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), which augments 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 expects the agent can infer good policy from the dataset. Behavior cloning (BC) (Pomerleau, 1988), an approach of imitation learning, can directly imitate the action 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, also the collection process of behavior policy π_β. In the rest of this paper, improving behavior policy actually refers to improving the estimated behavior policy $\hat{\pi}_\beta$, because π_β is unknown.

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. We 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 $\hat{\pi}_\beta$ from behavior cloning (1) 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 $\rho_{\hat{\pi}_\beta}(s)$. But in offline setting, $\rho_{\hat{\pi}_\beta}(s)$ cannot be obtained through interaction with the environment like online situation. We use the state distribution recovered by offline dataset $\rho_{\mathcal{D}}(s)$ for replacement, where $\rho_{\mathcal{D}}(s) = \sum_{t=0}^{H}\gamma^t P\left(s_t = s|\mathcal{D}\right)$ and $P\left(s_t = s|\mathcal{D}\right)$ represents the probability of the t-th state equals to s in offline dataset. Therefore, the approximation of $J_\Delta\left(\pi, \pi_\beta\right)$ is derived as:

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

To measure the difference between $J_\Delta\left(\pi, \hat{\pi}_\beta\right)$ and its approximation, we introduce the midterm $\mathbb{E}_{s\sim\rho_{\hat{\pi}_\beta}(s), a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)\right]$ with state distribution $\rho_{\hat{\pi}_\beta}(s)$. During the proof, the commonly-used total variational divergence $D_{TV}\left(\pi\|\hat{\pi}_\beta\right)[s] = \frac{1}{2}\mathbb{E}_a\left|\pi\left(a|s\right) - \hat{\pi}_\beta\left(a|s\right)\right|$ between policy $\pi, \hat{\pi}_\beta$ at state s is necessary. However, the total variational divergence between offline dataset $\mathcal{D}$ and the estimated behavior policy $\hat{\pi}_\beta$ may not be straightforward:

Proposition 1. *For offline dataset $\mathcal{D} = \left\{\left(s_t, a_t, s_{t+1}, r_t\right)_{t=1}^{N}\right\}$ and policy $\hat{\pi}_\beta$, the total variational divergence can be expressed as $D_{TV}\left(\mathcal{D}\|\hat{\pi}_\beta\right)[s_t] = \frac{1}{2}\left(1 - \hat{\pi}_\beta\left(a_t|s_t\right)\right)$.*

Detailed derivation process is presented in Appendix B. Now we are ready to measure the difference:

Theorem 2. *Given the distance $D_{TV}\left(\pi\|\hat{\pi}_\beta\right)[s]$ and $D_{TV}\left(\mathcal{D}\|\hat{\pi}_\beta\right)[s] = \frac{1}{2}\left(1 - \hat{\pi}_\beta\left(a_t|s_t\right)\right)$, we can derive the following bound:*

$$\begin{aligned} J_\Delta\left(\pi, \hat{\pi}_\beta\right) \geq \widehat{J}_\Delta\left(\pi, \hat{\pi}_\beta\right) &- 4\gamma\mathbb{A}_{\hat{\pi}_\beta}\cdot\max_s D_{TV}\left(\pi\|\hat{\pi}_\beta\right)[s]\cdot\mathop{\mathrm{E}}_{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}\left[D_{TV}\left(\pi\|\hat{\pi}_\beta\right)[s]\right] \\ &- 2\gamma\mathbb{A}_{\hat{\pi}_\beta}\cdot\max_s D_{TV}\left(\pi\|\hat{\pi}_\beta\right)[s]\cdot\mathop{\mathrm{E}}_{s\sim\rho_{\mathcal{D}}(\cdot)}\left[1 - \hat{\pi}_\beta\left(a|s\right)\right], \end{aligned} \tag{5}$$

here $\mathbb{A}_{\hat{\pi}_\beta} = \max_{s,a}\left|A_{\hat{\pi}_\beta}(s,a)\right|$. The proof is presented in Appendix C.

Compared to the theorem in online setting (Schulman et al., 2015a; Achiam et al., 2017; Queeney et al., 2021), the second right term of Equation (5) is similar while the third term is unique for the offline. $\mathop{\mathrm{E}}_{s\sim\rho_{\mathcal{D}}(\cdot)}\left[1 - \hat{\pi}_\beta\left(a|s\right)\right]$ represents the difference caused by the mismatch between offline dataset $\mathcal{D}$ and $\hat{\pi}_\beta$. When $\hat{\pi}_\beta$ is determined, this term is one constant. And because the inequality $\max_s D_{TV}\left(\pi\|\hat{\pi}_\beta\right)[s] \geq \mathop{\mathrm{E}}_{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}\left[D_{TV}\left(\pi\|\hat{\pi}_\beta\right)[s]\right]$ holds, we can claim the following conclusion:

Conclusion1

To guarantee the true objective $J_\Delta\left(\pi, \hat{\pi}_\beta\right)$ non-decreasing, we should simultaneously maximize $\mathbb{E}_{s\sim\rho_{\mathcal{D}(\cdot)}, a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)\right]$ and minimize $\left[\max_s D_{TV}\left(\pi\|\hat{\pi}_\beta\right)[s]\right]$, which means offline dataset $\mathcal{D}$ is really able to monotonically improve the estimated behavior policy $\hat{\pi}_\beta$.

Suppose we have improved the behavior policy $\hat{\pi}_\beta$ and get one policy π_k with higher performance. The above theorem only guarantees that 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 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, common manner is replacing the ρ_π with old policy state distribution ρ_{π_k}. But 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 any other alternative is replacing ρ_{π_k} by the state distribution from offline dataset $\mathcal{D}$:

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

Intuitively, this replacement is reasonable if policy $\pi_k, \hat{\pi}_\beta$ are similar which means this approximation must be related to the distance $D_{TV}(\pi_k \| \hat{\pi}_\beta)[s]$. Concretely, the gap can be formulated as follows:

Theorem 3. *Given the distance $D_{TV}(\pi\|\pi_k)[s]$, $D_{TV}(\pi_k\|\hat{\pi}_\beta)[s]$ and $D_{TV}(\mathcal{D}\|\hat{\pi}_\beta)[s] = \frac{1}{2}(1-\hat{\pi}_\beta(a|s))$, we can derive the following bound:*

$$\begin{aligned} J_\Delta(\pi,\pi_k) \geq \widehat{J}_\Delta(\pi,\pi_k) &- 4\gamma\mathbb{A}_{\pi_k}\cdot\max_s D_{TV}(\pi\|\pi_k)[s]\cdot\mathop{\mathrm{E}}_{s\sim\rho_{\pi_k}(\cdot)}\left[D_{TV}(\pi\|\pi_k)[s]\right] \\ &- 4\gamma\mathbb{A}_{\pi_k}\cdot\max_s D_{TV}(\pi\|\pi_k)[s]\cdot\mathop{\mathrm{E}}_{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}\left[D_{TV}(\pi_k\|\hat{\pi}_\beta)[s]\right] \\ &- 2\gamma\mathbb{A}_{\pi_k}\cdot\max_s D_{TV}(\pi\|\pi_k)[s]\cdot\mathop{\mathrm{E}}_{s\sim\rho_{\mathcal{D}}(\cdot)}\left[1-\hat{\pi}_\beta(a|s)\right], \end{aligned} \tag{8}$$

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

Compared to the theorem 2, one additional term related to the distance of $\pi_k, \hat{\pi}_\beta$ has been introduced. The distance $\mathop{\mathrm{E}}_{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}\left[D_{TV}(\pi_k\|\hat{\pi}_\beta)[s]\right]$ is irrelevant to the target policy π which can also be viewed as one constant. Besides, theorem 2 is a specific case of this theorem if $\pi_k = \hat{\pi}_\beta$. Thus, we set $\boxed{\pi_0 = \hat{\pi}_\beta}$ since $\hat{\pi}_\beta$ is the first policy to be improved and in the following section we will no longer deliberately distinguish $\hat{\pi}_\beta, \pi_k$. Similarly, we can derive the following conclusion:

Conclusion2

To guarantee the true objective $J_\Delta(\pi,\pi_k)$ non-decreasing, we should simultaneously maximize $\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot), a\sim\pi(\cdot|s)}\left[A_{\pi_k}(s,a)\right]$ and minimize $\left[\max_s D_{TV}(\pi\|\pi_k)[s]\right]$, which means offline dataset $\mathcal{D}$ is still able to monotonically improve the policy π_k, where $k = 0, 1, 2, \cdots$.

4 Behavior Proximal Policy Optimization

In this section, we first discuss how to estimate the advantage $A_{\pi_k}(s,a)$ and then derive one practical algorithm based on the theoretical results. And surprisingly, the loss function of this algorithm is the same as the online on-policy method Proximal Policy Optimization (PPO) (Schulman et al., 2017). Besides, we directly explain why the loss of PPO can solve offline reinforcement learning naturally.

Advantage Approximation In traditional online situation, advantage $A_{\pi_k}(s,a)$ is estimated by Generalized Advantage Estimation (GAE) (Schulman et al., 2015b) using the data collected by policy π_k. But in offline RL, only offline dataset $\mathcal{D} = \left\{(s_t, a_t, s_{t+1}, r_t)_{t=1}^N\right\}$ from true behavior policy π_β is available. The advantage of (s_t, a_t) calculated by GAE is as follow:

$$A_{\pi_\beta}(s_t,a_t) = \sum_{l=0}^{\infty}(\gamma\lambda)^l\left(r_{t+l} + \gamma V_{\pi_\beta}(s_{t+l+1}) - V_{\pi_\beta}(s_{t+l})\right). \tag{9}$$

GAE can only calculate the advantage of $(s_t, a_t) \in \mathcal{D}$. For $(s_t, \tilde{a}_t) \sim \mathcal{D}$, where $\tilde{a}_t$ is an in-distribution action sampling but $(s_t, \tilde{a}_t) \notin \mathcal{D}$, GAE is **unable** to give any estimation. This is because the calculation process of GAE depends on the trajectory and does not have the ability to generalize to unseen state-action pairs. Therefore, GAE is not a satisfactory choice for offline RL. Offline RL forbids the interaction with environment, so data usage should be more efficient. Concretely, we expect advantage approximation method can not only calculate the advantage of (s_t, a_t), but also $(s_t, \tilde{a}_t)$. As a result, we directly estimate advantage with the definition $A_{\pi_\beta}(s,a) = Q_{\pi_\beta}(s,a) - V_{\pi_\beta}(s)$, 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. This function approximation method can generalize to the advantage of $(s_t, \tilde{a}_t)$.

However, strictly speaking, $A_{\pi_\beta} \neq A_{\pi_k}$ even for $k = 0$. Although we can estimate A_{π_k} by off-policy method Q-learning, the overestimation risk on out-of-distribution actions is unavoidable. Therefore, we choose to replace all A_{π_k} with trustworthy A_{π_β} then theoretically measure the difference rather than empirically make A_{π_k} learned by Q-learning more accurate. The difference caused by replacing the A_{π_k} in $\widehat{J}_\Delta(\pi, \pi_k)$ with $A_{\pi_\beta}(s, a)$ can be measured in the following theorem:

Theorem 4. *Given the distance $D_{TV}(\pi_k \| \pi_\beta)[s]$ and assume the reward function satisfies $|r(s,a)| \leq \mathbf{R}$ for all s, a, then*

$$\left| \widehat{J}_\Delta(\pi, \pi_k) - \mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot), a\sim\pi(\cdot|s)} \left[A_{\pi_\beta}(s,a) \right] \right| \leq 2\gamma(\gamma+1)\mathbf{R} \underset{s\sim\rho_{\pi_\beta}(\cdot)}{\mathrm{E}} \left[D_{TV}(\pi_k \| \pi_\beta)[s] \right]. \quad (10)$$

The proof is presented in Appendix E.

Note that the right end term of the equation is irrelevant to the policy π and can be viewed as a constant when optimizing π. Combining the result of theorem 3 and 4, we get the following corollary:

corollary 1. *Given the distance $D_{TV}(\pi \| \pi_k)[s]$, $D_{TV}(\pi_k \| \hat{\pi}_\beta)[s]$ and $D_{TV}(\mathcal{D} \| \hat{\pi}_\beta)[s] = \frac{1}{2}(1 - \hat{\pi}_\beta(a|s))$, we can derive the following bound:*

$$\begin{aligned} J_\Delta(\pi, \pi_k) \geq\ & \mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot), a\sim\pi(\cdot|s)} \left[A_{\pi_\beta}(s,a) \right] \\ & - 4\gamma \mathbb{A}_{\pi_k} \cdot \max_s D_{TV}(\pi \| \pi_k)[s] \cdot \underset{s\sim\rho_{\pi_k}(\cdot)}{\mathrm{E}} \left[D_{TV}(\pi \| \pi_k)[s] \right] \\ & - 4\gamma \mathbb{A}_{\pi_k} \cdot \max_s D_{TV}(\pi \| \pi_k)[s] \cdot \underset{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}{\mathrm{E}} \left[D_{TV}(\pi_k \| \hat{\pi}_\beta)[s] \right] \\ & - 2\gamma \mathbb{A}_{\pi_k} \cdot \max_s D_{TV}(\pi \| \pi_k)[s] \cdot \underset{s\sim\rho_{\mathcal{D}}(\cdot)}{\mathrm{E}} \left[1 - \hat{\pi}_\beta(a|s) \right] - \boxed{\mathcal{C}_{\pi_k, \pi_\beta}}, \end{aligned} \quad (11)$$

where $\mathbb{A}_{\pi_k} = \max_{s,a} |A_{\pi_k}(s,a)|$ and $\mathcal{C}_{\pi_k, \pi_\beta} = 2\gamma(\gamma+1) \cdot \max_{s,a} |r(s,a)| \underset{s\sim\rho_{\pi_\beta}(\cdot)}{\mathrm{E}} \left[D_{TV}(\pi_k \| \pi_\beta)[s] \right]$.

Conclusion3

To guarantee the true objective $J_\Delta(\pi, \pi_k)$ non-decreasing, we can also simultaneously maximize $\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot), a\sim\pi(\cdot|s)} \left[A_{\pi_\beta}(s,a) \right]$ and minimize $[\max_s D_{TV}(\pi \| \pi_k)[s]]$, $k = 0, 1, 2, \cdots$.

Loss Derivation According to the above **Conclusion3**, to monotonically improve policy π_k, we should jointly optimize:

$$\underset{\pi}{\textbf{Maximize}}\ \mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot), a\sim\pi(\cdot|s)} \left[A_{\pi_\beta}(s,a) \right] \ \ \& \ \ \underset{\pi}{\textbf{Minimize}}\ \max_s D_{TV}(\pi \| \pi_k)[s], \quad (12)$$

here $k = 0, 1, 2, \cdots$ and $\pi_0 = \hat{\pi}_\beta$. But minimizing the total divergence between π and π_k results in a trivial solution $\pi = \pi_k$ which is impossible to make improvement over π_k. A more reasonable optimization objective is to maximize $\widehat{J}_\Delta(\pi, \pi_k)$ while constraining the divergence:

$$\underset{\pi}{\textbf{Maximize}}\ \mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot), a\sim\pi(\cdot|s)} \left[A_{\pi_\beta}(s,a) \right] \ \text{s.t.}\ \max_s D_{TV}(\pi \| \pi_k)[s] \leq \epsilon. \quad (13)$$

For the term to be maximized, it may be confusing that this term does not contain policy π_k. But actually, we should adopt importance sampling to make the expectation only depend on the action distribution of old policy π_k rather than new policy π:

$$\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot), a\sim\pi(\cdot|s)} \left[A_{\pi_\beta}(s,a) \right] = \mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot), a\sim\pi_k(\cdot|s)} \left[\frac{\pi(a|s)}{\pi_k(a|s)} A_{\pi_\beta}(s,a) \right]. \quad (14)$$

In this way, this term is allowed to estimate by sampling state from offline dataset $s \sim \rho_{\mathcal{D}}(\cdot)$ then sampling action with old policy $a \sim \pi(\cdot|s)$. For the total variational divergence, we rewrite it as

$$\begin{aligned} & \max_s D_{TV}(\pi \| \pi_k)[s] = \max_s \frac{1}{2} \int_a |\pi(a|s) - \pi_k(a|s)| \, \mathrm{d}a \\ = & \max_s \frac{1}{2} \int_a \pi_k(a|s) \left| \frac{\pi(a|s)}{\pi_k(a|s)} - 1 \right| \mathrm{d}a = \max_s \underset{a\sim\pi_k(\cdot|s)}{\mathbb{E}} \left| \frac{\pi(a|s)}{\pi_k(a|s)} - 1 \right|. \end{aligned} \quad (15)$$

In offline setting, only states $s \sim \rho_{\mathcal{D}}(\cdot)$ are available and other states are unable to access. So the operation $\max_s$ can also be expressed as $\max_{s\sim\rho_{\mathcal{D}}(\cdot)}$. When comparing Equation (14) and (15), we find that the state distribution, the action distribution and the policy ratio both appear. Thus we consider how to insert the divergence constraint into Equation (14). The following constraints are equivalent:

$$\max_{s\sim\rho_{\mathcal{D}}(\cdot)} D_{TV}(\pi\|\pi_k)[s] \leq \epsilon \iff \max_{s\sim\rho_{\mathcal{D}}(\cdot)} \mathop{\mathbb{E}}_{a\sim\pi_k(\cdot|s)} \left|\frac{\pi(a|s)}{\pi_k(a|s)} - 1\right| \leq \epsilon$$
$$\iff \max_{s\sim\rho_{\mathcal{D}}(\cdot)} \mathop{\mathbb{E}}_{a\sim\pi_k(\cdot|s)} \text{clip}\left(\frac{\pi(a|s)}{\pi_k(a|s)}, 1-\epsilon, 1+\epsilon\right), \text{clip}(x,l,u) = \min(\max(x,l),u). \quad (16)$$

Here the $\max$ operation is impractical to solve, so we adopt a heuristic approximation (Schulman et al., 2015a) that changes $\max$ into expectation. Then divergence constraint (16) can be inserted:

$$L_k(\pi) = \mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi_k(\cdot|s)}\left[\min\left(\frac{\pi(a|s)}{\pi_k(a|s)}A_{\pi_\beta}(s,a), \text{clip}\left(\frac{\pi(a|s)}{\pi_k(a|s)}, 1-\epsilon, 1+\epsilon\right)A_{\pi_\beta}(s,a)\right)\right], \quad (17)$$

where the operation $\min$ makes this objective become the lower bound of Equation (14). This loss function is the same as PPO (Schulman et al., 2017) and that is why we claim that *some online on-policy algorithms are naturally able to solve offline RL*. On the one hand, the loss function of our method has a similar form with PPO. On the other hand, our method improves the (estimated) behavior policy and changes all the advantage with the advantage of behavior. Based on the two above characteristics, we call our method as **B**ehavior **P**roximal **P**olicy **O**ptimization, shorted as BPPO. The whole workflow of BPPO has been summarized in Algorithm 1.

Algorithm 1 **B**ehavior **P**roximal **P**olicy **O**ptimization (BPPO)

1: Estimate behavior policy $\hat{\pi}_\beta$ by behavior cloning;
2: Calculate Q-function Q_{π_β} by SARSA;
3: Calculate value function V_{π_β} by fitting returns with MSE;
4: Compute advantage function $A_{\pi_\beta} = Q_{\pi_\beta} - V_{\pi_\beta}$;
5: Set $\pi_0 = \hat{\pi}_\beta$, i.e., $k = 0$;
6: **for** $i = 0, 1, 2, \cdots, I$ **do**
7: Update the policy π by maximizing $\hat{L}_k(\pi)$ in Equation (17);
8: **if** $J(\pi) > J(\pi_k)$ **then**
9: Set $k = k + 1$, then $\pi_k \leftarrow \pi$;
10: **end if**
11: **end for**

What is the relation between BPPO and Onestep RL? Since BPPO is proposed based on on-policy algorithm, it may be naturally associated with Onestep RL (Brandfonbrener et al., 2021) that solves offline RL without off-policy evaluation. BPPO only estimates the advantage function once but updates policy k times, which means BPPO is neither Onestep nor multi-step. If we remove lines 8-10 in Algorithm 1, it becomes an Onestep version. The right figure shows the difference between BPPO and its Onestep version. Onestep strictly requires the new policy close to $\hat{\pi}_\beta$, while BPPO appropriately loosens this restriction. To prevent the learned policy completely away from $\hat{\pi}_\beta$, we design a technique called clip ratio decay. As the policy updates, the clip ratio ϵ gradually decreases:

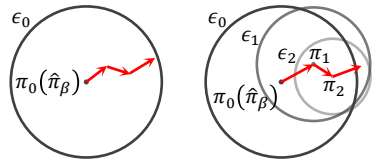


Figure 1: The difference between Onestep BPPO (left) and BPPO (right), where the decreasing circle corresponds to clip ratio decay.

$$\epsilon_i = \begin{cases} \epsilon \times (\sigma)^{i-1}, & \text{if } i \leq 200, \\ \epsilon_{200}, & \text{otherwise}, \end{cases} \quad (18)$$

here i denotes the training steps, ϵ denotes the initial clip ratio, and σ is the decay coefficient.

Why BPPO can solve offline RL? Now we try to directly explain why BPPO can solve offline RL from the perspective of overcoming overestimation. According to the final loss (17) and Equation (16), BPPO actually constrains the closeness by the expectation of the total variational divergence:

$$\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi_k(\cdot|s)}\left|\frac{\pi(a|s)}{\pi_k(a|s)} - 1\right| \leq \epsilon. \quad (19)$$

If $k = 0$, this equation ensures the closeness between learned policy and $\hat{\pi}_\beta$. When $k > 0$, the closeness between learned policy π and π_k indirectly constrains the closeness between π and $\hat{\pi}_\beta$. This inherent conservatism from online is exactly suitable for offline and we believe any other 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). Some 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 in online RL was first introduced by Kakade & Langford (2002). On this basis, two classical on-policy methods Trust Region Policy Optimization (TRPO) (Schulman et al., 2015a) and Proximal Policy Optimization (PPO) (Schulman et al., 2017) were proposed. Afterward, monotonic policy improvement has been extended in other settings such as constrained markov decision process (Achiam et al., 2017), model-based method (Luo et al., 2018) and off-policy RL (Queeney et al., 2021; Meng et al., 2021). In offline RL, Xu et al. (2021) mentions that offline algorithms lack guaranteed performance improvement over the behavior policy but do not consider offline monotonic policy improvement. Our work first introduces monotonic policy improvement to skillfully solve offline reinforcement learning.

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 questions: 1) How does BPPO compare with previous 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? To answer these, we evaluate BPPO on the Gym (v2), Adroit (v1), Antmaze (v2), and Kitchen (v0) from D4RL (Fu et al., 2020).

6.1 Results on D4RL Benchmarks

We first evaluate whether BPPO can outperform the prior iterative and Onestep offline reinforcement learning methods. We compare BPPO with iterative dynamic programming methods including CQL (Kumar et al., 2020), TD3+BC (Fujimoto & Gu, 2021), and F-BRC (Kostrikov et al., 2021a), and Onestep methods including Onestep RL (Brandfonbrener et al., 2021) and IQL (Kostrikov et al., 2021b). The results of Onestep RL, IQL, CQL, and TD3+BC are extracted from the paper IQL, while F-BRC is directly from the original article. 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 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, BPPO achieves 51% performance improvement on all D4RL tasks. Interestingly, our implemented BC on Adroit and Kitchen nearly outperform the baselines, which may imply improving behavior policy rather than learning from scratch is better.

Next, we evaluat whether BPPO can solve more difficult tasks with sparse reward. On the Antmaze benchmarks, we also compare BPPO with Decision Transformer (DT) (Chen et al., 2021), RvS-G and RvS-R (Emmons et al., 2021). DT conditions on past trajectories to predict future actions using Transformer. RvS-G and RvS-R condition on goals or rewards to learn policy via supervised learning.

Table 1: The normalized results on D4RL Gym, Adroit, and Kitchen. We **bold** the best results and BPPO is calculated by averaging 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***
Adroit	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***

As shown in Table 2, BPPO can outperform most tasks and is significantly better than other algorithms in the total performance of all tasks. Although the Filtered BC is used on the part of tasks, we only train the target policy by 100 gradient steps on less than 10% data of the original dataset used in other algorithms. It is very simple compared with training a goal- or reward-conditioned policy like RvS-G, DT, and RvS-R. The superior performance demonstrates the BPPO can also considerably improve the policy performance based on (Filtered) BC in some tasks with sparse reward.

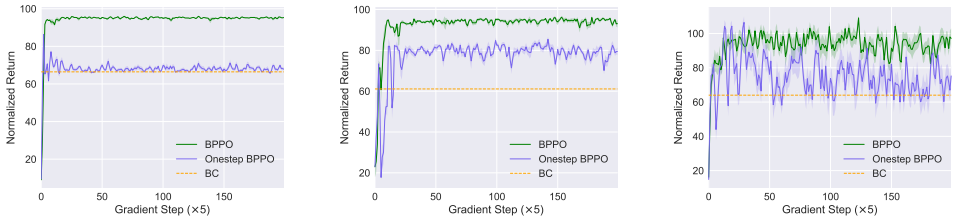
Table 2: The normalized results of all algorithms on D4RL Antmaze tasks. For a fair comparison, the results of CQL and IQL are extracted from IQL, and other results are extracted from RvS. In the BC column, we use a symbol * to specify that we adopt the Filtered BC (Emmons et al., 2021) instead of standard BC. In the Antmaze task, Filtered BC removes the failed trajectories from the offline dataset.

Environment	TD3+BC	Onestep	CQL	IQL	DT	RvS-R	RvS-G	BC (Ours)	BPPO (Ours)
Umaze-v2	78.6	64. 3	74.0	87.5	65.6	64.4	65.4	51.7±20.4	**95.0±5.5**
Umaze-diverse-v2	71.4	60.7	84.0	62.2	51.2	70.1	60.9	48.3±17.2	**91.7±4.1**
Medium-play-v2	10.6	0.3	61.2	**71.2**	1.0	4.5	58.1	16.7±5.2*	51.7±7.5
Medium-diverse-v2	3.0	0.0	53.7	**70.0**	0.6	7.7	67.3	33.3±10.3*	**70.0±6.3**
Large-play-v2	0.2	0.0	15.8	39.6	0.0	3.5	32.4	48.3±11.7*	**86.7±8.2**
Large-diverse-v2	0.0	0.0	14.9	47.5	0.2	3.7	36.9	46.7±20.7*	**88.3±4.1**
Total	*163.8*	*61.0*	*303.6*	*378.0*	*118.6*	*153.9*	*321.0*	*245.0±85.5*	***483.3±35.7***

6.2 THE SUPERIORITY OF BPPO OVER ONESTEP VERSION

We choose to improve policy π_k after it has been improved over behavior policy because theorem 2 does not ensure all information from offline dataset is extracted. But when improving policy π_k, we utilize twice approximation. Therefore, it is necessary 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 analyze the superiority of BPPO over its Onestep version empirically.

In Figure2, we observe that BPPO and Onestep BPPO can outperform the BC (the yellow dotted line). This indicates both of them can achieve 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, improving π_k to fully utilize information is necessary. Second, compared to strictly restricting the learned policy close to the behavior policy, appropriate looseness is useful. At the same time, the clip decay (18) prevents this looseness within control and Figure 6 shows this phenomenon.

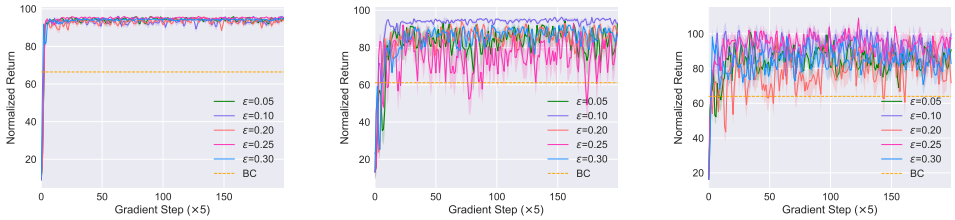


(a) hopper-medium-v2 (b) hopper-medium-replay-v2 (c) hopper-medium-expert-v2

Figure 2: The comparison between BPPO and Onestep BPPO. The hyperparameters of both methods are tuned through the grid search, and then we exhibit their learning curves with the best performance.

6.3 Ablation Study of Different Hyperparameters

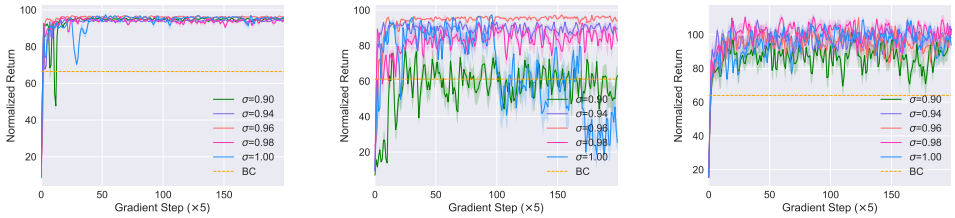
In this section, we evaluate the influence of clip ratio ϵ and its decay rate σ. clip ratio restricts the policy close to behavior policy and it directly solves the offline overestimation. Since ϵ also appears in PPO, we can set it properly to avoid catastrophic performance, which is the unique feature of BPPO. σ gradually tightens this restriction during policy improvement. We show how these coefficients contribute to the performance of BPPO, more ablations can be found in Appendix F, H, and G.



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

Figure 3: Ablation study on clip ratio 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 clip ratio decay σ. In particular, $\sigma = 1.00$ denotes no decay.

We then demonstrate how the clip ratio decay affects the performance of BPPO. We evaluate five decay rates ($\sigma = 0.90, 0.94, 0.96, 0.98, 1.00$) on BPPO. As shown in Figure 4(b), a low decay rate ($\sigma = 0.90$) or no decay ($\sigma = 1.00$) is possible to crash during training. Considering the performance of different environments, we use $\sigma = 0.96$ to achieve stable policy improvement.

7 Conclusion

Behavior **P**roximal **P**olicy **O**ptimization (BPPO) starts from offline monotonic policy improvement, using the loss function of PPO elegantly solves offline RL. This is because of the inherent conservatism from the on-policy method naturally suitable to overcome overestimation in offline RL. BPPO is very simple to implement and achieves superior performance on D4RL.

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International conference on machine learning*, pp. 22–31. PMLR, 2017.

Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-Based Offline Reinforcement Learning with Diversified Q-Ensemble. *arXiv e-prints*, art. arXiv:2110.01548, October 2021.

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.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

Xi Chen, Ali Ghadirzadeh, Tianhe Yu, Yuan Gao, Jianhao Wang, Wenzhe Li, Bin Liang, Chelsea Finn, and Chongjie Zhang. Latent-variable advantage-weighted policy optimization for offline rl. *arXiv preprint arXiv:2203.08949*, 2022.

Ching-An Cheng, Tengyang Xie, Nan Jiang, and Alekh Agarwal. Adversarially trained actor critic for offline reinforcement learning. *arXiv preprint arXiv:2202.02446*, 2022.

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.

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.

Wenjia Meng, Qian Zheng, Yue Shi, and Gang Pan. 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, 2021.

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.

James Queeney, Yannis Paschalidis, and Christos G Cassandras. Generalized proximal policy optimization with sample reuse. *Advances in Neural Information Processing Systems*, 34:11909–11919, 2021.

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.

Rui Yang, Chenjia Bai, Xiaoteng Ma, Zhaoran Wang, Chongjie Zhang, and Lei Han. Rorl: Robust offline reinforcement learning via conservative smoothing. *arXiv preprint arXiv:2206.02829*, 2022.

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,

$$
\begin{aligned}
&\mathbb{E}_{\tau\sim P_{\pi'}}\left[\sum_{t=0}^{H}\gamma^t A_\pi\left(s_t,a_t\right)\right]\\
=&\mathbb{E}_{\tau\sim P_{\pi'}}\left[\sum_{t=0}^{H}\gamma^t\left(r\left(s_t,a_t\right)+\gamma V_\pi\left(s_{t+1}\right)-V_\pi\left(s_t\right)\right)\right]\\
=&\mathbb{E}_{\tau\sim P_{\pi'}}\left[-V_\pi\left(s_0\right)+\sum_{t=0}^{H}\gamma^t r\left(s_t,a_t\right)\right]\\
=&-\mathbb{E}_{s_0}\left[V_\pi\left(s_0\right)\right]+\mathbb{E}_{\tau\sim P_{\pi'}}\left[\sum_{t=0}^{H}\gamma^t r\left(s_t,a_t\right)\right]\\
=&-J\left(\pi\right)+J\left(\pi'\right)\\
\triangleq&J_\Delta\left(\pi',\pi\right)
\end{aligned}
\tag{20}
$$

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:

$$
\begin{aligned}
&\mathbb{E}_{\tau\sim P_{\pi'}}\left[\sum_{t=0}^{H}\gamma^t A_\pi\left(s_t,a_t\right)\right]\\
=&\sum_{t=0}^{H}\sum_{s}\left[P\left(s_t=s|\pi'\right)\mathbb{E}_{a\sim\pi'(\cdot|s)}\left[\gamma^t A_\pi\left(s,a\right)\right]\right]\\
=&\sum_{s}\left[\sum_{t=0}^{H}\gamma^t P\left(s_t=s|\pi'\right)\mathbb{E}_{a\sim\pi'(\cdot|s)}\left[A_\pi\left(s,a\right)\right]\right]\\
=&\sum_{s}\left[\rho_{\pi'(s)}\mathbb{E}_{a\sim\pi'(\cdot|s)}\left[A_\pi\left(s,a\right)\right]\right]\\
=&\mathbb{E}_{s\sim\rho_{\pi'(s)},a\sim\pi'(\cdot|s)}\left[A_\pi\left(s,a\right)\right]
\end{aligned}
\tag{21}
$$

B PROOF OF PROPOSITION 1

Proposition 2. *For offline dataset* $\mathcal{D}=\left\{\left(s_t,a_t,s_{t+1},r_t\right)_{t=1}^{N}\right\}$ *and policy* $\hat{\pi}_\beta$*, the total variational divergence can be expressed as* $D_{TV}\left(\mathcal{D}\|\hat{\pi}_\beta\right)\left[s_t\right]=\frac{1}{2}\left(1-\hat{\pi}_\beta\left(a_t|s_t\right)\right)$.

Proof. For state-action pair $(s_t,a_t)\in\mathcal{D}$, it can be viewed as one deterministic policy that satisfies $\pi_{\mathcal{D}}\left(a=a_t|s_t\right)=1$ and $\pi_{\mathcal{D}}\left(a\neq a_t|s_t\right)=0$. So

$$
\begin{aligned}
&D_{TV}\left(\mathcal{D}\|\hat{\pi}_\beta\right)\left[s_t\right]=D_{TV}\left(\pi_{\mathcal{D}}\|\hat{\pi}_\beta\right)\left[s_t\right]\\
=&\frac{1}{2}\mathbb{E}_a\left|\pi_{\mathcal{D}}\left(a|s_t\right)-\hat{\pi}_\beta\left(a|s_t\right)\right|\\
=&\frac{1}{2}\int\left[P\left(a_t\right)\left|\pi_{\mathcal{D}}\left(a_t|s_t\right)-\hat{\pi}_\beta\left(a_t|s_t\right)\right|+P\left(a\neq a_t\right)\left|\pi_{\mathcal{D}}\left(a|s_t\right)-\hat{\pi}_\beta\left(a|s_t\right)\right|\right]\mathrm{d}a\\
=&\frac{1}{2}\int\left[P\left(a_t\right)\left(1-\hat{\pi}_\beta\left(a_t|s_t\right)\right)+P\left(a\neq a_t\right)\hat{\pi}_\beta\left(a\neq a_t|s_t\right)\right]\mathrm{d}a\\
=&\frac{1}{2}\int\left[P\left(a_t\right)\left(1-\hat{\pi}_\beta\left(a_t|s_t\right)\right)+\left(1-P\left(a_t\right)\right)\left(1-\hat{\pi}_\beta\left(a_t|s_t\right)\right)\right]\mathrm{d}a\\
=&\frac{1}{2}\left(1-\hat{\pi}_\beta\left(a_t|s_t\right)\right)
\end{aligned}
\tag{22}
$$

C PROOF OF THEOREM 2

The definition of $\bar{A}_{\pi,\hat{\pi}_\beta}(s)$ is as follows:

$$\bar{A}_{\pi,\hat{\pi}_\beta}(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)\right] \tag{23}$$

Note that the expectation of advantage function $A_{\hat{\pi}_\beta}(s,a)$ depends on another policy π rather than $\hat{\pi}_\beta$, so $\bar{A}_{\pi,\hat{\pi}_\beta}(s) \neq 0$. Furthermore, given the $\bar{A}_{\pi,\hat{\pi}_\beta}(s)$, the performance difference in theorem 2 can be rewritten as:

$$J_\Delta(\pi,\hat{\pi}_\beta) = \mathbb{E}_{s\sim\rho_\pi(\cdot),a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)\right] = \mathbb{E}_{s\sim\rho_\pi(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right] \tag{24}$$

$$\widehat{J}_\Delta(\pi,\hat{\pi}_\beta) = \mathbb{E}_{s\sim\rho_\mathcal{D}(\cdot),a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)\right] = \mathbb{E}_{s\sim\rho_\mathcal{D}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right] \tag{25}$$

Lemma 1. *For all state s,*

$$\left|\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right| \leq 2\max_a\left|A_{\hat{\pi}_\beta}(s,a)\right| \cdot D_{TV}(\pi\|\hat{\pi}_\beta)[s] \tag{26}$$

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

$$\mathbb{E}_{a\sim\pi}\left[A_\pi(s,a)\right] = \mathbb{E}_{a\sim\pi}\left[Q_\pi(s,a) - V_\pi(s)\right] = \mathbb{E}_{a\sim\pi}\left[Q_\pi(s,a)\right] - V_\pi(s) = 0 \tag{27}$$

Thus, with the help of Hölder's inequality, we get

$$\begin{aligned}
&\left|\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right| = \left|\mathbb{E}_{a\sim\pi(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)\right] - \mathbb{E}_{a\sim\hat{\pi}_\beta(\cdot|s)}\left[A_{\hat{\pi}_\beta}(s,a)\right]\right| \\
&\leq \left\|\pi(a|s) - \hat{\pi}_\beta(a|s)\right\|_1 \left\|A_{\hat{\pi}_\beta}(s,a)\right\|_\infty \\
&= 2D_{TV}(\pi\|\hat{\pi}_\beta)[s] \cdot \max_a\left|A_{\hat{\pi}_\beta}(s,a)\right|, \forall s
\end{aligned} \tag{28}$$

Lemma 2. ((Achiam et al., 2017)) *The divergence between two unnormalized visitation frequencies, $\|\rho_\pi(\cdot) - \rho_{\pi'}(\cdot)\|_1$, is bounded by an average total variational divergence of the policies π and π':*

$$\|\rho_\pi(\cdot) - \rho_{\pi'}(\cdot)\|_1 \leq 2\gamma \mathop{\mathrm{E}}_{s\sim\rho_{\pi'}(\cdot)}\left[D_{TV}(\pi\|\pi')[s]\right] \tag{29}$$

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

$$\begin{aligned}
&\left|J_\Delta(\pi,\hat{\pi}_\beta) - \widehat{J}_\Delta(\pi,\hat{\pi}_\beta)\right| = \left|\mathbb{E}_{s\sim\rho_\pi(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right] - \mathbb{E}_{s\sim\rho_\mathcal{D}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]\right| \\
=&\left|\left(\mathbb{E}_{s\sim\rho_\pi(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right] - \mathbb{E}_{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]\right) + \left(\mathbb{E}_{s\sim\rho_{\hat{\pi}_\beta}}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right] - \mathbb{E}_{s\sim\rho_\mathcal{D}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]\right)\right|
\end{aligned} \tag{30}$$

Based on Hölder's inequality and lemma 2, we can bound the first term as follows:

$$\begin{aligned}
&\left|\left(\mathbb{E}_{s\sim\rho_\pi(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right] - \mathbb{E}_{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}\left[\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right]\right)\right| \leq \left\|\rho_\pi(\cdot) - \rho_{\hat{\pi}_\beta}(\cdot)\right\|_1 \left\|\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right\|_\infty \\
\leq& 2\gamma \mathop{\mathrm{E}}_{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}\left[D_{TV}(\pi\|\hat{\pi}_\beta)[s]\right] \cdot \max_s\left|\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right|
\end{aligned} \tag{31}$$

For the second term, we can derive similar bound and furthermore let $D_{TV}(\mathcal{D}\|\hat{\pi}_\beta)[s] = \frac{1}{2}(1 - \hat{\pi}_\beta(a_t|s_t))$. Finally, using lemma 1, we get

$$\begin{aligned}
&\left|J_\Delta(\pi,\hat{\pi}_\beta) - \widehat{J}_\Delta(\pi,\hat{\pi}_\beta)\right| \\
\leq& 2\gamma\max_s\left|\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right|\left(\mathop{\mathrm{E}}_{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}\left[D_{TV}(\pi\|\hat{\pi}_\beta)[s]\right] + \mathop{\mathrm{E}}_{s\sim\rho_\mathcal{D}(\cdot)}\left[D_{TV}(\mathcal{D}\|\hat{\pi}_\beta)[s]\right]\right) \\
=& 2\gamma\max_s\left|\bar{A}_{\pi,\hat{\pi}_\beta}(s)\right|\left(\mathop{\mathrm{E}}_{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}\left[D_{TV}(\pi\|\hat{\pi}_\beta)[s]\right] + \mathop{\mathrm{E}}_{s\sim\rho_\mathcal{D}(\cdot)}\frac{1}{2}\left[1 - \hat{\pi}_\beta(a|s)\right]\right) \\
\leq& 4\gamma\max_{s,a}\left|A_{\hat{\pi}_\beta}(s,a)\right| \cdot \max_s D_{TV}(\pi\|\hat{\pi}_\beta)[s] \cdot \left(\mathop{\mathrm{E}}_{s\sim\rho_{\hat{\pi}_\beta}(\cdot)}\left[D_{TV}(\pi\|\hat{\pi}_\beta)[s]\right] + \mathop{\mathrm{E}}_{s\sim\rho_\mathcal{D}(\cdot)}\frac{1}{2}\left[1 - \hat{\pi}_\beta(a|s)\right]\right)
\end{aligned} \tag{32}$$

D PROOF OF THEOREM 3

As an extension of theorem 2, the proof process of theorem 3 is similar. Based on the Equation (31), we can directly derive the final bound:

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

E PROOF OF THEOREM 4

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

$$
\begin{aligned}
&\left|\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\left[A_{\pi_k}(s,a)\right]-\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\left[A_{\pi_\beta}(s,a)\right]\right| \\
=&\left|\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\mathbb{E}_{s'\sim p(s'|s,a)}\left[\gamma\left(V_{\pi_k}\left(s'\right)-V_{\pi_\beta}\left(s'\right)\right)-\left(V_{\pi_k}(s)-V_{\pi_\beta}(s)\right)\right]\right| \\
\leq&\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\mathbb{E}_{s'\sim p(s'|s,a)}\left[\gamma\left|V_{\pi_k}\left(s'\right)-V_{\pi_\beta}\left(s'\right)\right|+\left|V_{\pi_k}(s)-V_{\pi_\beta}(s)\right|\right]
\end{aligned}
\tag{34}
$$

Similarly to Equation (21), the value function can be rewritten as $V_\pi(s)=\mathbb{E}_{s\sim\rho_{\pi(\cdot)}}[r(s)]$. Then the difference between two value function can be measured using Hölder's inequality and lemma 2:

$$
\begin{aligned}
&\left|V_{\pi_k}(s)-V_{\pi_\beta}(s)\right| = \left|\mathbb{E}_{s\sim\rho_{\pi_k(\cdot)}}[r(s)]-\mathbb{E}_{s\sim\rho_{\pi_\beta(\cdot)}}[r(s)]\right| \\
\leq&\left\|\rho_{\pi_k}(\cdot)-\rho_{\pi_\beta}(\cdot)\right\|_1\|r(s)\|_\infty\leq 2\gamma\mathop{\mathrm{E}}_{s\sim\rho_{\pi_\beta}(\cdot)}\left[D_{TV}\left(\pi_k\|\pi_\beta\right)[s]\right]\cdot\max_s|r(s)|
\end{aligned}
\tag{35}
$$

Thus, the final bound is

$$
\begin{aligned}
&\left|\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\left[A_{\pi_k}(s,a)\right]-\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\left[A_{\pi_\beta}(s,a)\right]\right| \\
\leq&\mathbb{E}_{s\sim\rho_{\mathcal{D}}(\cdot),a\sim\pi(\cdot|s)}\mathbb{E}_{s'\sim p(s'|s,a)}\Bigg[2\gamma^2\mathop{\mathrm{E}}_{s'\sim\rho_{\pi_\beta}(\cdot)}\left[D_{TV}\left(\pi_k\|\pi_\beta\right)[s']\right]\cdot\max_{s'}|r(s')| \\
&\qquad+2\gamma\mathop{\mathrm{E}}_{s\sim\rho_{\pi_\beta}(\cdot)}\left[D_{TV}\left(\pi_k\|\pi_\beta\right)[s]\right]\cdot\max_s|r(s)|\Bigg] \\
=&2\gamma\left(\gamma+1\right)\max_s|r(s)|\mathop{\mathrm{E}}_{s\sim\rho_{\pi_\beta}(\cdot)}\left[D_{TV}\left(\pi_k\|\pi_\beta\right)[s]\right]
\end{aligned}
\tag{36}
$$

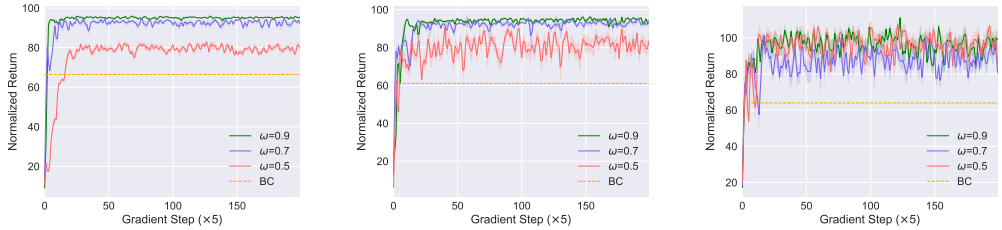
F ABLATION STUDY ON AN ASYMMETRIC COEFFICIENT

In this section, we give the details of all hyperparameter selections in our experiments. In addition to the aforementioned clip ratio ϵ and its clip decay coefficient σ, we introduce the $\omega\in(0,1)$ as an asymmetric coefficient to adjust the advantage $\bar{A}_{\pi_\beta}$ based on the positive or negative of advantage:

$$
\bar{A}_{\pi_\beta}=|\omega-\mathbf{1}(A_{\pi_\beta}<0)|A_{\pi_\beta}. \tag{37}
$$

For $\omega > 0.5$, that downweights the contributions of the state-action value Q_{π_β} smaller than it's expectation, i.e., V_{π_β} while distributing more weights to larger Q_{π_β}. 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. We analyze how the three coefficients affect the performance of BPPO.

We analyze three values of the asymmetric coefficient $\omega = (0.5, 0.7, 0.9)$ in three Gym environments. Figure 5 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. 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, Antmaze, and Kitchen datasets training, respectively.

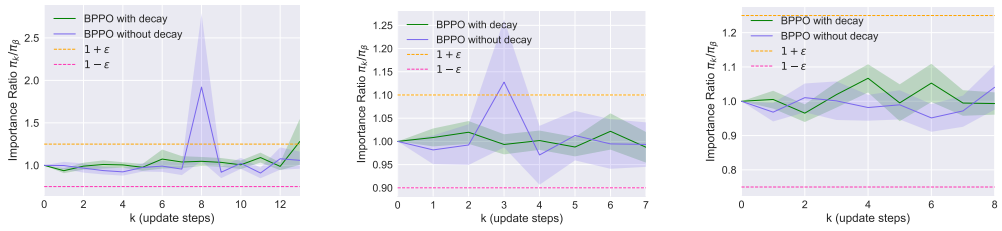


(a) hopper-medium-v2 (b) hopper-medium-replay-v2 (c) hopper-medium-expert-v2

Figure 5: Ablation study on coefficient ω. We optimize the hyperparameters through the grid search, then we fix the value of other coefficients with the best performance and change the value of the asymmetric coefficient to analyze how it affects the BPPO. In particular, $\omega = 0.5$ denotes without the asymmetric coefficient during the training phase (contributing equal value to all Advantages).

G IMPORTANCE RATIO DURING TRAINING

In this section, we consider exploring whether the importance weight between the improved policy π_k and the behavior policy π_β will be arbitrarily large. To this end, we quantify this importance weight in the training phase in Figure 6. In Figure 6, we often observe that the ratio of the BPPO with decay always stays in the clipped region (the region surrounded by the dotted yellow and red line). However, the BPPO without decay is beyond the region in Figure 6(a) and 6(b). That demonstrates the improved policy without decay is farther away from the behavior policy than the case of BPPO with decay. It may cause unstable performance and even crashing, as shown in Figure 4 and 9 when $\sigma = 1.00$ (i.e., without decay).

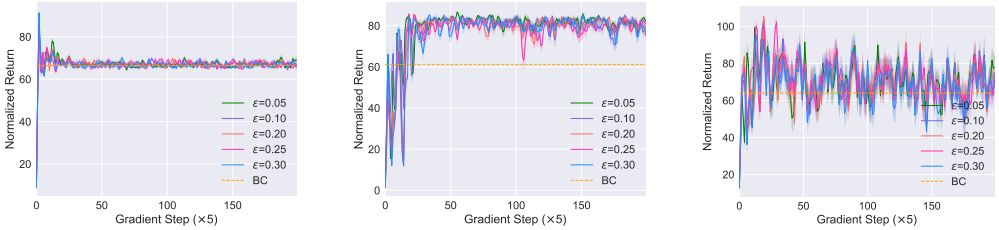


(a) hopper-medium-v2 (b) hopper-medium-replay-v2 (c) hopper-medium-expert-v2

Figure 6: Visualization of the importance weight between the updated policy and the behavior policy trained by BC. When the performance of the policy is improved, we calculate the importance weight (i.e., the probability ratio) between the improved policy and the behavior policy.

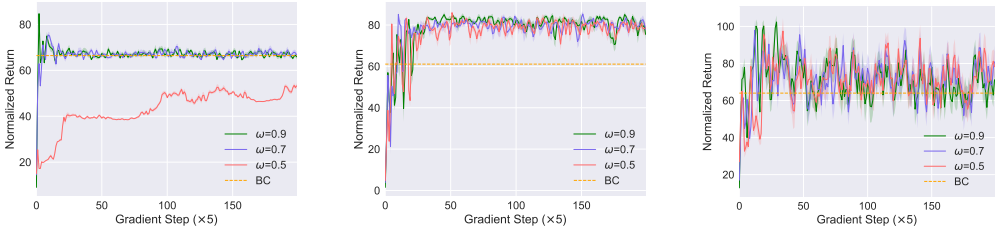
H COEFFICIENT PLOTS OF ONESTEP BPPO

In this section, we exhibit the learning curves and coefficient plots of Onestep BPPO. As shown in Figure 7 and 8, $\epsilon = 0.25$ and $\omega = 0.9$ are best for those tasks. Figure 9 shows how the clip coefficient decay affects the performance of the Onestep BPPO. We can observe that the performance of the curve without decay or with low decay is unstable over three tasks and even crash during training in the *"hopper-medium-replay-v2"* task. Thus, we select $\sigma = 0.96$ to achieve a stable policy improvement for Onestep BPPO. that We use the coefficients with the best performance to compare with the BPPO in Figure 2.



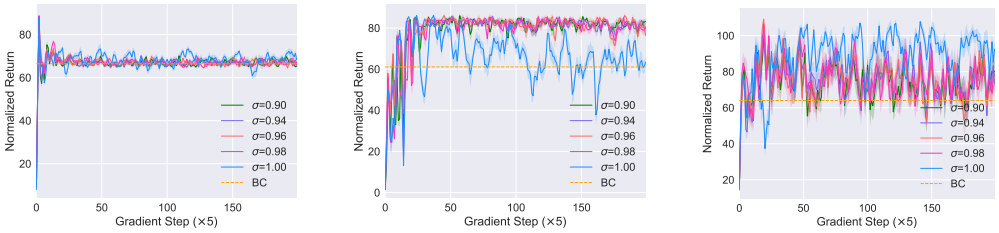
(a) hopper-medium-v2 (b) hopper-medium-replay-v2 (c) hopper-medium-expert-v2

Figure 7: Ablation study of Onestep BPPO on coefficient ϵ. We optimize the hyperparameters through the grid search, then we fix the value of other coefficients with the best performance and change the value of the clip coefficient to analyze how it affects the Onestep BPPO.



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

Figure 8: Ablation study of Onestep BPPO on coefficient ω. We optimize the hyperparameters through the grid search, then we fix the value of other coefficients with the best performance and change the value of the asymmetric coefficient to analyze how it affects the Onestep BPPO.



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

Figure 9: Ablation study of Onestep BPPO on clip coefficient decay and its decay rate. We optimize the hyperparameters through the grid search, then we fix the value of other coefficients with the best performance and change the value of the clip decay coefficient to analyze how it affects the Onestep BPPO. In particular, $\sigma = 1.00$ denotes without the decay coefficient during the training phase.

I EXTRA COMPARISONS

In this section, we have added the EDAC (An et al., 2021), LAPO (Chen et al., 2022), RORL (Yang et al., 2022), and ATAC (Cheng et al., 2022) as the comparison baselines to further evaluate the superiority of the BPPO. Although the performance of the BPPO is slightly worse than the SOTA methods, the BPPO significantly outperforms all methods on the Adroit, Kitchen, and Antmaze datasets and has the best overall performance over all datasets.

Table 3: The normalized results of all algorithms on Gym locomotion and Adroit datasets. The results of the EDAC, RORL, and ATAC are extracted from their original articles.

Environment/method	EDAC	RORL	ATAC	Ours
halfcheetah-medium-v2	**65.9**	**66.8**	54.3	44.0±0.2
hopper-medium-v2	101.6	**104.8**	102.8	93.9±3.9
walker2d-medium-v2	92.5	**102.4**	91.0	83.6±0.9
halfcheetah-medium-replay-v2	61.3	61.9	49.5	41.0±0.6
hopper-medium-replay-v2	101	**102.8**	**102.8**	92.5±3.4
walker2d-medium-replay-v2	87.1	90.4	**94.1**	77.6±7.8
halfcheetah-medium-expert-v2	**106.3**	**107.8**	95.5	92.5±1.9
hopper-medium-expert-v2	110.7	**112.7**	**112.6**	**112.8±1.7**
walker2d-medium-expert-v2	114.7	121.2	**116.3**	113.1±2.4
Gym locomotion-v2 total	*841.1*	***870.8***	*818.9*	*751.0±21.8*
pen-human-v1	52.1	33.7	79.3	**117.8±11.9**
hammer-human-v1	0.8	2.3	6.7	**14.9±3.2**
door-human-v1	10.7	3.8	8.7	**25.9±7.5**
relocate-human-v1	0.1	0	0.3	**4.8±2.2**
pen-cloned-v1	68.2	35.7	73.9	**110.8±6.3**
hammer-cloned-v1	0.3	1.7	2.3	**8.9±5.1**
door-cloned-v1	**9.6**	-0.1	8.2	6.2±1.6
relocate-cloned-v1	0	0	0.8	**1.9±1.0**
adroit-v1 total	*141.8*	*77.1*	*180.2*	***291.4±38.8***
locomotion + adroit total	*982.9*	*947.9*	*999.1*	***1042.4±60.6***

Table 4: The normalized results of all algorithms on Kitchen dataset. The results of the LAPO are extracted from its original article.

Environment/method	LAPO	Ours
kitchen-complete-v0	53.2	**91.5±8.9**
kitchen-partial-v0	53.7	**57.0±2.4**
kitchen-mixed-v0	**62.4**	**62.5±6.7**
kitchen-v0 total	*169.3*	***211.0±18.0***

Table 5: The normalized results of all algorithms on Antmaze dataset. The results of the RORL are extracted from its original article.

Environment/method	RORL	Ours
Umaze-v2	**96.7**	**95.0**±5.5
Umaze-diverse-v2	**90.7**	**91.7±4.1**
Medium-play-v2	**76.3**	51.7±7.5
Medium-diverse-v2	**69.3**	**70.0±6.3**
Large-play-v2	16.3	**86.7±8.2**
Large-diverse-v2	41.0	**88.3±4.1**
Antmaze-v2 total	*390.3*	***483.3±35.7***

J 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). We clip the concatenated gradient of all parameters such that the 'global L2 norm' does not exceed 0.5. 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. Our method is constructed by Pytorch (Paszke et al., 2019). Next, we introduce the training details of the Q, V, behavior policy π_β, and target policy π, respectively.

- Q and V networks training: we run 2×10^6 steps for fitting value Q and V functions using learning rate 10^{-4}, respectively
- Behavior policy π_β training: we run 5×10^5 steps for π_β cloning using learning rate 10^{-4}.
- Target policy π: during policy improvement, we use the learning rate decay, i.e., decaying in each interval step in the first 200 gradient steps and then remaining the learning rate (decay rate $\sigma = 0.96$). We run 1,000 gradient steps for policy improvement for Gym, Adroit, and Kitchen tasks and run 100 gradient steps for Antmaze tasks. The selections of the initial policy learning rate, initial clip ratio, and asymmetric coefficient are listed in Table 6, respectively.

Table 6: The selections of part of hyperparameters during policy improvement phase.

Hyperparameter	Task	Value
Initial policy learning rate	Gym locomotion and cloned tasks of Adroit	10^{-4}
	Kitchen, Antmaze, and human tasks of Adroit	10^{-5}
Initial clip ratio ϵ	Hopper-medium-replay-v2	0.1
	Antmaze	0.5
	Others	0.25
Asymmetric coefficient ω	Gym locomotion	0.9
	Others	0.7