta = \underset{\pi}{\operatorname{argmax}}\, \mathbb{E}_{(s,a)\sim\mathcal{D}}\left[\log \pi\left(a|s\right)\right] \tag{1}$$

Note that the performance of $\hat{\pi}_\beta$ trained by behavior cloning highly depends on the quality of transitions. Generally, since the behavior policy π_β generates the offline dataset $\mathcal{D}$, we expect that the behavior policy can be restored by the offline dataset in turn although the error always exists. So we propose an assumption to ensure this.

Assumption 1. *We assume estimated behavior policy and offline dataset $(\hat{\pi}_\beta, \mathcal{D})$ are ξ-coupled pair, that is, $P\left(\hat{\pi}_\beta\left(s\right) \neq \pi_\beta\left(s\right)\right) \leq \xi$ for any state $s \sim \rho_{\mathcal{D}}\left(s\right) = \sum_{t=0}^{H} \gamma^t P\left(s_t = s|\mathcal{D}\right)$, where $P\left(s_t = s|\mathcal{D}\right)$ represents the probability of the t-th state equals to s in offline dataset.*

This means the behavior cloned policy $\hat{\pi}_\beta$ can recover the true behavior policy π_β in at least $1 - \xi$ probability, which means it can serve as an acceptable estimation of π_β. Thus, in the rest of this paper, improving behavior policy actually refers to improving $\hat{\pi}_\beta$.

2.3 Performance Difference Theorem

Theorem 1. *(Kakade & Langford, 2002) Let the discounted unnormalized visitation frequencies as $\rho_\pi\left(s\right) = \sum_{t=0}^{H} \gamma^t P\left(s_t = s|\pi\right)$. For any two policies π and π', the performance difference $J_\Delta\left(\pi', \pi\right) \triangleq J\left(\pi'\right) - J\left(\pi\right)$ can be measured by the advantage function:*

$$J_\Delta\left(\pi', \pi\right) = \mathbb{E}_{\tau\sim P_{\pi'}(\tau)}\left[\sum_{t=0}^{H} \gamma^t A_\pi(s_t, a_t)\right] = \mathbb{E}_{s\sim\rho_{\pi'}(\cdot), a\sim\pi'(\cdot|s)}\left[A_\pi(s, a)\right] \tag{2}$$

Derivation detail is presented in Appendix A. This theorem implies that improving policy from π to π' can be achieved by maximizing (2). From this theorem, Trust Region Policy Optimization (TRPO) (Schulman et al., 2015a) derives an algorithm that can guarantee monotonic performance improvement. Our analysis also applies this theorem to formulate offline monotonic policy improvement.

3 Offline Monotonic Improvement over Behavior Policy

In this section, we theoretically analyze offline monotonic policy improvement based on theorem 1, namely improving the $\hat{\pi}_\beta$ with offline dataset $\mathcal{D}$. Applying the Performance Difference Theorem 1 to the estimated behavior policy $\hat{\pi}_\beta$, and we can get

$$J_\Delta\left(\pi, \hat{\pi}_\beta\right) = \mathbb{E}_{s\sim\rho_\pi(\cdot), a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)\right] \tag{3}$$

Maximizing this equation can obtain a policy better than behavior policy $\hat{\pi}_\beta$. But the above equation is not tractable due to the dependence of new policy's state distribution $\rho_{\pi'}(s)$. For standard online method, $\rho_\pi(s)$ is replaced by the old state distribution $\rho_{\hat{\pi}_\beta}(s)$. In offline setting, we can use the state distribution recovered by offline dataset $\rho_{\mathcal{D}}(s)$ for replacement. Therefore, the approximation of $J_\Delta\left(\pi, \pi_\beta\right)$ is derived as:

$$\widehat{J}_\Delta\left(\pi, \hat{\pi}_\beta\right) = \mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot), a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)\right] \tag{4}$$

Then we measure the difference between $J_\Delta\left(\pi, \hat{\pi}_\beta\right)$ and its approximation. For two discrete probability distributions p, q, the total variation divergence is defined as $D_{TV}(p\|q) = \frac{1}{2}\sum_i |p_i - q_i|$. Use total divergence to describe the distance between policy π and $\hat{\pi}_\beta$: $D_{\mathrm{TV}}^{\max}(\pi, \hat{\pi}_\beta) = \max_s D_{TV}(\pi(\cdot \mid s)\|\pi_\beta(\cdot \mid s))$.

Theorem 2. *Given the distance of π and $\hat{\pi}_\beta$ and the* Assumption 1*, we can derive the following bound:*

$$J_\Delta\left(\pi, \hat{\pi}_\beta\right) \geq \widehat{J}_\Delta\left(\pi, \hat{\pi}_\beta\right) - \frac{4\mathbb{A}_{\hat{\pi}_\beta}\gamma}{(1-\gamma)^2}\alpha^2_{\pi,\hat{\pi}_\beta} - \frac{4\mathbb{A}_{\hat{\pi}_\beta}\gamma\xi^2}{(1-\gamma)\left(1-\gamma\left(1-\xi\right)\right)}, \tag{5}$$

here $\mathbb{A}_{\hat{\pi}_\beta} = \max_{s,a}\left|A_{\hat{\pi}_\beta}(s,a)\right|$ and $\alpha_{\pi,\hat{\pi}_\beta} = D_{\mathrm{TV}}^{\max}(\pi, \hat{\pi}_\beta)$. The proof is presented in Appendix B.

Compared to the theorem in TRPO (Schulman et al., 2015a), the second right term of equation (5) is the same while the third term is unique for the offline situation. The third term only relates to the offline dataset. According to assumption 1, when offline dataset $\mathcal{D}$ is able to fully represent the true behavior policy, ξ approaches zero. In this situation, the third term also approaches zero, which means this theorem degenerates into the online setting. Besides, even if $\xi > 0$, the third term is independent of the target policy π, which means it can be viewed as ***constant***. Thus, we claim:

> Maximizing $\mathbb{E}_{s\sim\rho_{\mathcal{D}(\cdot)}, a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)\right] - \frac{4\mathbb{A}_{\hat{\pi}_\beta}\gamma}{(1-\gamma)^2}\alpha^2_{\pi,\hat{\pi}_\beta}$ can guarantee the true objective $J_\Delta\left(\pi, \hat{\pi}_\beta\right) = \mathbb{E}_{s\sim\rho_\pi(\cdot), a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)\right]$ non-decreasing, which means offline dataset $\mathcal{D}$ does monotonically improve the behavior policy $\hat{\pi}_\beta$.

Suppose we have improved the behavior policy $\hat{\pi}_\beta$ and get one policy π_k with higher performance than the behavior policy. The above theorem can guarantee π_k has higher performance than $\hat{\pi}_\beta$ but we are not sure whether the information in $\mathcal{D}$ has been fully utilized. Although improving $\hat{\pi}_\beta$ enough times is more likely to extract all the information from $\mathcal{D}$ but this intuition lacks strong support. So we might as well assume that π_k does not use the full information of $\mathcal{D}$ and consider whether π_k can be further improved. Applying the Performance Difference Theorem 1 to the policy π_k,

$$J_\Delta\left(\pi, \pi_k\right) = \mathbb{E}_{s\sim\rho_\pi(\cdot), a\sim\pi(\cdot|s)}\left[A_{\pi_k}(s,a)\right] \tag{6}$$

To approximate the above equation, one common manner is replacing the ρ_π with old policy state distribution ρ_{π_k}. However, in offline RL, π_k is forbidden from acting in the environment. As a result, the state distribution ρ_{π_k} is impossible to estimate. Thus, the only choice without other alternative is replaced by the state distribution from offline dataset $\mathcal{D}$:

$$\widehat{J}_\Delta\left(\pi, \pi_k\right) = \mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot), a\sim\pi(\cdot|s)}\left[A_{\pi_k}(s,a)\right] \tag{7}$$

Intuitively, this replacement is reasonable if policy $\pi_k, \hat{\pi}_\beta$ are similar which means this approximation must be related to the distance $\alpha_0 = D_{\mathrm{TV}}^{\max}(\pi_k, \hat{\pi}_\beta)$. Concretely, the gap can be formulated as follows.

Theorem 3. *Given the distance* $\alpha_{\pi,\pi_k} = D_{\mathrm{TV}}^{\max}(\pi, \pi_k)$, $\alpha_0 = D_{\mathrm{TV}}^{\max}(\pi_k, \hat{\pi}_\beta)$ *and the* Assumption 1, *we can derive the following bound:*

$$J_\Delta(\pi, \pi_k) \geq \widehat{J}_\Delta(\pi, \pi_k) - \frac{4\gamma \mathbb{A}_{\pi_k}}{(1-\gamma)^2}\alpha_{\pi,\pi_k}^2 - \frac{4\xi^2\gamma\mathbb{A}_{\pi_k}}{(1-\gamma)(1-\gamma(1-\xi))} - \frac{4\gamma\mathbb{A}_{\pi_k}\alpha_0^2}{(1-\gamma)(1-\gamma(1-\alpha_0))} \tag{8}$$

here $\mathbb{A}_{\pi_k} = \max_{s,a} |A_{\pi_k}(s,a)|$. *The proof is presented in Appendix C.*

Compared to the theorem 2, one additional term related to the distance of $\pi_k, \hat{\pi}_\beta$, namely α_0, has been introduced. Corresponding to the intuition, when the distance approaches zero, $\alpha_0 \to 0$, this additional term also approaches zero. More importantly, this term is irrelevant to the target policy π and can be regarded as ***constant*** when improving π_k. Therefore, we have the following conclusion:

> Maximizing $\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\left[A_{\pi_k}(s,a)\right] - \frac{4\mathbb{A}_{\pi_k}\gamma}{(1-\gamma)^2}\alpha_{\pi,\pi_k}^2$ can guarantee the true objective $J_\Delta(\pi, \pi_k) = \mathbb{E}_{s\sim\rho_\pi(\cdot),a\sim\pi(\cdot|s)}\left[A_{\pi_k}(s,a)\right]$ non-decreasing, which means offline dataset $\mathcal{D}$ can still monotonically improve the π_k, one policy that has been improved by $\mathcal{D}$.

4 Behavior Proximal Policy Optimization

In this section, we describe how to derive one practical algorithm based on the theoretical results from the last section. Within this process, we also explain why the online on-policy method Proximal Policy Optimization (PPO) (Schulman et al., 2017) can solve offline reinforcement learning naturally.

Loss Derivation by PPO: From theorem 2, we know that maximizing $\mathbb{E}_{s\sim\rho_{\mathcal{D}(\cdot)},a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)\right] - \frac{4\mathbb{A}_{\pi_\beta}\gamma}{(1-\gamma)^2}\alpha_{\pi,\hat{\pi}_\beta}^2$ improves the behavior policy $\hat{\pi}_\beta$ monotonically. However, the expectation relates to the new policy: $a \sim \pi(\cdot|s)$, so we adopt importance sampling to make the expectation only depend on the behavior policy $\hat{\pi}_\beta$:

$$L_\beta(\pi) = \mathbb{E}_{s\sim\rho_{\mathcal{D}(\cdot)},a\sim\hat{\pi}_\beta(\cdot|s)}\left[\frac{\pi(a|s)}{\hat{\pi}_\beta(a|s)}A_{\hat{\pi}_\beta}(s,a)\right] - \frac{4\mathbb{A}_{\hat{\pi}_\beta}\gamma}{(1-\gamma)^2}\alpha_{\pi,\hat{\pi}_\beta}^2 \tag{9}$$

For the total divergence $\alpha_{\pi,\hat{\pi}_\beta}$, it is difficult to directly estimate. TRPO uses KL divergence to approximate $\alpha_{\pi,\hat{\pi}_\beta}$ and change the $\max$ operation into expectation. While PPO takes a simpler but more effective approach which clips the ratio of target and old policy $\mathcal{R}_\beta = \frac{\pi(a|s)}{\hat{\pi}_\beta(a|s)}$. We employ identical approach to handle our objective and obtain the following loss function:

$$\hat{L}_\beta(\pi) = \mathbb{E}_{s\sim\rho_{\mathcal{D}},a\sim\hat{\pi}_\beta(\cdot|s)}\left[\min\left(\mathcal{R}_\beta A_{\hat{\pi}_\beta}(s,a), \text{clip}\left(\mathcal{R}_\beta, 1-\epsilon, 1+\epsilon\right)A_{\hat{\pi}_\beta}(s,a)\right)\right] \tag{10}$$

Advantage Estimation by Function Approximation: In traditional online setting, advantage is calculated by Generalized Advantage Estimation (GAE) (Schulman et al., 2015b). The TD residual $\delta_t = r(s_t, a_t) + \gamma V_\pi(s_{t+1}) - V_\pi(s_t)$ can be regarded as an estimate of the advantage of action a_t, namely $A_\pi(s_t, a_t) = \mathbb{E}_{s_{t+1}}\delta_t$. Here only single step reward is considered. GAE simultaneously considers TD residual with different steps and sums them with certain weights. After derivation, the expression of GAE is:

$$\hat{A}_t^{\mathrm{GAE}(\gamma,\lambda)} := \sum_{l=0}^{\infty}(\gamma\lambda)^l\delta_{t+l}^V \tag{11}$$

where $0 < \lambda < 1$ makes a compromise between bias and variance. Note that GAE is not a kind of function approximation method, which represents GAE is incapable of generalization. Therefore, in offline setting, GAE not a satisfactory choice since only the advantage of $(s,a) \in \mathcal{D}$ can be calculated while $(s,a) \sim (\rho_{\mathcal{D}}(s), \hat{\pi}_\beta(a|s))$ cannot. But actually, we expect the advantage has the ability of in-distribution generalization. In this way, when calculating loss function (10), the actions can be sampled by estimated behavior policy $\hat{\pi}_\beta$, not just actions from $\mathcal{D}$. As a result, we directly estimate advantage with the definition $A_\pi(s,a) = Q_\pi(s,a) - V_\pi(s)$, where Q-function is estimated by SARSA and value function by fitting returns $\sum_{t=0}^{H}\gamma^t r(s_t, a_t)$ with MSE loss.

Advantage Generalization of $A_{\pi_k}(s,a)$: When improving policy π_k, the objective to be maximized can be treated in the same way to derive a similar loss function. But the most tough thing is the existence of $A_{\pi_k}(s,a) = Q_{\pi_k}(s,a) - V_{\pi_k}(s)$. Q-learning does estimate $Q_{\pi_k}(s,a)$ but this is potentially risky because of poor estimation on out-of-distribution actions. The value function seems impossible to learn unless other models are introduced (such as the reward model and dynamics model). By comparison, one simple and straightforward approach is changing $A_{\pi_k}(s,a)$ into $A_{\hat{\pi}_\beta}(s,a)$. When improving policy from $\hat{\pi}_\beta$ to π_k, the penalty term forces them to be close, which means this change is acceptable in some certain situations. More concretely, we give the following bound to measure the gap:

Theorem 4. *Assume the reward function satisfies $|r(s,a)| \leq R_{max}$ for all s, a.*

$$\left|\widehat{J}_\Delta(\pi,\pi_k) - \mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)\right]\right| \leq 2R_{max}\frac{1+(1-\alpha_0)\left(\gamma^2-\gamma\right)}{(1-\gamma)\left(1-\gamma\left(1-\alpha_0\right)\right)}(\gamma+1) \quad (12)$$

Here $\alpha_0 = D_{\mathrm{TV}}^{\max}(\pi_k, \hat{\pi}_\beta)$. The proof is presented in Appendix D.

Besides the environment property R_{max}, the distance of $\pi_k, \hat{\pi}_\beta$ is introduced. But fortunately, this bound does not relate to target policy π which means when updating π, this additional term has no negative effects. Therefore, we decide to replace $A_{\pi_k}(s,a)$ with $A_{\hat{\pi}_\beta}(s,a)$ rather than re-estimate. Then the loss function for improving $\hat{\pi}_\beta$ and π_k can be unified into one expression:

$$\hat{L}_k(\pi) = \mathbb{E}_{s\sim\rho_{\mathcal{D}},a\sim\pi_k(\cdot|s)}\left[\min\left(\mathcal{R}_k A_{\hat{\pi}_\beta}(s,a), \mathrm{clip}\left(\mathcal{R}_k, 1-\epsilon, 1+\epsilon\right)A_{\hat{\pi}_\beta}(s,a)\right)\right], k = 0,1,2,\cdots \quad (13)$$

Especially, we set $\pi_0 = \hat{\pi}_\beta$ since $\hat{\pi}_\beta$ is the first policy to be improved and in the following section we will no longer deliberately distinguish $\hat{\pi}_\beta, \pi_k$. So far, we have made it clear how to start from offline monotonic policy improvement to PPO. The whole algorithm process has been summarized in Algorithm 1.

Algorithm 1 Behavior Proximal Policy Optimization

Require: Offline dataset $\mathcal{D}$;
Estimate behavior policy $\hat{\pi}_\beta$ by behavior cloning;
Calculate Q-function $Q_{\hat{\pi}_\beta}$ by SARSA;
Calculate value function $V_{\hat{\pi}_\beta}$ by fitting returns with MSE;
Compute advantage function $A_{\hat{\pi}_\beta} = Q_{\hat{\pi}_\beta} - V_{\hat{\pi}_\beta}$;
Set $\pi_0 = \hat{\pi}_\beta$;
for $k = 0, 1, 2, \cdots, K$ **do**
  while $\pi \leq \pi_k$ **do**
    Update policy π by maximizing $\hat{L}_k(\pi)$ in Equation (13);
  end while
end for

What is the relation between BPPO and Onestep RL? Because BPPO is proposed based on on-policy algorithm, it is easy to associate with Onestep RL (Brandfonbrener et al., 2021), a method that solves offline RL without off-policy evaluation. The final version of BPPO only estimates the advantage function once but updates policy K times, which means BPPO is neither Onestep method nor multi-step method. If we remove the outside loop in algorithm 1, that is let $K = 0$, our algorithm becomes Onestep version. The right figure shows the difference between BPPO and its Onestep version. Onestep strictly requires that the new policy be close to the behavior policy $\hat{\pi}_\beta$, while BPPO appropriately loosens this restriction which may lead to a better policy. At the same time, to prevent the learned policy completely away from $\hat{\pi}_\beta$, we also adopt clip ratio decay.

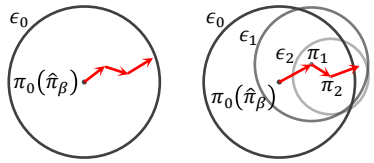


Figure 1: The difference between Onestep BPPO (left) and BPPO (right), where the decreasing circle corresponds to clip ratio decay and the detail can be found in Section 6.3.

Why BPPO can solve offline RL? Although the previous proofs and derivations have proved the rationality of solving offline RL by BPPO, we want to directly explain it from the perspective of offline itself. Lots of previous work has demonstrated that closeness to behavior policy is an effective manner. According to (13), we make the following restrictions on the learned policy:

$$1-\epsilon \leq \frac{\pi\left(a|s\right)}{\pi_k\left(a|s\right)} \leq 1+\epsilon \tag{14}$$

If $k=0$, this equation ensures the closeness between learned policy and $\hat{\pi}_\beta$ by constraining the ratio. When $k>0$, the closeness between learned policy and π_k indirectly constrains the closeness between $\hat{\pi}_\beta$. This conservatism from an online setting is exactly suitable for offline RL, so we believe that any extra constraint or regularization is unnecessary.

5 Related Work

Offline Reinforcement Learning Most of the online off-policy methods fail or underperform in offline reinforcement learning due to extrapolation error (Fujimoto et al., 2019) or distributional shift (Levine et al., 2020). Thus most offline algorithms typically augment existing off-policy algorithms with a penalty measuring divergence between the policy and the offline data (or behavior policy). Depending on how to implement this penalty, a variety of methods were proposed such as batch constrained (Fujimoto et al., 2019), KL-control (Jaques et al., 2019), behavior-regularized (Wu et al., 2019; Fujimoto & Gu, 2021) and policy constraint (Kumar et al., 2019; Levine et al., 2020; Kostrikov et al., 2021b). Other methods augment BC with a weight to make the policy favor high advantage actions (Wang et al., 2018; Siegel et al., 2020; Peng et al., 2019; Wang et al., 2020). Another methods extra introduced Uncertainty estimation (An et al., 2021; Bai et al., 2022) or conservative (Kumar et al., 2020; Yu et al., 2021b; Nachum et al., 2019) estimation to overcome overestimation.

Monotonic Policy Improvement Monotonic policy improvement has been well studied both in model-free (Kakade & Langford, 2002; Schulman et al., 2015a) and model-based methods (Luo et al., 2018) in online RL. Xu et al. (2021) mentions offline algorithms lack guaranteed performance improvement over the behavior policy but does not consider offline monotonic policy improvement.

6 Experiments

We conduct a series of experiments to evaluate the performance of **B**ehavior **P**roximal **P**olicy **O**ptimization (BPPO) in comparison to the previous offline RL methods. To this end, we aim to answer the following key questions: 1) How does our BPPO compare with the state-of-the-art offline RL methods, i.e., the Onestep methods and iterative dynamic programming methods? 2) What is the superiority of BPPO over its Onestep version? 3) What is the influence of different hyperparameters and is BPPO robust to these hyperparameters? To answer these questions, we evaluate our method on the Gym dataset (v2), Adroit dataset (v1), and Kitchen dataset (v0) from D4RL (Fu et al., 2020).

6.1 Results on D4RL Benchmarks

We first evaluate whether BPPO can outperform the prior state-of-the-art iterative and Onestep offline reinforcement learning methods. We compare BPPO with Onestep methods including Onestep RL (Brandfonbrener et al., 2021) and IQL (Kostrikov et al., 2021b), and iterative dynamic programming methods including CQL (Kumar et al., 2020), TD3+BC (Fujimoto & Gu, 2021), and F-BRC (Kostrikov et al., 2021a). The performance of all baseline methods is listed in Table 1. The results of Onestep RL, IQL, CQL, and TD3+BC are extracted from the paper IQL, while F-BRC is directly from the original paper. Since BPPO first estimated a behavior policy and then improves it, we list the results of BC on the left side of BPPO.

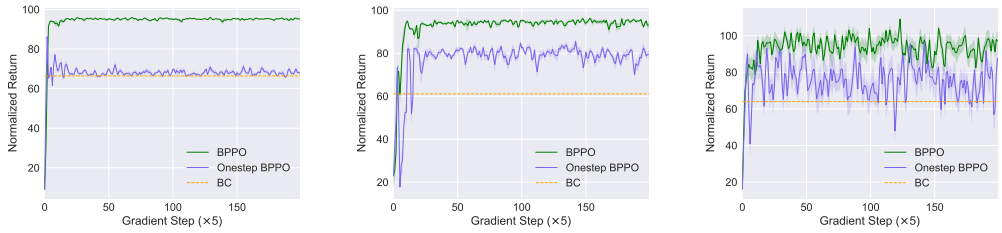
From Table 1, we can find BPPO achieves comparable performance on each task of Gym and slightly outperformed when considering the total performance. For Adroit and Kitchen, BPPO prominently outperforms other methods. Compared to BC, the start point of our method, BPPO achieves 51% improvement in total returns of all D4RL tasks. Interestingly, the results of our implemented BC on Adroit and Kitchen nearly outperform the baselines, which may imply improving behavior policy rather than learning from scratch is a better approach.

Table 1: The normalized results of iterative, Onestep, our BC, and our BPPO algorithms on D4RL Gym, Adroit, and Kitchen tasks. We **bold** the best results on each environment. For all benchmark environments, we average mean returns overs 10 evaluation trajectories and five random seeds.

Suite	Environment	Iterative methods			Onestep methods		BC (Ours)	BPPO (Ours)
		CQL	TD3+BC	F-BRC	Onestep RL	IQL		
Gym	halfcheetah-medium-v2	44.0	**48.3**	41.3	**48.4**	**47.4**	43.5±0.1	44.0±0.2
	hopper-medium-v2	58.5	59.3	**99.4**	59.6	66.3	61.3±3.2	93.9±3.9
	walker2d-medium-v2	72.5	**83.7**	78.8	81.8	78.3	74.2±4.6	**83.6±0.9**
	halfcheetah-medium-replay-v2	**45.5**	**44.6**	43.2	38.1	**44.2**	40.1±0.1	41.0±0.6
	hopper-medium-replay-v2	95.0	60.9	35.6	**97.5**	94.7	66.0±18.3	92.5±3.4
	walker2d-medium-replay-v2	77.2	**81.8**	41.8	49.5	73.9	33.4±11.2	77.6±7.8
	halfcheetah-medium-expert-v2	91.6	90.7	**93.3**	**93.4**	86.7	64.4±8.5	**92.5±1.9**
	hopper-medium-expert-v2	105.4	98	**112.4**	103.3	91.5	64.9±7.7	**112.8±1.7**
	walker2d-medium-expert-v2	108.8	110.1	105.2	**113.0**	109.6	107.7±3.5	**113.1±2.4**
	Gym locomotion-v2 total	*698.5*	*677.4*	*651.0*	*684.6*	*692.4*	*555.5±57.2*	***751.0±21.8***
Adorit	pen-human-v1	37.5	-	-	-	71.5	61.6±9.7	**117.8±11.9**
	hammer-human-v1	4.4	-	-	-	1.4	2.0±0.9	**14.9±3.2**
	door-human-v1	9.9	-	-	-	4.3	7.8±3.5	**25.9±7.5**
	relocate-human-v1	0.2	-	-	-	0.1	0.1±0.0	**4.8±2.2**
	pen-cloned-v1	39.2	-	-	60.0	37.3	58.8±16.0	**110.8±6.3**
	hammer-cloned-v1	2.1	-	-	2.0	2.1	0.5±0.2	**8.9±5.1**
	door-cloned-v1	0.4	-	-	0.4	1.6	0.9±0.8	**6.2±1.6**
	relocate-cloned-v1	-0.1	-	-	-0.1	-0.2	-0.1±0.0	**1.9±1.0**
	adroit-v1 total	*93.6*	-	-	*62.3*	*118.1*	*131.6±31.1*	***291.4±38.8***
Kitchen	kitchen-complete-v0	43.8	-	-	-	62.5	55.0±11.5	**91.5±8.9**
	kitchen-partial-v0	49.8	-	-	-	46.3	44.0±4.9	**57.0±2.4**
	kitchen-mixed-v0	51.0	-	-	-	51.0	45.0±1.6	**62.5±6.7**
	kitchen-v0 total	*144.6*	-	-	-	*159.8*	*144.0±18.0*	***211.0±18.0***
	locomotion+kitchen+adroit	*936.7*	-	-	-	*970.3*	*831.1±106.3*	***1253.4±78.6***

6.2 THE SUPERIORITY OF BPPO OVER ONESTEP VERSION

About why we choose to improve policy π_k again after it has been improved over behavior policy, our previous explanation is that theorem 2 does not ensure all information from the offline dataset is extracted. But when improving policy π_k, we utilize twice approximation. Therefore, it is necessary for us to demonstrate the effectiveness of improving policy π_k. Besides, BPPO is similar to Onestep RL. Although Figure 1 gives an intuitive interpretation to show the advantage of BPPO over Onestep, the soundness is relatively weak. Based on the above two considerations, we carry on this experiment to analyze the superiority of BPPO over its Onestep version.



(a) hopper-medium-v2 (b) hopper-medium-replay-v2 (c) hopper-medium-expert-v2

Figure 2: The comparison between BPPO and Onestep BPPO.

As shown in Figure2, we often observe that BPPO and Onestep BPPO can outperform the BC (shown in the yellow dotted line). This phenomenon shows that both of them are able to achieve offline monotonic improvement over behavior policy $\hat{\pi}_\beta$. Another important result is that BPPO stably better than Onestep BPPO and this demonstrates two key points. First, the necessity of improving π_k again to fully utilize information from the offline dataset has been proved. Second, compared to strictly restricting the learned policy close to behavior policy, appropriate looseness is particularly helpful. At the same time, the clip decay (15) prevents this looseness within control.

The stable performance benefits from the on-policy improvement based on the behavior policy. BPPO is distinct from prior iterative methods which achieve off-policy evaluation and improvement via maximizing the Q function. In this setting, Q-function will poorly estimate the value of OOD state-action pairs in the policy evaluation. Meanwhile, prior iterative methods that rely on off-policy evaluation will struggle with iterative error exploitation simultaneously. As a consequence, the combination of distribution shift and iterative error exploitation causes overestimation dramatically (Brandfonbrener et al., 2021). In contrast, BPPO is not achieving policy improvement by iterative evaluation and improvement by maximizing the Q function, but rather realizing monotonic policy improvement in an on-policy manner which can avoid the evaluation error (Brandfonbrener et al., 2021).

6.3 Is BPPO Robust on Different Hyperparameters?

In this section, we evaluate whether BPPO is sensitive to different hyperparameters via conducting a series of ablation experiments. We analyze how the clip coefficient ϵ and the asymmetric coefficient ω affect the performance of BPPO. The clip coefficient restricts the policy improvement in a certain region close to behavior policy and it can be viewed as special coefficient for dealing with Overestimation. Because ϵ is from PPO, we have enough prior information to ensure a proper range to avoid catastrophic performance, which is the unique feature of BPPO. And the asymmetric coefficient can adjust the weight of advantage based on the Q performance, which downweights the contributions of the state-action value Q smaller than its expectation while distributing more weights to advantage with a larger Q value.

Besides, we empirically implement some special tricks for the BPPO:

- clip coefficient decay. The clip function can be regarded as the conservative item that limit the distance between the target policy and old policy. As the policy updates, we tight the clip coefficient ϵ gradually:

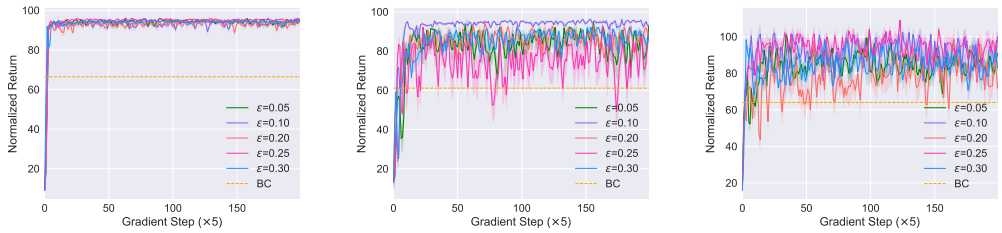
$$\epsilon_k = \begin{cases} \epsilon \times (\sigma)^{k-1}, & if \quad k \leq 200 \\ \epsilon_{200}, & else \end{cases}. \tag{15}$$

  here $k \in \{1, 2, .., K\}$ denotes the gradient steps, ϵ denotes the initial clip coefficient value, and σ is the decay coefficient (here we use $\sigma = 0.96$).

- Advantage with asymmetric coefficient. We introduce the $\omega \in (0, 1)$ as an asymmetric coefficient for advantage calculation:

$$\bar{A}_{\hat{\pi}_\beta} = |\omega - \mathbf{1}(A_{\hat{\pi}_\beta} < 0)| A_{\hat{\pi}_\beta}. \tag{16}$$

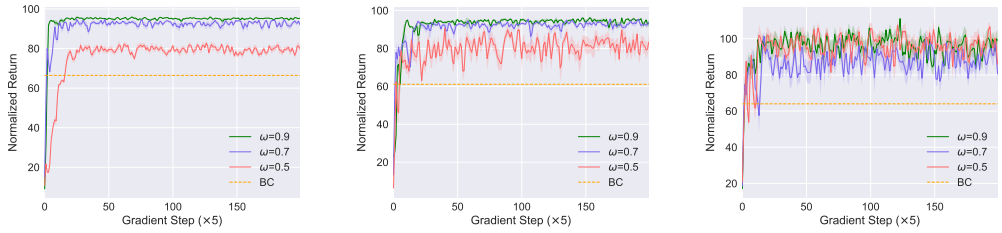
  For $\omega > 0.5$, that downweights the contributions of the state-action value $Q_{\hat{\pi}_\beta}$ smaller than it's expectation, i.e., $V_{\hat{\pi}_\beta}$ while distributing more weights to larger $Q_{\hat{\pi}_\beta}$.



(a) hopper-medium-v2 (b) hopper-medium-replay-v2 (c) hopper-medium-expert-v2

Figure 3: Ablation study on coefficient ϵ.

Firstly, we analyze five values of the clip coefficient $\epsilon = (0.05, 0.1, 0.2, 0.25, 0.3)$. As shown in Figure 3, for the environment *hopper-medium-v2*, it is stable on various coefficient values. We observe that for *hopper-medium-v2* and *hopper-medium-expert-v2* different ϵ shows no significant difference so we choose $\epsilon = 0.25$, while $\epsilon = 0.1$ is obviously better than others for *hopper-medium-replay-v2*.



(a) hopper-medium-v2 (b) hopper-medium-replay-v2 (c) hopper-medium-expert-v2

Figure 4: Ablation study on coefficient ω.

We then analyze three values of the asymmetric coefficient $\omega = (0.5, 0.7, and 0.9)$ in three Gym environments. Figure 4 shows that $\omega = 0.9$ is best for these tasks, especially in *hopper-medium-v2* and *hopper-medium-replay-v2*. With a larger value ω, the policy improvement can be guided in a better direction, leading to better performance in Gym environments. Although there are existing different performances on various coefficient values, all results significantly outperform the BC performances and consistently maintain a high performance. Comparing the delicate performance of iterative methods on various offline hyperparameters, BPPO is robust to the coefficients.

Based on the performance of different coefficient values above, we use the asymmetric advantage coefficient $\omega = 0.9$ for the Gym dataset training and $\omega = 0.7$ for the Adroit and Kitchen datasets training, respectively. For the initial clip coefficient ϵ, we select $\epsilon = 0.25$ for all three datasets except the 'medium-replay' tasks of the Gym dataset while $\epsilon = 0.1$ performs best for those tasks. For more details about the use of other hyperparameters see Appendix E.

Other offline methods like CQL (Kumar et al., 2020) and BCQ (Fujimoto et al., 2019) all introduces additional term to solve overestimation. Thus, the value of this additional hyperparameter must greatly affect the performance and careful selection is essential (Emmons et al., 2021). But this situation will not occur in BPPO. We have repeatedly stressed BPPO does not introduce any extra constraint or regularization compared to PPO and only the logic of the training stage has been modified in the offline setting. Therefore, we can directly select the hyperparameters within a reasonable range. In contrast, the performance of most of the prior offline RL methods is very sensitive and delicate on the regularization hyperparameters in the offline RL setting (Brandfonbrener et al., 2021), especially for iterative methods. BPPO has great potential to solve real-world applications, especially in those cases where online interaction with the environment is strictly prohibited.

7 Discussion and Future Work

Most previous work solves offline reinforcement learning by "addition" based on the off-policy method while we focus on the on-policy method. It's counter-intuitive to use the on-policy method to solve offline RL, a special case of an off-policy situation. But we believe the conservatism of the on-policy method, which is designed to ensure the stability of online learning, is naturally suitable for offline reinforcement learning. Thus we think about how to utilize this conservatism and simultaneously eliminate the conflict between on-policy and offline (a kind of more severe off-policy). Finally, starting from offline monotonic policy improvement, we derive a method using the PPO loss function for offline reinforcement learning. We name it Behavior Proximal Policy Optimization and it solves offline RL without "addition".

Although BPPO achieves inspiring results on some datasets, the "stitching" problem is extremely challenging and BPPO performs poorly in an Antmaze environment. We will consider how to enhance BPPO with the ability of combining data. Offline to online finetuning may be the problem that suits BPPO since no additional coefficient is introduced which means switching is avoided. But a common setting is to collect data first and then add it into offline dataset, which is conflict with our approach, since we should apply PPO in online setting. Besides, testing BPPO on some real-world applications is attractive such as a quadruped robot. Owing to the robustness to hyperparameters, BPPO may be able achieve good performance without expensive online interaction.

REFERENCES

Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-based offline reinforcement learning with diversified q-ensemble. *Advances in neural information processing systems*, 34:7436–7447, 2021.

Chenjia Bai, Lingxiao Wang, Zhuoran Yang, Zhihong Deng, Animesh Garg, Peng Liu, and Zhaoran Wang. Pessimistic bootstrapping for uncertainty-driven offline reinforcement learning. *arXiv preprint arXiv:2202.11566*, 2022.

David Brandfonbrener, Will Whitney, Rajesh Ranganath, and Joan Bruna. Offline rl without off-policy evaluation. *Advances in Neural Information Processing Systems*, 34:4933–4946, 2021.

Scott Emmons, Benjamin Eysenbach, Ilya Kostrikov, and Sergey Levine. Rvs: What is essential for offline rl via supervised learning? *arXiv preprint arXiv:2112.10751*, 2021.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep rl: A case study on ppo and trpo. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=r1etN1rtPB`.

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. *Advances in neural information processing systems*, 34:20132–20145, 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.

Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. *arXiv preprint arXiv:1907.00456*, 2019.

Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *In Proc. 19th International Conference on Machine Learning*. Citeseer, 2002.

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, 2021a.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021b.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in Neural Information Processing Systems*, 32, 2019.

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

Sascha Lange, Thomas Gabel, and Martin Riedmiller. Batch reinforcement learning. In *Reinforcement learning*, pp. 45–73. Springer, 2012.

David A. Levin, Yuval Peres, and Elizabeth Wilmer. Markov chains and mixing times. 2008.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Yuping Luo, Huazhe Xu, Yuanzhi Li, Yuandong Tian, Trevor Darrell, and Tengyu Ma. Algorithmic framework for model-based deep reinforcement learning with theoretical guarantees. *arXiv preprint arXiv:1807.03858*, 2018.

Piotr Mirowski, Matt Grimes, Mateusz Malinowski, Karl Moritz Hermann, Keith Anderson, Denis Teplyashin, Karen Simonyan, Andrew Zisserman, Raia Hadsell, et al. Learning to navigate in cities without a map. *Advances in Neural Information Processing Systems*, 31, 2018.

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.

Tom Le Paine, Cosmin Paduraru, Andrea Michi, Caglar Gulcehre, Konrad Zolna, Alexander Novikov, Ziyu Wang, and Nando de Freitas. Hyperparameter selection for offline reinforcement learning. *arXiv preprint arXiv:2007.09055*, 2020.

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

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, 1, 1988.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015a.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015b.

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

Noah Y Siegel, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael Neunert, Thomas Lampe, Roland Hafner, Nicolas Heess, and Martin Riedmiller. Keep doing what worked: Behavioral modelling priors for offline reinforcement learning. *arXiv preprint arXiv:2002.08396*, 2020.

Richard S Sutton, Andrew G Barto, et al. Introduction to reinforcement learning. 1998.

Qing Wang, Jiechao Xiong, Lei Han, Han Liu, Tong Zhang, et al. Exponentially weighted imitation learning for batched historical data. *Advances in Neural Information Processing Systems*, 31, 2018.

Ziyu Wang, Alexander Novikov, Konrad Zolna, Josh S Merel, Jost Tobias Springenberg, Scott E Reed, Bobak Shahriari, Noah Siegel, Caglar Gulcehre, Nicolas Heess, et al. Critic regularized regression. *Advances in Neural Information Processing Systems*, 33:7768–7778, 2020.

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

Haoran Xu, Xianyuan Zhan, Jianxiong Li, and Honglei Yin. Offline reinforcement learning with soft behavior regularization. *arXiv preprint arXiv:2110.07395*, 2021.

Chao Yu, Jiming Liu, Shamim Nemati, and Guosheng Yin. Reinforcement learning in healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(1):1–36, 2021a.

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. Combo: Conservative offline model-based policy optimization. *Advances in neural information processing systems*, 34:28954–28967, 2021b.

Siyuan Zhang and Nan Jiang. Towards hyperparameter-free policy selection for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 34:12864–12875, 2021.

A PROOF OF PERFORMANCE DIFFERENCE THEOREM 1

Proof. First note that $A_\pi(s,a) = \mathbb{E}_{s'\sim p(s'|s,a)}\left[r(s,a) + \gamma V_\pi\left(s'\right) - V_\pi(s)\right]$. Therefore,

$$
\begin{aligned}
&\mathbb{E}_{\tau\sim P_{\pi'}}\left[\sum_{t=0}^{H}\gamma^t A_\pi\left(s_t,a_t\right)\right]\\
=&\mathbb{E}_{\tau\sim P_{\pi'}}\left[\sum_{t=0}^{H}\gamma^t\left(r\left(s_t,a_t\right)+\gamma V_\pi\left(s_{t+1}\right)-V_\pi\left(s_t\right)\right)\right]\\
=&\mathbb{E}_{\tau\sim P_{\pi'}}\left[-V_\pi\left(s_0\right)+\sum_{t=0}^{H}\gamma^t r\left(s_t,a_t\right)\right]\\
=&-\mathbb{E}_{s_0}\left[V_\pi\left(s_0\right)\right]+\mathbb{E}_{\tau\sim P_{\pi'}}\left[\sum_{t=0}^{H}\gamma^t r\left(s_t,a_t\right)\right]\\
=&-J\left(\pi\right)+J\left(\pi'\right)\\
\triangleq&J_\Delta\left(\pi',\pi\right)
\end{aligned}
\tag{17}
$$

Now the first equation in 1 has been proved. For the proof of second equation, we decompose the expectation over the trajectory into the sum of expectation over state-action pairs:

$$
\begin{aligned}
&\mathbb{E}_{\tau\sim P_{\pi'}}\left[\sum_{t=0}^{H}\gamma^t A_\pi\left(s_t,a_t\right)\right]\\
=&\sum_{t=0}^{H}\sum_{s}\left[P\left(s_t=s|\pi'\right)\mathbb{E}_{a\sim\pi'(\cdot|s)}\left[\gamma^t A_\pi\left(s,a\right)\right]\right]\\
=&\sum_{s}\left[\sum_{t=0}^{H}\gamma^t P\left(s_t=s|\pi'\right)\mathbb{E}_{a\sim\pi'(\cdot|s)}\left[A_\pi\left(s,a\right)\right]\right]\\
=&\sum_{s}\left[\rho_{\pi'(s)}\mathbb{E}_{a\sim\pi'(\cdot|s)}\left[A_\pi\left(s,a\right)\right]\right]\\
=&\mathbb{E}_{s\sim\rho_{\pi'(s)},a\sim\pi'(\cdot|s)}\left[A_\pi\left(s,a\right)\right]
\end{aligned}
\tag{18}
$$

B PROOF OF THEOREM 2

The expectation of advantage function $A_\pi\left(s,a\right)$ over its policy π equals zero:

$$
\mathbb{E}_{a\sim\pi}\left[A_\pi(s,a)\right] = \mathbb{E}_{a\sim\pi}\left[Q_\pi(s,a) - V_\pi(s)\right] = \mathbb{E}_{a\sim\pi}\left[Q_\pi(s,a)\right] - V_\pi(s) = 0 \tag{19}
$$

In theorem 2, the expectation of advantage function $A_{\hat{\pi}_\beta}\left(s,a\right)$ depends on another policy π rather than $\hat{\pi}_\beta$. The definition of $\bar{A}_{\pi,\hat{\pi}_\beta}\left(s\right)$ is as follows:

$$
\bar{A}_{\pi,\hat{\pi}_\beta}\left(s\right) = \mathbb{E}_{a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}\left(s,a\right)\right] \tag{20}
$$

Furthermore, given the $\bar{A}_{\pi,\hat{\pi}_\beta}\left(s\right)$, the performance difference in theorem 2 can be rewritten as:

$$
J_\Delta\left(\pi,\hat{\pi}_\beta\right) = \mathbb{E}_{s\sim\rho_\pi(\cdot),a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)\right] = \mathbb{E}_{s\sim\rho_\pi(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}\left(s\right)\right] \tag{21}
$$

$$
\widehat{J}_\Delta\left(\pi,\hat{\pi}_\beta\right) = \mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)\right] = \mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}\left(s\right)\right] \tag{22}
$$

Lemma 1. *Let $\pi,\hat{\pi}_\beta$ be α-coupled policies, that is, $P\left(a\neq a_{\hat{\beta}}|s\right) = P\left(\pi\left(s\right)\neq\hat{\pi}_\beta\left(s\right)\right)\leq\alpha$ holds for all state s. Then for all state s,*

$$
|\bar{A}_{\pi,\hat{\pi}_\beta}\left(s\right)| \leq 2\alpha\max_{s,a}\left|A_{\hat{\pi}_\beta}(s,a)\right| \tag{23}
$$

Proof. Since $\mathbb{E}_{a_{\hat{\beta}}\sim\hat{\pi}_\beta}\left[A_{\hat{\pi}_\beta}(s,a)\right]=0$ (from equation (19)), we can subtract it if needed:

$$\begin{aligned}\bar{A}_{\pi,\hat{\pi}_\beta}(s)&=\mathbb{E}_{a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)\right]=\mathbb{E}_{a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)\right]-\mathbb{E}_{a_{\hat{\beta}}\sim\hat{\pi}_\beta(\cdot|s)}\left[A_{\hat{\pi}_\beta}\left(s,a_{\hat{\beta}}\right)\right]\\&=\mathbb{E}_{a\sim\pi(\cdot|s),a_{\hat{\beta}}\sim\hat{\pi}_\beta(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)-A_{\hat{\pi}_\beta}\left(s,a_{\hat{\beta}}\right)\right]\end{aligned}\tag{24}$$

The α-coupled condition implies that only the $a\neq a_{\hat{\beta}}$ situation needs to be discussed:

$$\begin{aligned}\left|\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right|&=\left|P(a\neq a_{\hat{\beta}})\mathbb{E}_{a\neq a_{\hat{\beta}}}\left[A_{\hat{\pi}_\beta}(s,a)-A_{\hat{\pi}_\beta}(s,a_{\hat{\beta}})\right]\right|\\&\leq\alpha\cdot\left|\mathbb{E}_{a\neq a_{\hat{\beta}}}\left[A_{\hat{\pi}_\beta}(s,a)-A_{\hat{\pi}_\beta}(s,a_{\hat{\beta}})\right]\right|\\&\leq 2\alpha\max_{s,a}\left|A_{\hat{\pi}_\beta}(s,a)\right|\end{aligned}\tag{25}$$

Lemma 2. *Let $\pi,\hat{\pi}_\beta$ be α-coupled policies, that is, $P\left(a\neq a_{\hat{\beta}}|s\right)=P\left(\pi(s)\neq\hat{\pi}_\beta(s)\right)\leq\alpha$ holds for all state s. Then*

$$\left|\mathbb{E}_{s\sim\rho_\pi(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]-\mathbb{E}_{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]\right|\leq\frac{4\alpha^2\gamma\max_{s,a}\left|A_{\hat{\pi}_\beta}(s,a)\right|}{(1-\gamma)^2}\tag{26}$$

Proof. According to the equation (18), we have

$$\begin{aligned}&\left|\mathbb{E}_{s\sim\rho_\pi(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]-\mathbb{E}_{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]\right|\\=&\left|\sum_{t=0}^{H}\sum_s\left[\gamma^tP(s_t=s|\pi)\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]\right]-\sum_{t=0}^{H}\sum_s\left[\gamma^tP(s_t=s|\hat{\pi}_\beta)\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]\right]\right|\\=&\sum_{t=0}^{H}\gamma^t\left|\sum_sP(s_t=s|\pi)\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]-\sum_sP(s_t=s|\hat{\pi}_\beta)\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]\right|\end{aligned}\tag{27}$$

Use $s_{t|\pi}$ to denote the t-th timestep state generated by policy π and the same as $s_{t|\hat{\pi}_\beta}$. If $s_{t|\pi}=s_{t|\hat{\pi}_\beta}$ holds, when and only when all the actions taken before timestep t are identical. Based on the definition of α-coupled, the probability of $s_{t|\pi}=s_{t|\hat{\pi}_\beta}$ is:

$$P\left(s_{t|\pi}=s_{t|\hat{\pi}_\beta}\right)\geq(1-\alpha)^t\tag{28}$$

As the complementary set, the probability of $s_{t|\pi}\neq s_{t|\hat{\pi}_\beta}$ is:

$$P\left(s_{t|\pi}\neq s_{t|\hat{\pi}_\beta}\right)\leq 1-(1-\alpha)^t\tag{29}$$

Only the $s_{t|\pi}\neq s_{t|\hat{\pi}_\beta}$ situation needs to be considered, since the subtraction of equivalent part is zero:

$$\begin{aligned}&\left|\sum_sP(s_t=s|\pi)\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]-\sum_sP(s_t=s|\hat{\pi}_\beta)\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]\right|\\=&P\left(s_{t|\pi}\neq s_{t|\hat{\pi}_\beta}\right)\left|\sum_sP(s_t=s|\pi)\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]-\sum_sP(s_t=s|\hat{\pi}_\beta)\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]\right|_{s_{t|\pi}\neq s_{t|\hat{\pi}_\beta}}\\\leq&\left(1-(1-\alpha)^t\right)\left[\sum_sP(s_t=s|\pi)\left|\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right|+\sum_sP(s_t=s|\hat{\pi}_\beta)\left|\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right|\right]\\\leq&\left(1-(1-\alpha)^t\right)\left[\sum_sP(s_t=s|\pi)+\sum_sP(s_t=s|\hat{\pi}_\beta)\right]\cdot 2\alpha\max_{s,a}\left|A_{\hat{\pi}_\beta}(s,a)\right|\\=&4\alpha\left(1-(1-\alpha)^t\right)\max_{s,a}\left|A_{\hat{\pi}_\beta}(s,a)\right|\end{aligned}\tag{30}$$

Then we can get the final bound:

$$
\begin{aligned}
&\left|\mathbb{E}_{s\sim\rho_{\pi}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]-\mathbb{E}_{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]\right| \\
=&\sum_{t=0}^{H}\gamma^t\Big|\sum_s P\left(s_t=s|\pi\right)\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]-\sum_s P\left(s_t=s|\hat{\pi}_\beta\right)\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]\Big| \\
\leq&\sum_{t=0}^{H}\gamma^t\cdot 4\alpha\left(1-(1-\alpha)^t\right)\max_{s,a}\left|A_{\hat{\pi}_\beta}(s,a)\right| \\
=&4\alpha\max_{s,a}\left|A_{\hat{\pi}_\beta}(s,a)\right|\sum_{t=0}^{H}\gamma^t\left(1-(1-\alpha)^t\right) \\
=&4\alpha\max_{s,a}\left|A_{\hat{\pi}_\beta}(s,a)\right|\left(\frac{1-\gamma^H}{1-\gamma}-\frac{1-\gamma^H(1-\alpha)^H}{1-\gamma(1-\alpha)}\right) \\
\leq&4\alpha\max_{s,a}\left|A_{\hat{\pi}_\beta}(s,a)\right|\left(\frac{1}{1-\gamma}-\frac{1}{1-\gamma(1-\alpha)}\right) \\
=&4\alpha\max_{s,a}\left|A_{\hat{\pi}_\beta}(s,a)\right|\frac{\gamma\alpha}{(1-\gamma)(1-\gamma(1-\alpha))} \\
\leq&\frac{4\alpha^2\gamma\max_{s,a}\left|A_{\hat{\pi}_\beta}(s,a)\right|}{(1-\gamma)^2}
\end{aligned}
\tag{31}
$$

Lemma 3. *Based on the assumption 1 that $(\hat{\pi}_\beta,\mathcal{D})$ are ξ-coupled, then we get*

$$
\left|\mathbb{E}_{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]-\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]\right|\leq\frac{4\xi^2\gamma\max_{s,a}\left|A_{\hat{\pi}_\beta}(s,a)\right|}{(1-\gamma)(1-\gamma(1-\xi))}
\tag{32}
$$

Proof. The whole proof process is the same as lemma 2 and the only difference is that the final estimation of (31) is unnecessary.

Given above preparation, we are able to derive the bound of $\left|J_\Delta(\pi,\hat{\pi}_\beta)-\widehat{J}_\Delta(\pi,\hat{\pi}_\beta)\right|$:

$$
\begin{aligned}
&\left|J_\Delta(\pi,\hat{\pi}_\beta)-\widehat{J}_\Delta(\pi,\hat{\pi}_\beta)\right|=\left|\mathbb{E}_{s\sim\rho_{\pi}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]-\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]\right| \\
=&\left|\left(\mathbb{E}_{s\sim\rho_{\pi}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]-\mathbb{E}_{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]\right)-\left(\mathbb{E}_{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]-\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]\right)\right| \\
\leq&\frac{4\alpha^2\gamma\max_{s,a}\left|A_{\hat{\pi}_\beta}(s,a)\right|}{(1-\gamma)^2}+\frac{4\xi^2\gamma\max_{s,a}\left|A_{\hat{\pi}_\beta}(s,a)\right|}{(1-\gamma)(1-\gamma(1-\xi))}
\end{aligned}
\tag{33}
$$

The α can be replaced by the total variation divergence of two policy. Suppose that p_X and p_Y are distributions with $D_{TV}(p_X\|p_Y)=\alpha$. Then there exists a joint distribution (X,Y) whose marginals are p_X,p_Y, for which $X=Y$ with probability $1-\alpha$ (Levin et al., 2008). If there are two policies π and $\hat{\pi}_\beta$ such that $\max_s D_{TV}(\pi(\cdot\mid s)\|\hat{\pi}_\beta(\cdot\mid s))\leq\alpha$, then we can define an α-coupled policy pair $(\pi,\hat{\pi}_\beta)$ with appropriate marginals. Taking $\alpha=\max_s D_{TV}(\pi(\cdot\mid s)\|\hat{\pi}_\beta(\cdot\mid s))$, theorem 2 follows.

C Proof of Theorem 3

Let π,π_k be α_1-coupled policies, then the results of lemma 1,2 can be directly generalized:

$$
\left|\mathbb{E}_{s\sim\rho_{\pi}(\cdot)}\left[\bar{A}_{\pi,\pi_k}(s)\right]-\mathbb{E}_{s\sim\rho_{\pi_k}(\cdot)}\left[\bar{A}_{\pi,\pi_k}(s)\right]\right|\leq\frac{4\alpha_1^2\gamma\max_{s,a}\left|A_{\pi_k}(s,a)\right|}{(1-\gamma)^2}
\tag{34}
$$

Similarly, given α_0-coupled policies $\pi_k,\hat{\pi}_\beta$, we get

$$
\left|\mathbb{E}_{s\sim\rho_{\pi_k}(\cdot)}\left[\bar{A}_{\pi,\pi_k}(s)\right]-\mathbb{E}_{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}\left[\bar{A}_{\pi,\pi_k}(s)\right]\right|\leq\frac{4\alpha_0^2\gamma\max_{s,a}\left|A_{\pi_k}(s,a)\right|}{(1-\gamma)(1-\gamma(1-\alpha_0))}
\tag{35}
$$

Now we combine the above two inequalities, definition (20) and lemma 3 to prove the last bound in theorem 3:

$$\begin{aligned}
&\left|J_{\Delta}\left(\pi,\pi_k\right)-\widehat{J}_{\Delta}\left(\pi,\pi_k\right)\right|=\left|\mathbb{E}_{s\sim\rho_{\pi}(\cdot),a\sim\pi(\cdot|s)}\left[A_{\pi_k}(s,a)\right]-\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\left[A_{\pi_k}(s,a)\right]\right| \\
=&\Bigg|\left(\mathbb{E}_{s\sim\rho_{\pi}(\cdot)}\left[\bar{A}_{\pi,\pi_k}(s)\right]-\mathbb{E}_{s\sim\rho_{\pi_k}(\cdot)}\left[\bar{A}_{\pi,\pi_k}(s)\right]\right)+\left(\mathbb{E}_{s\sim\rho_{\pi_k}(\cdot)}\left[\bar{A}_{\pi,\pi_k}(s)\right]-\mathbb{E}_{s\sim\rho_{\pi_\beta}(\cdot)}\left[\bar{A}_{\pi,\pi_k}(s)\right]\right) \\
&+\left(\mathbb{E}_{s\sim\rho_{\pi_\beta}(\cdot)}\left[\bar{A}_{\pi,\pi_k}(s)\right]-\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot)}\left[\bar{A}_{\pi,\pi_k}(s)\right]\right)\Bigg| \\
\leq&\frac{4\alpha_1^2\gamma\max_{s,a}\left|A_{\pi_k}(s,a)\right|}{(1-\gamma)^2}+\frac{4\alpha_0^2\gamma\max_{s,a}\left|A_{\pi_k}(s,a)\right|}{(1-\gamma)\left(1-\gamma\left(1-\alpha_0\right)\right)}+\frac{4\xi^2\gamma\max_{s,a}\left|A_{\pi_k}(s,a)\right|}{(1-\gamma)\left(1-\gamma\left(1-\xi\right)\right)}
\end{aligned} \tag{36}$$

The α_0,α_1 can be replaced by the total variation divergence in the same way, where $\alpha_0=\max_s D_{TV}(\pi_k(\cdot\mid s)\|\hat{\pi}_\beta(\cdot\mid s))$ and $\alpha_1=\max_s D_{TV}(\pi(\cdot\mid s)\|\pi_k(\cdot\mid s))$.

D Proof of Theorem 4

Lemma 4. *Let π_k, $\hat{\pi}_\beta$ be α_0-coupled policies, that is, $P\left(a_k\neq a_{\hat{\beta}}|s\right)=P\left(\pi_k\left(s\right)\neq\hat{\pi}_\beta\left(s\right)\right)\leq\alpha_0$ holds for all state s. Assume the reward function satisfies $|r\left(s,a\right)|\leq R_{max}$ for all s,a. Then*

$$\begin{aligned}
&\left|\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\left[A_{\pi_k}\left(s,a\right)\right]-\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}\left(s,a\right)\right]\right| \\
\leq&2R_{max}\frac{1+\left(1-\alpha_0\right)\left(\gamma^2-\gamma\right)}{\left(1-\gamma\right)\left(1-\gamma\left(1-\alpha_0\right)\right)}\left(\gamma+1\right)
\end{aligned} \tag{37}$$

Proof. First note that $A_\pi(s,a)=\mathbb{E}_{s'\sim p(s'|s,a)}\left[r(s,a)+\gamma V_\pi\left(s'\right)-V_\pi(s)\right]$. Then we have

$$\begin{aligned}
&\left|\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\left[A_{\pi_k}\left(s,a\right)\right]-\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}\left(s,a\right)\right]\right| \\
=&\left|\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\mathbb{E}_{s'\sim p(s'|s,a)}\left[\gamma\left(V_{\pi_k}\left(s'\right)-V_{\hat{\pi}_\beta}\left(s'\right)\right)-\left(V_{\pi_k}\left(s\right)-V_{\hat{\pi}_\beta}\left(s\right)\right)\right]\right|
\end{aligned} \tag{38}$$

Value function is the expectation over the trajectory, we decompose it into the sum of each timestep:

$$\begin{aligned}
V_\pi\left(s\right)=&\mathbb{E}_{\tau\sim P_\pi(\tau|s)}\left[\sum_{t=0}^{H}\gamma^t r(s_t,a_t)|s_0=s\right] \\
=&P\left(s_0=s\right)\mathbb{E}_{a\sim\pi(\cdot|s_0)}r\left(s_0,a\right)+\sum_{t=1}^{H}\sum_{s}\gamma^t P\left(s_t=s|\pi\right)\mathbb{E}_{a\sim\pi(\cdot|s_t)}r\left(s_t,a\right) \\
=&1\cdot\mathbb{E}_{a\sim\pi(\cdot|s_0)}r\left(s_0,a\right)+\sum_{t=1}^{H}\gamma^t\sum_{s}P\left(s_t=s|\pi\right)\mathbb{E}_{a\sim\pi(\cdot|s_t)}r\left(s_t,a\right)
\end{aligned} \tag{39}$$

Since the reward function satisfies $|r\left(s,a\right)|\leq R_{max}$ for all s,a. Then

$$|\bar{r}_\pi\left(s\right)|=\left|\mathbb{E}_{a\sim\pi(\cdot|s)}r\left(s,a\right)\right|\leq\mathbb{E}_{a\sim\pi(\cdot|s)}\left|r\left(s,a\right)\right|\leq\mathbb{E}_{a\sim\pi(\cdot|s)}R_{max}=R_{max} \tag{40}$$

For value function of policy π_k and $\hat{\pi}_\beta$, their initial states are identical, $s_{0|\pi_k}=s_{0|\hat{\pi}_\beta}$. For the state s_t, if $s_{t|\pi}=s_{t|\hat{\pi}_\beta}$ holds, when and only when all the actions taken before timestep t are identical. Given the definition of α_0-coupled, the probability of $s_{t|\pi_k}=s_{t|\hat{\pi}_\beta}$ is:

$$P\left(s_{t|\pi_k}=s_{t|\hat{\pi}_\beta}\right)\geq\left(1-\alpha_0\right)^t \tag{41}$$

The probability of $s_{t|\pi}\neq s_{t|\hat{\pi}_\beta}$ is the complementary set:

$$P\left(s_{t|\pi_k}\neq s_{t|\hat{\pi}_\beta}\right)\leq 1-\left(1-\alpha_0\right)^t \tag{42}$$

Therefore, we can get

$$
\begin{aligned}
&\left|\sum_s P\left(s_t=s|\pi_k\right)\mathbb{E}_{a\sim\pi_k(\cdot|s_t)}r\left(s_t,a\right)-\sum_s P\left(s_t=s|\hat{\pi}_\beta\right)\mathbb{E}_{a\sim\hat{\pi}_\beta(\cdot|s_t)}r\left(s_t,a\right)\right| \\
=&P\left(s_{t|\pi_k}\neq s_{t|\hat{\pi}_\beta}\right)\left|\sum_s P\left(s_t=s|\pi_k\right)\bar{r}_{\pi_k}\left(s\right)-\sum_s P\left(s_t=s|\hat{\pi}_\beta\right)\bar{r}_{\hat{\pi}_\beta}\left(s\right)\right|_{s_{t|\pi_k}\neq s_{t|\hat{\pi}_\beta}} \\
\leq&\left(1-\left(1-\alpha_0\right)^t\right)\left[\sum_s P\left(s_t=s|\pi_k\right)\left|\bar{r}_{\pi_k}\left(s\right)\right|+\sum_s P\left(s_t=s|\hat{\pi}_\beta\right)\left|\bar{r}_{\hat{\pi}_\beta}\left(s\right)\right|\right] \\
\leq&\left(1-\left(1-\alpha_0\right)^t\right)\left[\sum_s P\left(s_t=s|\pi_k\right)R_{max}+\sum_s P\left(s_t=s|\hat{\pi}_\beta\right)R_{max}\right] \\
=&2\left(1-\left(1-\alpha_0\right)^t\right)R_{max}
\end{aligned}
\tag{43}
$$

Now we are ready to measure the difference of two value functions:

$$
\begin{aligned}
&\left|V_{\pi_k}\left(s\right)-V_{\hat{\pi}_\beta}\left(s\right)\right| \\
=&\left|1\cdot\left(\bar{r}_{\pi_k}\left(s_0\right)-\bar{r}_{\hat{\pi}_\beta}\left(s_0\right)\right)+\sum_{t=1}^{H}\gamma^t\left(\sum_s P\left(s_t=s|\pi_k\right)\bar{r}\left(s_t\right)-\sum_s P\left(s_t=s|\hat{\pi}_\beta\right)\bar{r}\left(s_t\right)\right)\right| \\
\leq&\left|\bar{r}_{\pi_k}\left(s_0\right)-\bar{r}_{\hat{\pi}_\beta}\left(s_0\right)\right|+\sum_{t=1}^{H}\gamma^t\left|\sum_s P\left(s_t=s|\pi_k\right)\bar{r}_{\pi_k}\left(s_t\right)-\sum_s P\left(s_t=s|\hat{\pi}_\beta\right)\bar{r}_{\hat{\pi}_\beta}\left(s_t\right)\right| \\
\leq&\left|\mathbb{E}_{a\sim\pi_k(\cdot|s_0)}r\left(s_0,a\right)-\mathbb{E}_{a\sim\hat{\pi}_\beta(\cdot|s_0)}r\left(s_0,a\right)\right|+\sum_{t=1}^{H}\gamma^t\left[2\left(1-\left(1-\alpha_0\right)^t\right)R_{max}\right] \\
\leq&P\left(\pi_k\left(s\right)\neq\hat{\pi}_\beta\left(s\right)\right)\left|\mathbb{E}_{a\sim\pi_k(\cdot|s_0)}r\left(s_0,a\right)-\mathbb{E}_{a\sim\hat{\pi}_\beta(\cdot|s_0)}r\left(s_0,a\right)\right|_{\pi_k(s)\neq\hat{\pi}_\beta(s)}+2R_{max}\sum_{t=1}^{H}\gamma^t\left[\left(1-\left(1-\alpha_0\right)^t\right)\right] \\
\leq&\alpha_0\cdot 2R_{max}+2R_{max}\left(\frac{\gamma-\gamma^{H-1}}{1-\gamma}-\frac{\gamma\left(1-\alpha_0\right)-\gamma^{H-1}\left(1-\alpha_0\right)^{H-1}}{1-\gamma\left(1-\alpha_0\right)}\right) \\
\leq&2R_{max}\alpha_0+2R_{max}\left(\frac{\gamma}{1-\gamma}-\frac{\gamma\left(1-\alpha_0\right)}{1-\gamma\left(1-\alpha_0\right)}\right) \\
=&2R_{max}\frac{\left(\gamma^2-\gamma\right)\left(1-\alpha_0\right)+1}{\left(1-\gamma\right)\left(1-\gamma\left(1-\alpha_0\right)\right)}\alpha_0
\end{aligned}
\tag{44}
$$

Furthermore, the final bound is available:

$$
\begin{aligned}
&\left|\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\left[A_{\pi_k}\left(s,a\right)\right]-\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}\left(s,a\right)\right]\right| \\
\leq&\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\mathbb{E}_{s'\sim p(s'|s,a)}\left[\gamma\left|V_{\pi_k}\left(s'\right)-V_{\hat{\pi}_\beta}\left(s'\right)\right|+\left|V_{\pi_k}\left(s\right)-V_{\hat{\pi}_\beta}\left(s\right)\right|\right] \\
\leq&\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\mathbb{E}_{s'\sim p(s'|s,a)}\left[2R_{max}\frac{\left(\gamma^2-\gamma\right)\left(1-\alpha_0\right)+1}{\left(1-\gamma\right)\left(1-\gamma\left(1-\alpha_0\right)\right)}\left(\gamma+1\right)\alpha_0\right] \\
=&2R_{max}\frac{\left(\gamma^2-\gamma\right)\left(1-\alpha_0\right)+1}{\left(1-\gamma\right)\left(1-\gamma\left(1-\alpha_0\right)\right)}\left(\gamma+1\right)\alpha_0
\end{aligned}
\tag{45}
$$

E Implementation and Experiment Details

Following the online PPO method, we use the tricks called 'code-level optimization' including learning rate decay, orthogonal initialization, and normalization of the advantage in each mini-batch, which are considered very important to the success of the online PPO algorithm (Engstrom et al., 2020). For the implementation details, we use the annealing learning rate $10^{-4}\times 0.96^k$ (i.e.,

decaying in each interval step in the first 200 gradient steps and then remaining the learning rate) for all environments of Gym and the cloned tasks of Adroit. For Kitchen and the human tasks of Adroit, we use the initial learning rate 10^{-5} with decay before-mentioned which performs best for those tasks. We use 2 layer MLP with 1024 hidden units for the Q and policy networks, and use 3 layer MLP with 512 hidden units for value function V. We run 2×10^6 steps for fitting value Q and V functions respectively and run 5×5^{10} steps for π_β cloning. After that, we run 1,000 steps for policy improvement in the PPO setting. We clip the concatenated gradient of all parameters such that the 'global L2 norm' does not exceed 0.5. Our method is constructed by Pytorch (Paszke et al., 2019).