

---

# Iteratively Refined Behavior Regularization for Offline Reinforcement Learning

---

Yi Ma<sup>1,2</sup>, Jianye Hao<sup>3,4\*</sup>, Xiaohan Hu<sup>3</sup>, Yan Zheng<sup>3\*</sup>, Chenjun Xiao<sup>5</sup>

<sup>1</sup>School of Computer and Information Technology, Shanxi University, mayi@sxu.edu.cn

<sup>2</sup>Key Laboratory of Computational Intelligence  
and Chinese Information Processing of Ministry of Education

<sup>3</sup>College of Intelligence and Computing, Tianjin University  
{jianye.hao, huxiaohan, yanzheng}@tju.edu.cn

<sup>4</sup>Noah's Ark Lab, Huawei

<sup>5</sup>The Chinese University of Hongkong, Shenzhen, chenjunx@cuhk.edu.cn

## Abstract

One of the fundamental challenges for offline reinforcement learning (RL) is ensuring robustness to data distribution. Whether the data originates from a near-optimal policy or not, we anticipate that an algorithm should demonstrate its ability to learn an effective control policy that seamlessly aligns with the inherent distribution of offline data. Unfortunately, *behavior regularization*, a simple yet effective offline RL algorithm, tends to struggle in this regard. In this paper, we propose a new algorithm that substantially enhances behavior-regularization based on *conservative policy iteration*. Our key observation is that by iteratively refining the reference policy used for behavior regularization, conservative policy update guarantees gradually improvement, while also implicitly avoiding querying out-of-sample actions to prevent catastrophic learning failures. We prove that in the tabular setting this algorithm is capable of learning the optimal policy covered by the offline dataset, commonly referred to as the *in-sample optimal* policy. We then explore several implementation details of the algorithm when function approximations are applied. The resulting algorithm is easy to implement, requiring only a few lines of code modification to existing methods. Experimental results on the D4RL benchmark indicate that our method outperforms previous state-of-the-art baselines in most tasks, clearly demonstrate its superiority over behavior regularization.

## 1 Introduction

Reinforcement learning (RL) has achieved considerable success in various decision-making problems, including games [1], software automation testing [2], recommendation and advertising [3], logistics optimization [4] and robotics [5]. A critical obstacle that hinders a broader application of RL is its trial-and-error learning paradigm. For applications such as education, autonomous driving and healthcare, active data collection could be either impractical or dangerous [6]. Instead of learning actively, a more favorable approach is to employ scalable data-driven learning methods that can utilize existing data and progressively improve as more training data becomes available. This motivates *offline RL*, of which the primary objective is to learn a control policy solely from previously collected data without online interactions with environment.

Offline datasets frequently offer limited coverage of the state-action space. Directly utilizing standard RL algorithms to such datasets can result in extrapolation errors when bootstrapping from out-of-distribution (OOD) state-actions, consequently causing significant overestimations in value functions.

---

\*Corresponding authors.



To address this, previous works impose various types of constraints to promote pessimism towards accessing OOD state-actions. One simple yet effective approach is *behavior regularization*, which penalizes significant deviations from the behavior policy that collects the offline dataset [7, 8, 9, 10, 11].

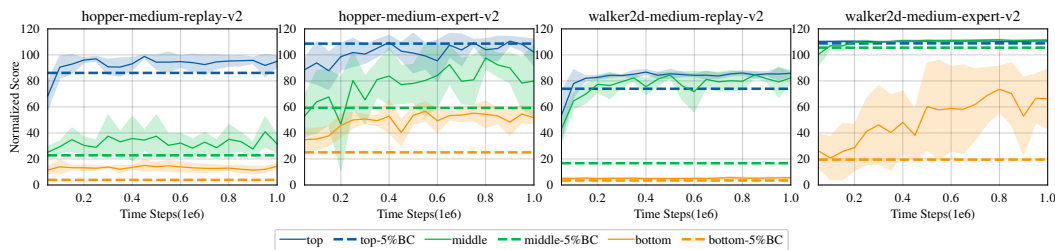


Figure 1: The dashed lines indicate the performance of reference policies trained on filtered sub-datasets consisting of top 5%, median 5%, and bottom 5% trajectories. The solid lines indicate the performance of TD3 with different reference policies. It can be concluded that behavior regularization has advantage over behavior cloning and the efficacy of behavior regularization is strongly contingent on the reference policy.

A fundamental challenge in behavior regularization lies in its performance being closely tied to the underlying data distribution. Previous works suggest that it often yields subpar results when applied to datasets originating from suboptimal policies [12, 13]. To better understand this phenomenon, we investigate how the quality of a behavior policy influences the performance of behavior regularization through utilizing *percentile behavior cloning* [14]. Given an offline dataset, we create three sub-datasets by filtering trajectories representing the top 5%, median 5%, and bottom 5% performance. We then leverage these sub-datasets to train three policies—referred to as *top*, *median*, and *bottom*—using behavior cloning. We then develop *TD3 with percentile behavior cloning (TD3+%BC)*, which optimizes the policy by  $\max_{\pi} \mathbb{E}_{s \sim \mathcal{D}} [v^{\pi}(s) - \lambda(\pi(s) - \pi_{\%}(s))]$  where  $\pi_{\%} \in \{\text{top}, \text{median}, \text{bottom}\}$ . This variant of TD3+BC simply replaces the behavior policy with the percentile cloning policy. We refer to  $\pi_{\%}$  as the *reference policy*. Our main hypothesis is that a better reference policy (e.g. top) can dramatically improve the efficiency of behavior regularization compared to a worse reference policy (e.g. bottom). Figure 1 provides the results on *hopper-medium-replay*, *hopper-medium-expert*, *walker-medium-replay* and *walker-medium-expert* from the D4RL datasets [15]. A few key observations. *First*, the efficacy of behavior regularization is strongly contingent on the reference policy. For example, TD3+top%BC dominates TD3+bottom%BC across all benchmarks. This confirms our hypothesis. *Second*, behavior regularization guarantees improvement. No matter using top, median or bottom, TD3+%BC produces a better or similar policy compared to the reference policy, clearly demonstrating the advantage of behavior regularization over behavior cloning.

Inspired by these findings, we ask: *is it possible to automatically discover good reference policies from the data to increase the efficiency of behavior regularization?* This paper attempts to give an affirmative answer to this question. Our key observation is that Conservative Policy Iteration, an algorithm that has been widely used for online RL [16, 17, 18, 19], can also be extended to the offline setting with minimal changes. The core concept behind this approach hinges on the iterative refinement of the reference policy utilized for behavior regularization. This iterative process enables the algorithm to implicitly avoid resorting to out-of-sample actions, all while ensuring continuous policy enhancement. We provide both theoretical and empirical evidence to establish that, for the tabular setting, our approach is capable of learning the optimal policy covered by the offline dataset, commonly referred to as the in-sample optimal policy [12, 13]. We then discuss several implementation details of the algorithm when function approximations are applied. A previous work STR [20] shares the similar idea with CPI. However, the theoretical foundation and the resulting algorithms are quite distinct, which is discussed in appendix. Our method can be seamlessly implemented with just a few lines of code modifications built upon TD3+BC [11]. We evaluate our method on the D4RL benchmark [15]. Experiment results show that it outperform previous state-of-the-art methods on the majority of tasks, with both rapid training speed and reduced computational overhead.



## 2 Preliminaries

### 2.1 Markov Decision Process

We consider Markov Decision Process (MDP) determined by  $M = \{\mathcal{S}, \mathcal{A}, P, r, \gamma\}$  [21], where  $\mathcal{S}$  and  $\mathcal{A}$  represent the state and action spaces. The discount factor is given by  $\gamma \in [0, 1)$ ,  $r : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$  denotes the reward function,  $P : \mathcal{S} \times \mathcal{A} \rightarrow \Delta(\mathcal{S})$  defines the transition dynamics<sup>2</sup>. Given a policy  $\pi : \mathcal{S} \rightarrow \Delta(\mathcal{A})$ , we use  $\mathbb{E}^\pi$  to denote the expectation under the distribution induced by the interconnection of  $\pi$  and the environment. The *value function* specifies the future discounted total reward obtained by following policy  $\pi$ ,

$$V^\pi(s) = \mathbb{E}^\pi \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \middle| s_0 = s \right], \quad (1)$$

The *state-action value function* is defined as

$$Q^\pi(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} [V^\pi(s')]. \quad (2)$$

There exists an *optimal policy*  $\pi^*$  that maximizes values for all states  $s \in \mathcal{S}$ . The optimal value functions,  $V^*$  and  $Q^*$ , satisfy the *Bellman optimality equation*,

$$\begin{aligned} V^*(s) &= \max_a r(s, a) + \gamma \mathbb{E}_{s'} [V^*(s')], \\ Q^*(s, a) &= r(s, a) + \gamma \mathbb{E}_{s'} \left[ \max_{a'} Q^*(s', a') \right]. \end{aligned} \quad (3)$$

### 2.2 Offline Reinforcement Learning

In this work, we consider learning an optimal decision making policy from previously collected offline dataset, denoted as  $\mathcal{D} = \{s_i, a_i, r_i, s'_i\}_{i=0}^{n-1}$ . The dataset is generated following this procedure:  $s_i \sim \rho, a_i \sim \pi_{\mathcal{D}}, s'_i \sim P(\cdot|s_i, a_i), r_i = r(s_i, a_i)$ , where  $\rho$  represents an unknown probability distribution over states, and  $\pi_{\mathcal{D}}$  is an *unknown behavior policy*. In offline RL, the learning algorithm can only take samples from  $\mathcal{D}$  without collecting new data through interactions with the environment.

Behavior regularization is a simple yet efficient technique for offline RL [7, 8]. It imposes a constraint on the learned policy to emulate  $\pi_{\mathcal{D}}$  according to some distance measure. A popular choice is to use the KL-divergence [11, 10]<sup>3</sup>,

$$\max_{\pi} \mathbb{E}_{a \sim \pi} [Q(s, a)] - \tau D_{\text{KL}}(\pi(s) || \pi_{\mathcal{D}}(s)) , \quad (4)$$

where  $\tau > 0$  is a hyper-parameter. Here,  $Q$  is some value function. Typical choices include the value of  $\pi_{\mathcal{D}}$  [10], or the value of the learned policy  $\pi$  [8, 11]. As shown by Figure 1, the main limitation of behavior regularization is that it relies on the dataset being generated by an expert or near-optimal  $\pi_{\mathcal{D}}$ . When used on datasets derived from more suboptimal policies—typical of those prevalent in real-world applications—these methods do not yield satisfactory results.

## 3 Iteratively Refined Behavior Regularization

In this paper, we introduce a new offline RL algorithm by exploring idea of iteratively improving the reference policy used for behavior regularization. Our primary goal is to improve the robustness of existing behavior regularization methods while making minimal changes to existing implementation. We first introduce *conservative policy optimization (CPO)*, a commonly used technique in online RL, then describe how it can also shed some light on developing offline RL algorithms.

Let  $\tau > 0$ , the CPO update the policy for any  $s \in \mathcal{S}$  with the following policy optimization problem

$$\max_{\pi} \mathbb{E}_{a \sim \pi} [Q^{\bar{\pi}}(s, a)] - \tau D_{\text{KL}}(\pi(s) || \bar{\pi}(s)) , \quad (5)$$

<sup>2</sup>We use  $\Delta(\mathcal{X})$  to denote the set of probability distributions over  $\mathcal{X}$  for a finite set  $\mathcal{X}$ .

<sup>3</sup>We note that although [11] applies a behavior cloning term as regularization, this is indeed a KL regularization under Gaussian parameterization with standard deviation.



which generalizes behavior regularization (4) using an arbitrary *reference policy*  $\bar{\pi}$ . The idea is to optimize a policy without moving too far away from the reference policy to increase learning stability. As shown in the following proposition, this conservative policy update rule enjoys two intriguing properties: *first*, it guarantees policy improvement over the reference policy  $\bar{\pi}$ ; and *second*, the updated policy still stays on the support of the reference policy. These properties also echo our empirical findings presented in Figure 1: as a special case of CPO with  $\bar{\pi} = \pi_{\mathcal{D}}$ , behavior regularization is stable in offline learning and always produces a better policy than  $\pi_{\mathcal{D}}$ .

**Proposition 1.** *let  $\bar{\pi}^*$  be the optimal policy of (5). For any  $s \in \mathcal{S}$ , we have that  $V^{\bar{\pi}^*}(s) \geq V^{\bar{\pi}}(s)$ ; and  $\bar{\pi}^*(a|s) = 0$  given  $\bar{\pi}(a|s) = 0$ .*

*Proof.* The proof of this result, as well as other results, are provided in the Appendix.  $\square$

In summary, the conservative policy update (5) *implicitly guarantees policy improvement constrained on the support of the reference policy*  $\bar{\pi}$ . By extending this key observation in an iteratively manner, we obtain the following *Conservative Policy Iteration (CPI)* algorithm for offline RL. It starts with the behavior policy  $\pi_0 = \pi_{\mathcal{D}}$ . Then in each iteration  $t = 0, 1, 2, \dots$ , the following computations are done:

- *Policy evaluation:* compute  $Q^{\pi_t}$  and;
- *Policy improvement:*  $\forall s \in \mathcal{S}, \pi_{t+1} = \arg \max_{\pi} \mathbb{E}_{a \sim \pi} [Q^{\pi_t}(s, a)] - \tau D_{\text{KL}}(\pi || \pi_t)$ .

In this approach, the algorithm commences with the behavior policy and proceeds to iteratively refine the reference policy used for behavior regularization. Thanks to the conservative policy update, CPI ensures policy improvement while mitigating the risk of querying any OOD actions that could potentially introduce instability to the learning process.

We note that CPO is a special case of *mirror descent* in the online learning literature [22]. Extending this technique to sequential decision making has been previously investigated in online RL [16, 17, 18, 19]. In particular, the *Polite* algorithm considers exactly the same update as (5) for online RL [17]. This algorithm can be viewed as a softened or averaged version of policy iteration. Such averaging reduces noise of the value estimator and increases the robustness of policy update. Perhaps surprisingly, our key contribution is to show this simple yet powerful policy update rule also facilitates offline learning, as it guarantees policy improvement while implicitly avoid querying OOD actions.

### 3.1 Theoretical Analysis

We now analyze the convergence properties of CPI. In particular, we consider the tabular setting with finite state and action space. Our analysis reveals that in this setting, CPI converges to the optimal policy that are well-covered by the dataset, commonly referred to as the in-sample optimal policy. Consider the *in-sample Bellman optimality equation* [23]

$$V_{\pi_{\mathcal{D}}}^*(s) = \max_{a: \pi_{\mathcal{D}}(a|s) > 0} \left\{ r(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s, a)} [V_{\pi_{\mathcal{D}}}^*(s')] \right\}. \quad (6)$$

This equation explicitly avoids bootstrapping from OOD actions while still guaranteeing optimality for transitions that are well-supported by the data. A fundamental challenge lies in the development of scalable algorithms for its resolution [23, 12, 13]. Our next result shows that CPI provides a simple yet effective solution.

**Theorem 1.** *We consider tabular MDPs with finite  $\mathcal{S}$  and  $\mathcal{A}$ . Let  $\pi_t$  be the produced policy of CPI at iteration  $t$ . There exists a parameter  $\tau > 0$  such that for any  $s \in \mathcal{S}$*

$$V_{\pi_{\mathcal{D}}}^*(s) - V^{\pi_t}(s) \leq \frac{1}{(1 - \gamma)^2} \sqrt{\frac{2 \log |\mathcal{A}|}{t}}. \quad (7)$$

To ensure robustness to data distribution, an offline RL algorithm must possess the *stitching* capability, which involves seamlessly integrating suboptimal trajectories from the dataset. In-sample optimality provides a formal definition to characterize this ability. As discussed in previous works and confirmed by our experiments, existing behavior regularization approaches, such as TD3+BC [11], fall short in this regard. In contrast, Theorem 1 suggests that iteratively refining the reference policy for behavior regularization can enable the algorithm to acquire the stitching ability, marking a significant improvement over existing approaches. Our empirical studies in the experiments section further support this claim.



## 4 Practical Implementations

In this section we discuss how to implement CPI properly when function approximation is applied. Throughout this section we develop algorithms for continuous actions. Extension to discrete action setting is straightforward. We build CPI as an actor-critic algorithm. We learn an actor  $\pi_\omega$  with parameters  $\omega$ , and critic  $Q_\theta$  with parameters  $\theta$ . The policy  $\pi_\omega$  is parameterized using a Gaussian policy with learnable mean [24]. We also normalize features of every states in the offline dataset as discussed in [11].

CPI consists of two steps at each iteration. The first step involves policy evaluation, which is carried out using standard TD learning: we learn the critic by

$$\min_{\theta} \mathbb{E}_{s,a,r,s' \sim \mathcal{D}, a' \sim \pi_\omega(s')} \left[ \frac{1}{2} (r + \gamma Q_{\bar{\theta}}(s', a') - Q_\theta(s, a))^2 \right], \quad (8)$$

where  $Q_{\bar{\theta}}$  is a target network. We also apply the double-Q trick to stabilize training [24]. The policy improvement step requires more careful algorithmic design. The straightforward implementation is to use gradient descent on the following

$$\max_{\omega'} \mathbb{E}_{s \sim \mathcal{D}} \left[ \mathbb{E}_{a \sim \pi_{\omega'}} [Q_\theta(s, a)] - \tau D_{\text{KL}}(\pi_\omega(s) \| \bar{\pi}(s)) \right]. \quad (9)$$

Here, the reference policy  $\bar{\pi}$  is a copy of the current parameters  $\omega$  and kept frozen during optimization. This leads to the so-called *iterative actor-critic* or *multi-step* algorithm [10]. In their study, [10] observe that this iterative algorithm frequently encounters practical challenges, mainly attributed to the substantial variance associated with off-policy evaluation. Similar observations have also been made in our experiments (See Figure 4). We conjecture that while Proposition 1 establishes that the exact solution of (9) remains within the data support when the actor is initialized as  $\pi_\omega = \pi_{\mathcal{D}}$ , practical implementations often rely on a limited number of gradient descent steps for optimizing (9), thus could suffer from out-of-support samples. This leads to policy optimization errors which are further exacerbated iteratively. An option to approximate the in-support learning is to utilize importance sampling on the dataset to mimic sampling from the current policy. However, the high variance of importance sampling could cause the training unstable [20]. We instead find it is useful to add the original behavior regularization,

$$\max_{\omega'} \mathbb{E}_{s \sim \mathcal{D}} \left[ \mathbb{E}_{a \sim \pi_{\omega'}} [Q_\theta(s, a)] - \tau \lambda D_{\text{KL}}(\pi_\omega(s) \| \bar{\pi}(s)) - \tau(1 - \lambda) D_{\text{KL}}(\pi_\omega(s) \| \pi_{\mathcal{D}}(s)) \right], \quad (10)$$

which further constrains the policy on the support of data to enhance learning stability. The parameter  $\lambda$  balances between one-step policy improvement and behavior regularization. Note that the term  $D_{\text{KL}}(\pi_\omega(s) \| \bar{\pi}(s))$  in CPI is the only difference compared to TD3+BC [11]. In practice this can be done with one line code modification based on TD3+BC implementations.

---

### Algorithm 1 CPI & CPI-RE

---

- 1: Initialize actors  $\pi_1, \pi_2$  and critic networks  $q_1, q_2$  with random parameters  $\omega^1, \omega^2, \theta^1, \theta^2$ ; target networks  $\bar{\theta}^1 \leftarrow \theta^1, \bar{\theta}^2 \leftarrow \theta^2, \bar{\omega}^1 \leftarrow \omega^1, \bar{\omega}^2 \leftarrow \omega^2$ .
  - 2: **for**  $t = 0, 1, 2, \dots, T$  **do**
  - 3:   Sample a mini-batch of transitions from dataset
  - 4:   Update the parameters of critic  $\theta^i$  using equation 8
  - 5:   **if**  $t \bmod 2$  **then**
  - 6:     // For CPI
  - 7:     Copy the historical snapshot of  $\omega^1$  to  $\omega^2$  and update  $\omega^1$  using equation 10
  - 8:     // For CPI-RE
  - 9:     Choose the current best policy between  $\omega^1$  and  $\omega^2$  as the reference policy according to Q-value
  - 10:    Update  $\omega^1$  and  $\omega^2$  using equation 10 respectively
  - 11:    Update target networks
  - 12:   **end if**
  - 13: **end for**
-



**Ensembles of Reference Policy** One limitation of (10) lies in the potential for a negligible difference between the learning policy and the reference policy, due to the limited gradient steps when optimizing. This minor discrepancy may restrict the policy improvement over the reference policy. To improve the efficiency of policy improvement, we explore the idea of using an ensembles of reference policies. In particular, we apply two policies with independently initialized parameters  $\omega^1$  and  $\omega^2$ . Let  $Q_{\theta^1}$  and  $Q_{\theta^2}$  be the value functions of these two policies respectively. When updating the parameters  $\omega^i$  for  $i \in \{1, 2\}$ , we choose the current best policy as the reference policy, where the superiority is decided according to the current value estimate. In other words, we only enable a superior reference policy to elevate the performance of the learning policy, preventing the learning policy from being dragged down by an inferior reference policy. We call this algorithm *Conservative Policy iteration with Reference Ensembles (CPI-RE)*. We give the pseudocode of both CPI and CPI-RE in Algorithm 1. For CPI, the training process is exactly the same with that of TD3+BC except for the policy updating (marked in pink). The difference between CPI-RE and CPI is also reflected in the policy updating (marked in teal).

## 5 Experiment

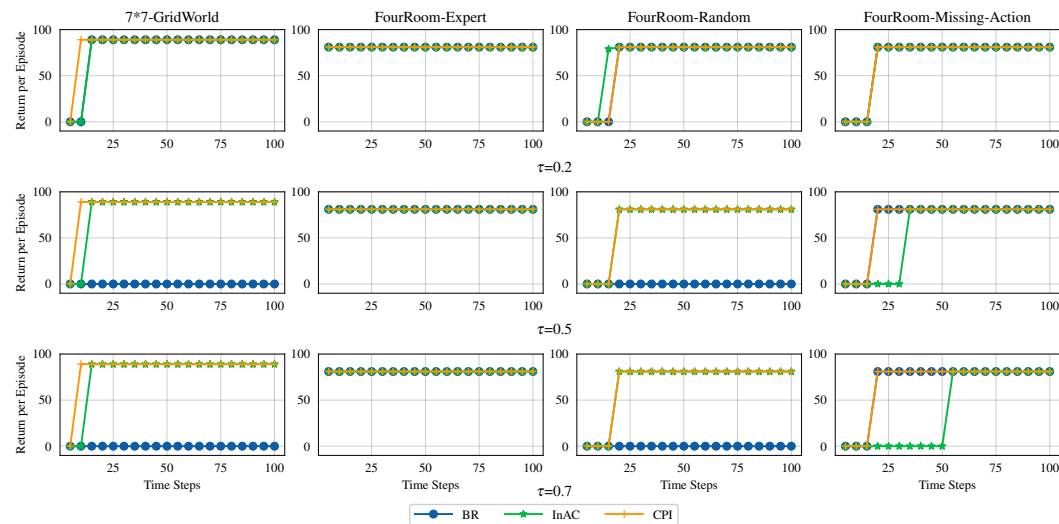


Figure 2: Training curves of BR, InAC and CPI on 7\*7-GridWorld and FourRoom. CPI converges to the oracle across various  $\tau$  and environments settings, similar to the in-sample optimal policy InAC.

We subject our algorithm to a series of rigorous experimental evaluations. We first present an empirical study to exemplify CPI’s optimality in the tabular setting. Then, we compare the practical implementations of CPI, utilizing function approximation, against prior state-of-the-art algorithms in the D4RL benchmark tasks [15], to highlight its superior performance. In addition, we also present the resource consumption associated with different algorithms. Finally, comprehensive analysis of various designs to scrutinize their impact on the algorithm’s performance are provided.

### 5.1 Optimality in the tabular setting

We first conduct an evaluation of CPI within two GridWorld environments. The first environment’s map comprises a  $7 \times 7$  grid layout and the second environment’s map is the FourRoom [13]. In both environments, the agent is tasked with navigating from the bottom-left to the goal positioned in the upper-right in as few steps as possible. The agent has access to four actions:  $\{up, down, right, left\}$ . The reward is set to -1 for each movement, with a substantial reward of 100 upon reaching the goal; this incentivizes the agent to minimize the number of steps taken. Each episode is terminated after 30 steps, and  $\gamma$  is set to 0.9. We use an *inferior behavior policy* to collect 10k transitions, of which the action probability is  $\{up:0.1, down:0.4, right:0.1, left:0.4\}$  at every state in the  $7 \times 7$  grid environment. For the FourRoom environment, we use three types of behavior policy to collect data: (1) Expert dataset: collect 10k transitions with the optimal policy; (2) Random dataset: collect 10k transitions



with a random restart and equal probability of taking each action; (3) Missing-Action dataset: remove all *down* actions in transitions of the upper-left room from the Mixed dataset. Although some behavior policies are suboptimal, the optimal path is ensured to exist in the offline data, in which case a clear algorithm should still be able to identify the optimal path.

We consider two baseline algorithms: InAC [13], a method that guarantees to find the in-sample softmax, and a method that employs policy iteration with behavior regularization (BR), which could be viewed as an extension of TD3+BC [11] for the discrete action setting. The policies derived from each method are evaluated using greedy strategy that selects actions with highest probability.

As illustrated in Figure 2, both CPI and InAC converge to the oracle across various  $\tau$  and environments settings. In contrast, BR underperforms when a larger  $\tau$  is applied on 7\*7-GridWorld and FourRoom-random, as larger  $\tau$  means more powerful constraint on the learning policy, BR thus become more similar to the behavioral policy.

## 5.2 Results on Continuous Control Problems

Table 1: Average normalized scores of CPI with the mean and standard deviation and previous methods on the D4RL benchmark. D-QL is short for Diffusion-QL. CPI achieves best overall performance among all the methods and consumes quite few computing resources. The top-3 results on each dataset is marked as bold.

Dataset	DT	TD3+BC	CQL	IQL	POR	EDAC	InAC	STR	D-QL	CPI	CPI-RE
halfcheetah-random	2.2	11.0	<b>31.3</b>	13.7	29.0	28.4	19.6	20.6	22.0	<b>29.7±1.1</b>	<b>30.7±0.4</b>
hopper-random	5.4	8.5	5.3	8.4	12.0	25.3	<b>32.4</b>	<b>31.3</b>	18.3	29.5±3.7	<b>30.4±2.9</b>
waker2d-random	2.2	1.6	5.4	5.9	<b>6.3</b>	<b>16.6</b>	6.3	4.7	5.5	<b>5.9±1.7</b>	5.5±0.9
halfcheetah-medium	42.6	48.3	46.9	47.4	48.8	<b>65.9</b>	48.3	51.8	51.5	<b>64.4±1.3</b>	<b>65.9±1.6</b>
hopper-medium	67.6	59.3	61.9	66.3	98.2	<b>101.6</b>	60.3	<b>101.3</b>	96.6	<b>98.5±3.0</b>	97.9±4.4
waker2d-medium	74.0	83.7	79.5	78.3	81.1	<b>92.5</b>	82.7	85.9	<b>87.3</b>	85.8±0.8	<b>86.3±1.0</b>
halfcheetah-medium-replay	36.6	44.6	45.3	44.2	43.5	<b>61.3</b>	44.3	47.5	48.3	<b>54.6±1.3</b>	<b>55.9±1.5</b>
hopper-medium-replay	82.7	60.9	86.3	94.7	98.9	101.0	92.1	100.0	<b>102.0</b>	<b>101.7±1.6</b>	<b>103.2±1.4</b>
waker2d-medium-replay	66.6	81.8	76.8	73.9	76.6	87.1	69.8	85.7	<b>98.0</b>	<b>91.8±2.9</b>	<b>93.8±2.2</b>
halfcheetah-medium-expert	86.8	90.7	95.0	86.7	94.7	<b>106.3</b>	83.5	94.9	<b>97.2</b>	94.7±1.1	<b>95.6±0.9</b>
hopper-medium-expert	107.6	98.0	96.9	91.5	90.0	<b>110.7</b>	93.8	<b>111.9</b>	<b>112.3</b>	106.4±4.3	110.1±4.1
waker2d-medium-expert	108.1	110.1	109.1	109.6	109.1	<b>114.7</b>	109.0	110.2	<b>111.2</b>	110.9±0.4	<b>111.2±0.5</b>
halfcheetah-expert	87.7	96.7	<b>97.3</b>	94.9	93.2	<b>106.8</b>	93.6	95.2	96.3	96.5±0.2	<b>97.4±0.4</b>
hopper-expert	94.2	107.8	106.5	108.8	110.4	110.1	103.4	<b>111.2</b>	102.6	<b>112.2±0.5</b>	<b>112.3±0.5</b>
waker2d-expert	108.3	110.2	109.3	109.7	102.9	<b>115.1</b>	110.6	110.1	109.5	<b>110.6±0.1</b>	<b>111.2±0.2</b>
Gym-MuJoCo Total	972.6	1013.2	1052.8	1034.0	1094.7	<b>1243.4</b>	1049.7	1162.2	1158.6	<b>1193.2</b>	<b>1207.4</b>
antmaze-umaze	59.2	78.6	74.0	87.5	76.8	16.7	84.8	93.6	<b>96.0</b>	<b>98.8±1.1</b>	<b>99.2±0.5</b>
antmaze-umaze-diverse	53.0	71.4	84.0	62.2	64.8	0.0	82.4	77.4	<b>84.0</b>	<b>88.6±5.7</b>	<b>92.6±10.0</b>
antmaze-medium-play	0.0	3.0	61.2	71.2	<b>87.2</b>	0.0	-	<b>82.6</b>	79.8	82.4±5.8	<b>84.8±5.0</b>
antmaze-medium-diverse	0.0	10.6	53.7	70.0	75.2	0.0	-	<b>87.0</b>	<b>82.0</b>	80.4±8.9	<b>80.6±11.3</b>
antmaze-large-play	0.0	0.0	15.8	<b>39.6</b>	24.4	0.0	-	<b>42.8</b>	<b>49.0</b>	20.6±16.3	33.6±8.1
antmaze-large-diverse	0.0	0.2	14.9	47.5	<b>59.2</b>	0.0	-	46.8	<b>61.7</b>	45.2±6.9	<b>48.0±6.2</b>
Antmaze Total	112.2	163.8	303.6	378.0	387.6	16.7	-	<b>430.2</b>	<b>452.5</b>	416.0	<b>438.8</b>
pen-human	73.9	-1.9	35.2	71.5	<b>76.9</b>	52.1	52.3	-	75.7	<b>80.1±16.9</b>	<b>87.0±25.3</b>
pen-cloned	67.3	9.6	27.2	37.3	67.6	<b>68.2</b>	-8.0	-	60.8	<b>71.8±35.2</b>	<b>70.7±15.8</b>
Adroit Total	141.2	7.7	62.4	108.8	<b>144.5</b>	120.3	44.3	-	136.5	<b>151.9</b>	<b>157.7</b>
Total	1226.0	1184.7	1418.8	1520.8	<b>1626.8</b>	1380.4	-	-	<b>1747.6</b>	<b>1761.2</b>	<b>1803.9</b>
Runtime (s/epoch)	-	<b>7.4</b>	-	-	-	19.6	-	-	39.8	<b>8.5</b>	19.1
GPU Memory (GB)	-	<b>1.4</b>	-	-	-	1.9	-	-	1.5	<b>1.4</b>	<b>1.4</b>

In this section we provide a suite of results using three continuous control tasks from D4RL [15]: Mujoco, Antmaze, and Adroit. Mujoco, a benchmark often used in previous studies forms the basis of our experimental framework. Adroit is a high-dimensional robotic manipulation task with sparse rewards. Antmaze, with its sparse reward property, necessitates that the agent learns to discern segments within sub-optimal trajectories and to assemble them, thereby discovering the complete trajectory leading to a rewardable position. Given that the majority of datasets for these tasks contain a substantial volume of sub-optimal or low-quality data, relying solely on behavior-regularization may be detrimental to performance.

We compare CPI with several baselines, including DT [14], TD3+BC [11], CQL [25], IQL [12], POR [26], EDAC [27], Diffusion-QL [28], InAC [13] and STR [20]. The results of baseline methods are either reproduced by executing the official code or sourced directly from the original papers. Unless otherwise specified, results are depicted with 95% confidence intervals, represented by shaded areas in figures and expressed as standard deviations in tables. The average normalized results of the



final 10 evaluations for Mujoco and Adroit, and the final 100 evaluations for Antmaze, are reported. More details of the experiments are provided in the Appendix.

Our experiment results, summarized in Table 1, clearly show CPI outperforms baselines in overall. In most Mujoco tasks, CPI surpasses the extant, widely-utilized algorithms, and it only slightly trails behind the state-of-the-art EDAC method. For Antmaze and Adroit, CPI's performance is on par with the top-performing methods such as POR and Diffusion-QL. We also evaluate the resource consumption of different algorithms from two aspects: (1) runtime per training epoch (1000 gradient steps); (2) GPU memory consumption. The results in Table 1 show that CPI requires fewer resources which could be beneficial for practitioners.

### 5.3 Ablation Studies

#### 5.3.1 Effect of Reference Ensemble

We provide learning curves of CPI and CPI-RE on Antmaze in Figure 3 to further show the efficacy of using ensemble of reference policies. CPI-RE exhibits a more stable performance compared to vanilla CPI, also outperforming IQL. Learning curves on other domains are provided in the Appendix.

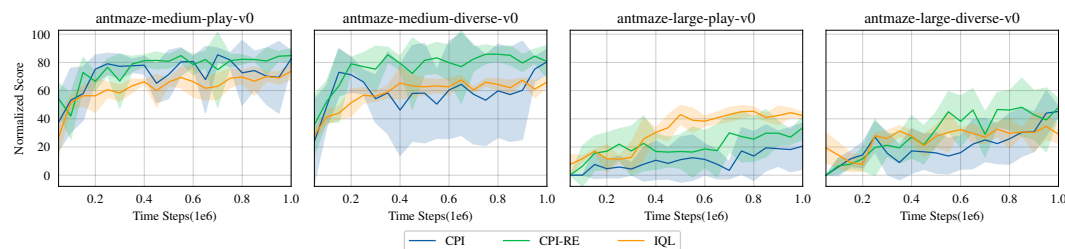


Figure 3: The learning curves of CPI, CPI-RE, and IQL on Antmaze. With reference ensemble, CPI-RE exhibits a more stable performance compared to vanilla CPI, also outperforming IQL.

#### 5.3.2 Effect of using Behavior Regularization

A direct implementation based on the theoretical results observed in the tabular setting could be derived. Specifically, we initialize  $\pi_\omega = \pi_D$  and execute the CPI without incorporating behavior regularization. For each gradient step, we perform *one-step* or *multi-step* updates for both policy evaluation and policy improvement. The empirical outcomes presented in Fig.4 underscore the poor performance of this straightforward approach. Similar trends are also observed in [10]. Such phenomenon arises primarily because, in the presence of function approximation, the deviation emerges when policy optimization does not necessarily remain within the defined support without the behavior regularization.

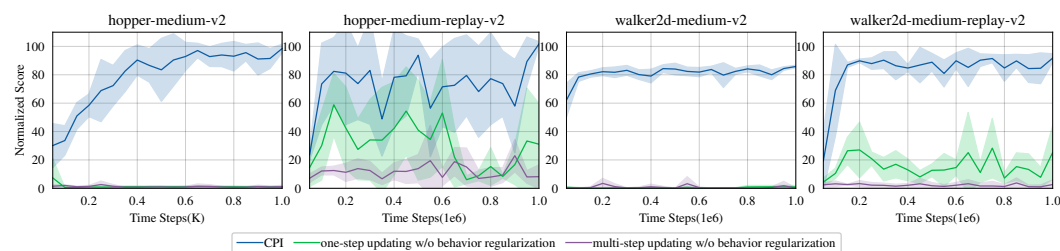


Figure 4: Effect of using Behavior Regularization. Without it, both one-step and multi-step updating methods at each gradient step suffer from deviation derived from out-of-support optimization.

#### 5.3.3 Effect of using different KL strategies

In our implementation of the CPI method, we employ the reverse KL divergence. This is distinct from the forward KL divergence approach adopted by [29]. A comprehensive ablation of incorporating different KL divergence strategies in CPI is presented in Fig.5. As evidenced from the



results, our reverse KL-based CPI exhibits superior performance compared to the forward KL-based implementations. Implementation details of CPI with forward KL are provided in Appendix.

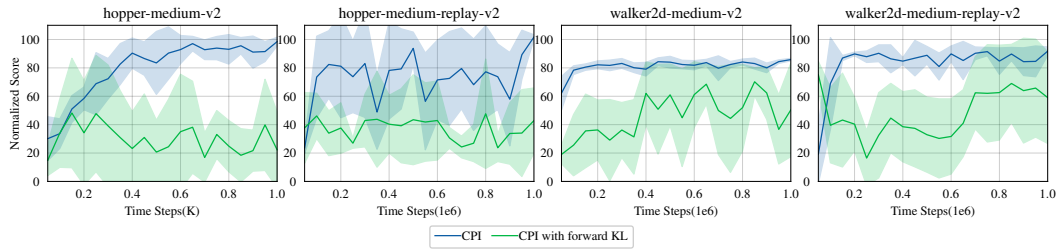


Figure 5: Effect of using different KL strategies. Reverse KL-based CPI exhibits superior performance compared to the forward KL-based CPI. See Appendix for details of forward KL.

### 5.3.4 Effect of Hyper Parameters

Figure 6 illustrates the effects of using different hyperparameters  $\tau$  and  $\lambda$ , offering valuable insights for algorithm tuning. The weighting coefficient  $\lambda$  regulates the extent of behavior policy integration into the training and affects the training process. As shown in Figure 6a, when  $\lambda = 0.1$ , the early-stage performance excels, as the behavior policy assists in locating appropriate actions in the dataset. However, this results in suboptimal final convergence performance, attributable to the excessive behavior policy constraint on performance improvement. For larger values, such as 0.9, the marginal weight of the behavior policy leads to performance increase during training. Unfortunately, the final performance might be poor. This is due to that the policy does not have sufficient behavior cloning guidance, leading to a potential distribution shift during the training process. Consequently, we predominantly select a  $\lambda$  value of 0.5 or 0.7 to strike a balance between the reference policy regularization and behavior regularization. The regularization parameter  $\tau$  plays a crucial role in determining the weightage of the joint regularization relative to the Q-value component. We find that (Figure 6b)  $\tau$  assigned to dataset of higher quality and lower diversity (e.g., expert dataset) ought to be larger than those associated with datasets of lower quality and higher diversity (e.g., medium dataset).

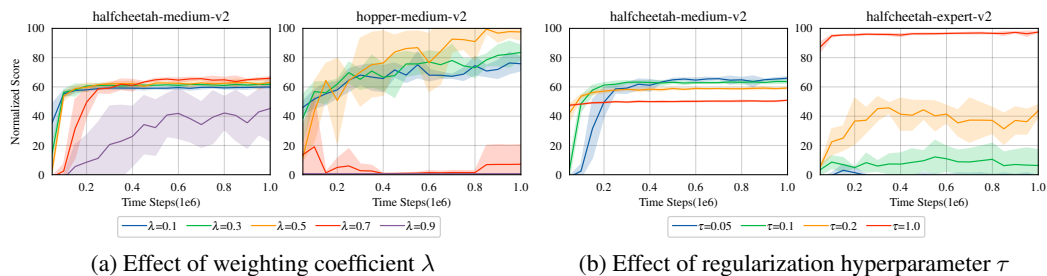


Figure 6: Hyperparameters ablation studies. The tuning experience drawn from this include: (1)  $\lambda$  could mostly be set to 0.5 or 0.7; (2) The higher the dataset quality is, the higher  $\tau$  should be set.

### 5.4 Online Fine-tuning

The distinctive feature of CPI lies in its policy iteration training, which makes it particularly well-suited for the online fine-tuning process. Fine adjustments to CPI enable its seamless application in online fine-tuning. In an online setting, as the agent can interact with the environment, there is a need to progressively enhance the level of exploration during the training process. To achieve this, the  $\lambda$  parameter in formula 10 is modified. Specifically, in the online training process, the weight of  $D_{KL}(\pi_\omega(s)||\pi(s))$  remains constant, while the weight of  $D_{KL}(\pi_\omega(s)||\pi_D(s))$  decreases exponentially, eventually reducing to 0.1. This progressive adaptive exploration mechanism allows the algorithm to adopt a conservative exploration strategy in its initial stages and then gradually expand its exploration scope, thereby enhancing the model's adaptability in the online learning phase.



Empirical evaluations were conducted on nine Mujoco datasets, including halfcheetah-medium-v2, halfcheetah-medium-replay-v2, halfcheetah-medium-expert-v2, hopper-medium-v2, hopper-medium-replay-v2, hopper-medium-expert-v2, walker2d-medium-v2, walker2d-medium-replay-v2, and walker2d-medium-expert-v2. We compare our algorithms with several baselines including TD3+BC, IQL, Cal-QL [30] and PEX [31]. As shown in Figure 7, the CPI algorithm achieves exceptional overall performance compared to the state-of-the-art offline-to-online algorithms Cal-QL and PEX. Detailed results for each Mujoco dataset are provided in the Appendix.

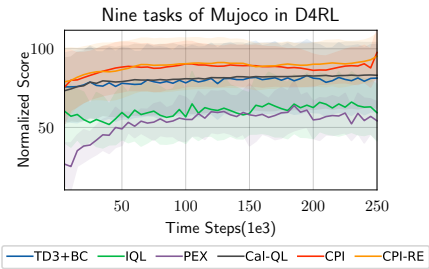


Figure 7: Overall finetuning performance on Mujoco datasets.

## 6 Conclusion

In this paper, we propose an innovative offline RL algorithm termed *Conservative Policy iteration (CPI)*. By iteratively refining the policy used for regularization, CPI progressively improves itself within the behavior policy’s support and provably converges to the in-sample optimal policy in the tabular setting. We then propose practical implementations of CPI for solving continuous control tasks. Experimental results on the D4RL benchmark show that CPI surpass previous cutting-edge methods in a majority of tasks of various domains, offering both expedited training speed and diminished computational overhead.

Nonetheless, our study is not devoid of limitations. For instance, our method’s performance with function approximation is contingent upon the selection of two hyperparameters, which may necessitate tuning for optimal results.

## Acknowledgments and Disclosure of Funding

The work is supported by the National Natural Science Foundation of China (Grant Nos: 62422605, 92370132, 62406271, 62106172), the National Key R&D Program of China (Grant No. 2022ZD0116402) and the Xiaomi Young Talents Program of Xiaomi Foundation.

## References

- [1] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.
- [2] Yan Zheng, Xiaofei Xie, Ting Su, Lei Ma, Jianye Hao, Zhaopeng Meng, Yang Liu, Ruimin Shen, Yingfeng Chen, and Changjie Fan. Wuji: Automatic online combat game testing using evolutionary deep reinforcement learning. In *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 772–784. IEEE, 2019.
- [3] Xiaotian Hao, Zhaoqing Peng, Yi Ma, Guan Wang, Junqi Jin, Jianye Hao, Shan Chen, Rongquan Bai, Mingzhou Xie, Miao Xu, et al. Dynamic knapsack optimization towards efficient multi-channel sequential advertising. In *International Conference on Machine Learning*, pages 4060–4070. PMLR, 2020.
- [4] Yi Ma, Xiaotian Hao, Jianye Hao, Jiawen Lu, Xing Liu, Tong Xialiang, Mingxuan Yuan, Zhigang Li, Jie Tang, and Zhaopeng Meng. A hierarchical reinforcement learning based optimization framework for large-scale dynamic pickup and delivery problems. *Advances in Neural Information Processing Systems*, 34:23609–23620, 2021.
- [5] Ajay Mandekar, Fabio Ramos, Byron Boots, Silvio Savarese, Li Fei-Fei, Animesh Garg, and Dieter Fox. Iris: Implicit reinforcement without interaction at scale for learning control from offline robot manipulation data. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4414–4420. IEEE, 2020.
- [6] 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.



- [7] 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.
- [8] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.
- [9] 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.
- [10] 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.
- [11] Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.
- [12] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022.
- [13] Chenjun Xiao, Han Wang, Yangchen Pan, Adam White, and Martha White. The in-sample softmax for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023.
- [14] 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.
- [15] 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.
- [16] Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation. In *International Conference on Learning Representations*, 2018.
- [17] Yasin Abbasi-Yadkori, Peter Bartlett, Kush Bhatia, Nevena Lazic, Csaba Szepesvari, and Gellért Weisz. Politex: Regret bounds for policy iteration using expert prediction. In *International Conference on Machine Learning*, pages 3692–3702. PMLR, 2019.
- [18] Jincheng Mei, Chenjun Xiao, Ruitong Huang, Dale Schuurmans, and Martin Müller. On principled entropy exploration in policy optimization. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 3130–3136, 2019.
- [19] Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. A theory of regularized markov decision processes. In *International Conference on Machine Learning*, pages 2160–2169. PMLR, 2019.
- [20] Yixiu Mao, Hongchang Zhang, Chen Chen, Yi Xu, and Xiangyang Ji. Supported trust region optimization for offline reinforcement learning. In *International Conference on Machine Learning*, pages 23829–23851. PMLR, 2023.
- [21] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.
- [22] Elad Hazan et al. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.
- [23] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pages 2052–2062. PMLR, 2019.
- [24] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.
- [25] 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.
- [26] Haoran Xu, Li Jiang, Jianxiong Li, and Xianyu Zhan. A policy-guided imitation approach for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2022.
- [27] 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.



- [28] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.
- [29] Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.
- [30] Mitsuhiko Nakamoto, Yuexiang Zhai, Anikait Singh, Yi Ma, Chelsea Finn, Aviral Kumar, and Sergey Levine. Cal-ql: Calibrated offline rl pre-training for efficient online fine-tuning. In *Workshop on Reincarnating Reinforcement Learning at ICLR 2023*, 2023.
- [31] Haichao Zhang, Wei Xu, and Haonan Yu. Policy expansion for bridging offline-to-online reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2022.
- [32] Michael Bain and Claude Sammut. A framework for behavioural cloning. In *Machine Intelligence 15*, pages 103–129, 1995.
- [33] Seyed Kamyar Seyed Ghasemipour, Dale Schuurmans, and Shixiang Shane Gu. Emaq: Expected-max q-learning operator for simple yet effective offline and online rl. In *International Conference on Machine Learning*, pages 3682–3691. PMLR, 2021.
- [34] Jialong Wu, Haixu Wu, Zihan Qiu, Jianmin Wang, and Mingsheng Long. Supported policy optimization for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:31278–31291, 2022.
- [35] Jianxiong Li, Xianyuan Zhan, Haoran Xu, Xiangyu Zhu, Jingjing Liu, and Ya-Qin Zhang. Distance-sensitive offline reinforcement learning. In *Deep Reinforcement Learning Workshop NeurIPS 2022*, 2022.
- [36] 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.
- [37] David Brandfonbrener, William F Whitney, Rajesh Ranganath, and Joan Bruna. Quantile filtered imitation learning. *arXiv preprint arXiv:2112.00950*, 2021.
- [38] 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.
- [39] Xinyue Chen, Zijian Zhou, Zheng Wang, Che Wang, Yanqiu Wu, and Keith Ross. Bail: Best-action imitation learning for batch deep reinforcement learning. *Advances in Neural Information Processing Systems*, 33:18353–18363, 2020.
- [40] Scott Emmons, Benjamin Eysenbach, Ilya Kostrikov, and Sergey Levine. Rvs: What is essential for offline RL via supervised learning? In *International Conference on Learning Representations*, 2022.
- [41] Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34:1273–1286, 2021.
- [42] Rui Yang, Yiming Lu, Wenzhe Li, Hao Sun, Meng Fang, Yali Du, Xiu Li, Lei Han, and Chongjie Zhang. Rethinking goal-conditioned supervised learning and its connection to offline RL. In *International Conference on Learning Representations*, 2022.
- [43] Yixiu Mao, Hongchang Zhang, Chen Chen, Yi Xu, and Xiangyang Ji. Supported value regularization for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.
- [44] Jing Zhang, Chi Zhang, Wenjia Wang, and Bingyi Jing. Constrained policy optimization with explicit behavior density for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.
- [45] Ofir Nachum, Mohammad Norouzi, Kelvin Xu, and Dale Schuurmans. Bridging the gap between value and policy based reinforcement learning. *Advances in neural information processing systems*, 30, 2017.
- [46] Alan Chan, Hugo Silva, Sungsu Lim, Tadashi Kozuno, A Rupam Mahmood, and Martha White. Greedification operators for policy optimization: Investigating forward and reverse kl divergences. *The Journal of Machine Learning Research*, 23(1):11474–11552, 2022.
- [47] Denis Tarasov, Alexander Nikulin, Dmitry Akimov, Vladislav Kurenkov, and Sergey Kolesnikov. CORL: Research-oriented deep offline reinforcement learning library. In *3rd Offline RL Workshop: Offline RL as a "Launchpad"*, 2022.



- [48] Alex Beeson and Giovanni Montana. Improving td3-bc: Relaxed policy constraint for offline learning and stable online fine-tuning. *arXiv preprint arXiv:2211.11802*, 2022.
- [49] Liyuan Mao, Haoran Xu, Weinan Zhang, and Xianyuan Zhan. Odice: Revealing the mystery of distribution correction estimation via orthogonal-gradient update. *arXiv preprint arXiv:2402.00348*, 2024.
- [50] Harshit Sikchi, Qinqing Zheng, Amy Zhang, and Scott Niekum. Dual rl: Unification and new methods for reinforcement and imitation learning. *arXiv preprint arXiv:2302.08560*, 2023.



## A Related Works

In the offline RL, policy constraint is the most straightforward and intuitive method for avoiding the issue of distributional shift. The core idea is to ensure that the learning policy cannot be too far away from the behavioral policy. Based on BC [32] that actions are selected from the dataset and cloned, some attempts are made to ensure that the distribution is bounded by carefully parameterizing the learning policy [23, 33]. In addition, several works explicitly apply divergence penalty terms to help adjust the learning policy, e.g., KL divergence [29] and MMD distance [7]. A general framework is proposed to evaluate these divergence-dependent methods [8]. More recently, other problem-specific regularization terms have been found to be more effective. These can be accomplished by adding a simple behavior cloning term that produces exceptional performance [11], regularizing behavior density directly to put more precise constraints [34], or generalizing OOD samples within a convex hull of training data using the distance function [35].

The policy constraint of these works can generally be assumed to be implemented based on the behavioral cloned policy. When data quality is high, BC is capable of achieving competitive performance thus the above methods can achieve satisfactory performance. However, a naive behavioral cloning approach is unworkable when the dataset is collected by a sub-optimal or multi-modal policy. To improve the quality of the behavioral policy used for constraint, several alternatives have been proposed. Weighted BC is a way to clone only the available best behavioral actions by down-weighting low-quality data or simply discarding them. Many works evaluate the relative advantages of transitions and rely on them to filter the most promising ones in the dataset [36, 37, 10, 12] or pick the trajectory samples based on Monte Carlo returns [38, 39, 9]. Conditional BC introduces an additional condition as the input of BC. [40] demonstrates the effectiveness of conditional BC and shows that an appropriate conditional variable is crucial when offline RL takes a supervised learning approach. These approaches use return-to-go as a condition to make better sequential decisions [14, 41] or select a future goal state as a condition and can extract the behavior of reaching the goal from sparse rewards [40, 42]. Furthermore, a recent research [26] learns guide-policy and execute-policy separately and uses the state generated by guide-policy to steer where execute-policy goes. [28] proposed to employ the diffusion model as a highly expressive policy class and use it to constrain the learning policy, which achieves the state-of-the-art performance among the policy constraint offline RL methods. CPI shares the same idea of improving the policy used for constraints, but CPI does not need to employ complex policy models such as diffusion model, still achieving similar or even better performance while greatly saving computational overhead.

In addition, there are several previous works aiming at achieving in-sample optimality [12, 34, 43, 44]. These methods adopt different methods to address the extrapolation error in offline reinforcement learning. For instance, SPOT [34] directly models the support set of the behavior policy, CPED [44] utilizes a GAN model for explicit density estimation, IQL [12] avoids querying values of unseen actions by implicitly approximating the policy improvement step, and SVR [43] uses the bias of importance sampling to calculate the penalty term. While CPI's core motivation is to enhance the robustness of behavior regularization methods by iteratively refining the reference policy used for regularization. This approach aims to progressively improve the policy's performance without significantly altering existing implementations. In addition, CPI is grounded in the conservative policy iteration, which guarantees progressive policy improvement and eventual convergence to the in-sample optimal policy within the support of the reference policy. SPOT, CPED, IQL, and SVR also have their theoretical foundations. For example, SPOT [34] is based on the theoretical formalization of density-based support constraints, CPED [44] provides theoretical results for the GAN estimator and performance guarantees, IQL [12] relies on the expectile regression of function approximators, and SVR [43] demonstrates the contractive property of its policy evaluation operator.

A previous work named STR [20] shares the similar idea with CPI. The key difference between the two algorithms is CPI uses reverse KL to optimize policy while STR uses forward KL to optimize policy and the resulting algorithms of CPI and STR are quite distinct. Note that we discuss the implementations between the two KL approaches in Appendix C. STR could be seen as adding an importance sampling weight to the Equation 40 in our paper's appendix.

## B Proofs

We first introduce some technical lemmas that will be used in the proof.

We consider a  $k$ -armed one-step decision making problem. Let  $\Delta$  be a  $k$ -dimensional simplex and  $\mathbf{q} = (q(1), \dots, q(k)) \in \mathbb{R}^k$  be the reward vector. Maximum entropy optimization considers

$$\max_{\pi \in \Delta} \pi \cdot \mathbf{q} + \tau \mathbb{H}(\pi). \quad (11)$$

The next result characterizes the solution of this problem (Lemma 4 of [45]).

**Lemma 1.** For  $\tau > 0$ , let

$$F_{\tau}(\mathbf{q}) = \tau \log \sum_a e^{q(a)/\tau}, \quad f_{\tau}(\mathbf{q}) = \frac{e^{\mathbf{q}/\tau}}{\sum_a e^{q(a)/\tau}} = e^{\frac{\mathbf{q} - F_{\tau}(\mathbf{q})}{\tau}}. \quad (12)$$



Then there is

$$F_\tau(\mathbf{q}) = \max_{\pi \in \Delta} \pi \cdot \mathbf{q} + \tau \mathbb{H}(\pi) = f_\tau(\mathbf{q}) \cdot \mathbf{q} + \tau \mathbb{H}(f_\tau(\mathbf{q})). \quad (13)$$

The second result gives the error decomposition of applying the Polite algorithm to compute an optimal policy. This result is adopted from [17].

**Lemma 2.** Let  $\pi_0$  be the uniform policy and consider running the following iterative algorithm on a MDP for  $t \geq 0$ ,

$$\pi_{t+1}(a|s) \propto \pi_t(a|s) \exp\left(\frac{q^{\pi_t}(a|s)}{\tau}\right), \quad (14)$$

Then

$$v^*(s) - v^{\pi_t}(s) \leq \frac{1}{(1-\gamma)^2} \sqrt{\frac{2 \log |\mathcal{A}|}{t}}. \quad (15)$$

*Proof.* We use vector and matrix operations to simplify the proof. In particular, we use  $v^\pi \in \mathbb{R}^{|\mathcal{S}|}$  and  $q^\pi \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{A}|}$ . Let  $P \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{A}| \times |\mathcal{S}|}$  be the transition matrix, and  $P^\pi \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{S}|}$  be the transition matrix between states when applying the policy  $\pi$ .

We first apply the following error decomposition

$$v^{\pi^*} - v^{\pi_{k-1}} = v^{\pi^*} - \frac{1}{k} \sum_{i=0}^{k-1} v^{\pi_i} + \frac{1}{k} \sum_{i=0}^{k-1} v^{\pi_i} - v^{\pi_{k-1}}. \quad (16)$$

For the first part,

$$v^{\pi^*} - \frac{1}{k} \sum_{i=0}^{k-1} v^{\pi_i} \quad (17)$$

$$= \frac{1}{k} \sum_{i=0}^{k-1} (I - \gamma P^{\pi^*})^{-1} (T^{\pi^*} v^{\pi_i} - v^{\pi_i}) \quad (18)$$

$$= \frac{1}{k} \sum_{i=0}^{k-1} (I - \gamma P^{\pi^*})^{-1} (M^{\pi^*} - M^{\pi_i}) q^{\pi_i} \quad (19)$$

$$\leq \frac{1}{(1-\gamma)^2} \sqrt{\frac{2 \log |\mathcal{A}|}{k}}, \quad (20)$$

where Equation (18) follows by the value difference lemma, Equation (20) follows by applying the regret bound of mirror descent algorithm for policy optimization. For the second part,

$$\frac{1}{k} \sum_{i=0}^{k-1} v^{\pi_i} - v^{\pi_{k-1}} \quad (21)$$

$$= \frac{1}{k} \sum_{i=0}^{k-1} (I - \gamma P^{\pi_{k-1}})^{-1} (v^{\pi_i} - T^{\pi_{k-1}} v^{\pi_i}) \quad (22)$$

$$= \frac{1}{k} \sum_{i=0}^{k-1} (I - \gamma P^{\pi_{k-1}})^{-1} (M^{\pi_i} - M^{\pi_{k-1}}) q^{\pi_i} \quad (23)$$

$$\leq 0 \quad (24)$$

where for the last step we use that for any  $s \in \mathcal{S}$ ,  $\sum_{i=0}^{k-1} (\pi_i - \pi_{k-1}) \hat{q}_i \leq 0$ . This follows by

$$\sum_{i=0}^{k-1} \pi_{k-1} \hat{q}_i = \sum_{i=0}^{k-2} \pi_{k-1} \hat{q}_i + \pi_{k-1} \hat{q}_{k-1} + \tau \mathcal{H}(\pi_{k-1}) - \tau \mathcal{H}(\pi_{k-1}) \quad (25)$$

$$\geq \sum_{i=0}^{k-2} \pi_{k-2} \hat{q}_i + \pi_{k-1} \hat{q}_{k-1} + \tau \mathcal{H}(\pi_{k-2}) - \tau \mathcal{H}(\pi_{k-1}) \quad (26)$$

$$\geq \dots \quad (27)$$

$$\geq \sum_{i=0}^{k-1} \pi_i \hat{q}_i + \tau \mathcal{H}(\pi_0) - \tau \mathcal{H}(\pi_{k-1}) \quad (28)$$

$$\geq \sum_{i=0}^{k-1} \pi_i \hat{q}_i, \quad (29)$$



where Equation (26) follows by applying Lemma 1 and the definition of  $\pi_{k-1}$ , Equation (29) follows by the definition of  $\pi_0$ . Combine the above together finishes the proof.  $\square$

*Proof of Theorem 1.* First recall the in-sample optimality equation

$$q_{\pi_D}^*(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s, a)} \left[ \max_{a': \pi_D(a'|s') > 0} q_{\pi_D}^*(s', a') \right], \quad (30)$$

which could be viewed as the optimal value of a MDP  $M_D$  covered by the behavior policy  $\pi_D$ , where  $M_D$  only contains transitions starting with  $(s, a) \in \mathcal{S} \times \mathcal{A}$  such that  $\pi_D(a|s) > 0$ . Then the result can be proved by two steps. First, note that the CPI algorithm will never consider actions such that  $\pi_D(a|s) = 0$ . This is directly implied by Lemma 1. Second, we apply Lemma 2 to show the error bound of using CPI on  $M_D$ . This finishes the proof.  $\square$

**Proposition 1.** *let  $\bar{\pi}^*$  be the optimal policy of (5). For any  $s \in \mathcal{S}$ , we have that  $V^{\bar{\pi}^*}(s) \geq V^{\bar{\pi}}(s)$ ; and  $\bar{\pi}^*(a|s) = 0$  given  $\bar{\pi}(a|s) = 0$ .*

*Proof of Proposition 1.* We first prove the first part.

$$\mathbb{E}_{a \sim \bar{\pi}^*} [Q^{\bar{\pi}}(s, a)] \geq \mathbb{E}_{a \sim \bar{\pi}^*} [Q^{\bar{\pi}}(s, a)] - \tau D_{\text{KL}}(\bar{\pi}^*(s) || \bar{\pi}(s)) \quad (31)$$

$$\geq \mathbb{E}_{a \sim \bar{\pi}} [Q^{\bar{\pi}}(s, a)] - \tau D_{\text{KL}}(\bar{\pi}(s) || \bar{\pi}(s)) \quad (32)$$

$$= \mathbb{E}_{a \sim \bar{\pi}} [Q^{\bar{\pi}}(s, a)], \quad (33)$$

where the first inequality follows the non-negativity of KL divergence, the second inequality follows  $\bar{\pi}^*$  is the optimal policy of (34). Then the first statement can be proved by applying a standard recursive argument. The second statement is directly implied by Lemma 1.

$$\max_{\pi} \mathbb{E}_{a \sim \pi} [Q^{\bar{\pi}}(s, a)] - \tau D_{\text{KL}}(\pi(s) || \bar{\pi}(s)) \quad (34)$$

$\square$

## C CPI with forward KL

We show how to optimize (34) using the *forward KL*. Extension to (9) is straightforward by picking  $\bar{\pi} = \pi_t$ .

Recall (34),

$$\max_{\pi} \mathbb{E}_{a \sim \pi} [Q^{\bar{\pi}}(s, a)] - \tau D_{\text{KL}}(\pi(s) || \bar{\pi}(s)) \quad (35)$$

By Lemma 1, the optimal policy  $\bar{\pi}^*$  has a closed form solution.

$$\bar{\pi}^*(a|s) \propto \bar{\pi}(a|s) \exp\left(\frac{Q^{\bar{\pi}}(s, a)}{\tau}\right). \quad (36)$$

This implies that to optimize (34), we can also consider the following

$$\min_{\pi} D_{\text{KL}}(\bar{\pi}^* || \pi) \propto -\bar{\pi}^* \log \pi \quad (37)$$

$$= \mathbb{E}_{a \sim \bar{\pi}^*} [\log \pi(a|s)] \quad (38)$$

$$= \mathbb{E}_{a \sim \bar{\pi}} \left[ \frac{\bar{\pi}^*(a|s)}{\bar{\pi}(a|s)} \log \pi(a|s) \right] \quad (39)$$

$$= \mathbb{E}_{a \sim \bar{\pi}} \left[ \frac{\exp(Q^{\bar{\pi}}(s, a)/\tau)}{Z(s)} \log \pi(a|s) \right]. \quad (40)$$

The first step follows by removing terms not dependent on  $\pi$ .  $Z(s) = \sum_a \bar{\pi}(a|s) \exp(Q^{\bar{\pi}}(s, a))$  is a normalization term. In practice, it is often approximated by a state value function [13, 12, 29].

**Connection to (34)** We now discuss how to connect the forward KL objective described above with the original optimization problem (34). Indeed, it can be verified that (34) corresponds to a *reverse KL* policy



optimization,

$$\arg \min_{\pi} D_{\text{KL}}(\pi || \bar{\pi}^*) = \arg \min_{\pi} \pi \log \pi - \pi \log \bar{\pi}^* \quad (41)$$

$$= \arg \min_{\pi} \pi \log \pi - \pi \log \frac{\bar{\pi} \exp(Q^{\bar{\pi}}/\tau)}{Z} \quad (42)$$

$$= \arg \min_{\pi} \pi \log \pi - \pi \log \bar{\pi} - \pi Q^{\bar{\pi}}/\tau + \log Z \quad (43)$$

$$= \arg \min_{\pi} D_{\text{KL}}(\pi || \bar{\pi}) - \pi Q^{\bar{\pi}}/\tau \quad (44)$$

$$= \arg \max_{\pi} \pi Q^{\bar{\pi}} - \tau D_{\text{KL}}(\pi || \bar{\pi}). \quad (45)$$

where we use vector notations for simplicity. Although forward and reverse KL has exactly the same solution in the tabular case, when function approximation is applied these two objectives can showcase different optimization properties. We refer to [46] for more discussions on these two objectives.

## D Detailed Experimental Settings

In this section we provide the complete details of the experiments in our paper.

### D.1 The effect of different regularizations

We analyze the impact on the algorithm when different policies are used as regularization terms. The most straightforward way to obtain policies with different performances is the baseline method  $X\%BC$  mentioned in DT [14]. We set  $x$  to 5 in order to make the difference between policies more significant and let the 5%BC policy be the policy used for constraint. In detail, we selectively choose different 5% data for behavioral cloning so that we can get various policies with different performances. First, we sort the trajectories in the dataset by their return (accumulated rewards) and select three different levels of data: top (highest return), middle or bottom (lowest return). Each level of data is sampled at 5% of the total data volume. Then we can train 5%BC using the MLP network via BC and get three 5%BC policies with different performances. We can then use each 5%BC policy as the regularization term of TD3 to implement the TD3+5%BC method. Besides, we also normalize the states and set the regularization parameter  $\alpha$  to 2.5. The only difference between the TD3+5%BC and TD3+BC is the action obtained in the regularization term. In TD3+BC, the action used for constraint is directly obtained from the dataset according to the corresponding state in the dataset, while in TD3+5%BC, the action for constraint is sampled using 5%BC. We set the training steps for 5% BC to 5e5 and the training steps for TD3+5%BC to 1M. Table 2 concludes the hyperparameters of 5% BC. The hyperparameters of TD3+5%BC are the same as those of TD3+BC [11].

Table 2: 5% BC Hyperparameters

Hyperparameter	Value
Hidden layers	3
Hidden dim	256
Activation function	ReLU
Mini-batch size	256
Optimizer	Adam
Dropout	0.1
Learning rate	3e-4

### D.2 Baselines

We conduct experiments on the benchmark of D4RL and use Gym-MuJoCo datasets of version v2, Antmaze datasets of version v0, and Adroit datasets of version v1. We compare CPI with BC, DT [14], TD3+BC [11], CQL [25], IQL [12], POR [26], EDAC [27], Diffusion-QL [28] and InAC [13]. In Gym-Mujoco tasks, our experimental results are preferentially selected from EDAC, Diffusion-QL papers, or their original papers. If corresponding results are unavailable from these sources, we rerun the code provided by the authors. Specifically, we run it on the expert dataset for POR<sup>4</sup>. For DT<sup>5</sup> and IQL<sup>6</sup>, we follow the hyperparameters given by the authors to run on random and expert datasets. For Diffusion-QL<sup>7</sup>, we set the hyperparameters for the random dataset to be the same as on the medium-replay dataset and the hyperparameters for the expert dataset to be the same as on the medium-expert dataset according to the similarity between these datasets. For InAC<sup>8</sup>, we run it on the random dataset. In Antmaze tasks, our experimental results are taken from the Diffusion-QL paper, except for EDAC<sup>9</sup> and POR. The results of EDAC are obtained by running the authors' provided code and setting the Q

<sup>4</sup><https://github.com/ryanxhr/POR>

<sup>5</sup><https://github.com/kzl/decision-transformer>

<sup>6</sup>[https://github.com/ikostrikov/implicit\\_policy\\_improvement](https://github.com/ikostrikov/implicit_policy_improvement)

<sup>7</sup><https://github.com/Zhendong-Wang/Diffusion-Policies-for-Offline-RL>

<sup>8</sup>[https://github.com/hwang-ua/inac\\_pytorch](https://github.com/hwang-ua/inac_pytorch)

<sup>9</sup><https://github.com/snu-mlab/EDAC>



Table 3: CPI Hyperparameters

	Hyperparameter	Value
Architecture	Actor hidden layers	3
	Actor hidden dim	256
	Actor activation function	ReLU
	Critic hidden layers	3
	Critic hidden dim	256
	Critic activation function	ReLU
Learning	Optimizer	Adam
	Critic learning rate	3e-4 for MuJoCo and Adroit 1e-3 for Antmaze
	Actor learning rate	3e-4
	Mini-batch size	256
	Discount factor	0.99 for MuJoCo and Adroit 0.995 for Antmaze
	Target update rate	5e-3
	Policy noise	0.2
	Policy noise clipping	(-0.5, 0.5)
	$\tau$	{0.05, 0.2, 1, 2} for MuJoCo {0.03, 0.05, 0.1, 1} for Antmaze
		{100, 200} for Adroit
	$\lambda$	{0.5, 0.7}

ensemble number to 10 and  $\eta = 1.0$ . Even when we transformed the rewards according to [25], the performance of EDAC on Antmaze still did not perform well, which matches the report in the offline reinforcement learning library CORL [47]. As the results in the POR paper are under Antmaze v2, we rerun them under Antmaze v0. In the Adroit task, we rerun the experiment for TD3+BC, DT (with return-to-go set to 3200), POR (with the same parameters as the antmaze tasks), and InAC (with tau set to 0.7).

### D.3 Conservative Policy iteration

To implement our idea, we made slight modifications to TD3+BC<sup>10</sup> to obtain CPI. For CPI, we select a historical snapshot policy as the reference policy. Specifically, the policy snapshot  $\pi^{k-2}$ , which is two gradient steps before the current step, is chosen as the reference policy for the current learning policy  $\pi^k$ . CPI is trained similarly to TD3. For CPI-RE, the complete network contains two identical policy networks with different initial parameters, so that two policies with distinct performances can be obtained. The two policy networks in CPI-RE are updated via cross-update to fully utilize the information from both value networks. During training, the value network evaluates actions induced by the two policy networks, and only the higher value action is used to pull up the performance of the learning policy. During evaluation, the two policy networks are also used to select high-value actions to interact with the environment. CPI only has one more actor compared to TD3+BC and therefore requires less computational overhead than other state-of-the-art offline RL algorithms with complex algorithmic architectures, such as EDAC (an ensemble of Q functions) and Diffusion-QL (Diffusion model).

According to TD3+BC, we normalize the state of the MuJoCo tasks and use the original rewards in the dataset. For the Antmaze datasets, we ignore the state normalization techniques and transform the rewards in the dataset according to [25]. For Adroit datasets, we also do not use state normalization and standardize the rewards according to Diffusion-QL [28]. To get the reported results, we average the returns of 10 trajectories with five random seeds evaluated every 5e3 steps for MuJoCo and Adroit, 100 trajectories with five random seeds evaluated every 5e4 steps for Antmaze. When training on Antmaze, it's observed that the randomness can heavily influence the algorithm performance as reported in [28]. Compared with the one in [28] that violates the setting of offline learning and selects model using finite online samples, we train several models using different random seeds and randomly choose five models of which the Q-value is not divergent and report the average returns of 100 trajectories with these five models evaluated every 5e4 steps. In addition, we evaluate the runtime and the memory consumption of different algorithms to train an epoch (1000 gradient update steps). All experiments are run on a GeForce GTX 2080TI GPU.

The most critical hyperparameters in CPI are the weight coefficient  $\lambda$  and the regularization parameter  $\tau$ . On all datasets, the choice of  $\lambda = 0.5$  or  $\lambda = 0.7$  is the most appropriate so that the two actions (from the behavioral policy  $\beta$  and the reference policy  $\bar{\pi}$ ) can be well-weighted to participate in the learning process. As mentioned in the main text, the choice of  $\tau$  depends heavily on the characteristics of the dataset. For a high-quality dataset,

<sup>10</sup>[https://github.com/sfujim/TD3\\_BC](https://github.com/sfujim/TD3_BC)



Table 4: Regularization parameter  $\tau$  and weighting factor  $\lambda$  of CPI for all datasets. On most datasets, CPI and CPI-RE have the same optimal parameters. On some datasets, the values in parentheses indicate the parameters of CPI, and the values outside the parentheses indicate the parameters of CPI-RE.

Dataset	regularization parameter $\tau$	weighting coefficient $\lambda$
halfcheetah-random-v2	0.05	0.7
halfcheetah-medium-v2	0.05	0.7
halfcheetah-medium-replay-v2	0.05	0.7
halfcheetah-medium-expert-v2	2	0.5
halfcheetah-expert-v2	1	0.5
hopper-random-v2	0.05	0.5
hopper-medium-v2	0.2	0.5
hopper-medium-replay-v2	0.05	0.5
hopper-medium-expert-v2	1	0.5(0.7)
hopper-expert-v2	1	0.5(0.7)
walker2d-random-v2	2	0.5
walker2d-medium-v2	1	0.7
walker2d-medium-replay-v2	0.2	0.7(0.5)
walker2d-medium-expert-v2	1	0.7
walker2d-expert-v2	1	0.7
antmaze-umaze-v0	1	0.7
antmaze-umaze-diverse-v0	0.1	0.5
antmaze-medium-play-v0	0.1	0.5
antmaze-medium-diverse-v0	0.1(0.05)	0.5
antmaze-large-play-v0	0.03	0.5(0.7)
antmaze-large-diverse-v0	0.05	0.5
pen-human-v1	100(200)	0.5(0.7)
pen-cloned-v1	100	0.7

$\tau$  should be larger to learn in a close imitation way, and for a high-diversity dataset, the  $\tau$  should be chosen to be smaller to make the whole learning process more similar to RL. We find that training is more stable and better performance can be achieved on Antmaze when the critic learning rate is set to  $1e-3$ . Also, since Antmaze is a sparse reward domain, we also set the discount factor to 0.995. Table 4 gives our selections of hyperparameters  $\tau$  and  $\lambda$  on different datasets. Other settings are given in Table 3. Codes are provided in this link <https://github.com/mamengyiyi/CPI>.

## E More experimental results

### E.1 Effect of different numbers of reference policies

We also present the effect of using different numbers of actors in CPI-RE. We can conclude from Figure 8 that increasing the actor number from 1 to 2 (i.e., introducing the reference policy) can significantly improve the performance of the learning policy. As further introducing reference policy derived from more actors does not bring significant benefits and entails significant resource consumption. Thus, we set the actor number in our method to two.

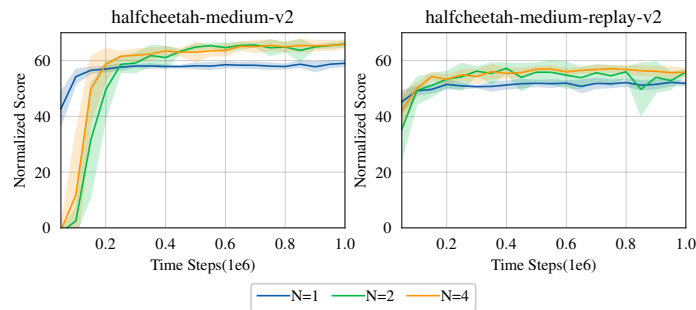


Figure 8: Effect of different actor number



## E.2 Results of Online Fintuning

We finetune CPI during online training after training on nine offline datasets including halfcheetah-medium-v2, halfcheetah-medium-replay-v2, halfcheetah-medium-expert-v2, hopper-medium-v2, hopper-medium-replay-v2, hopper-medium-expert-v2, walker2d-medium-v2, walker2d-medium-replay-v2 and walker2d-medium-expert-v2. Detailed results provided in Figure 9 show that CPI with online finetuning can outperform the state-of-the-art offline-to-online algorithms Cal-QL [30] and PEX [31].

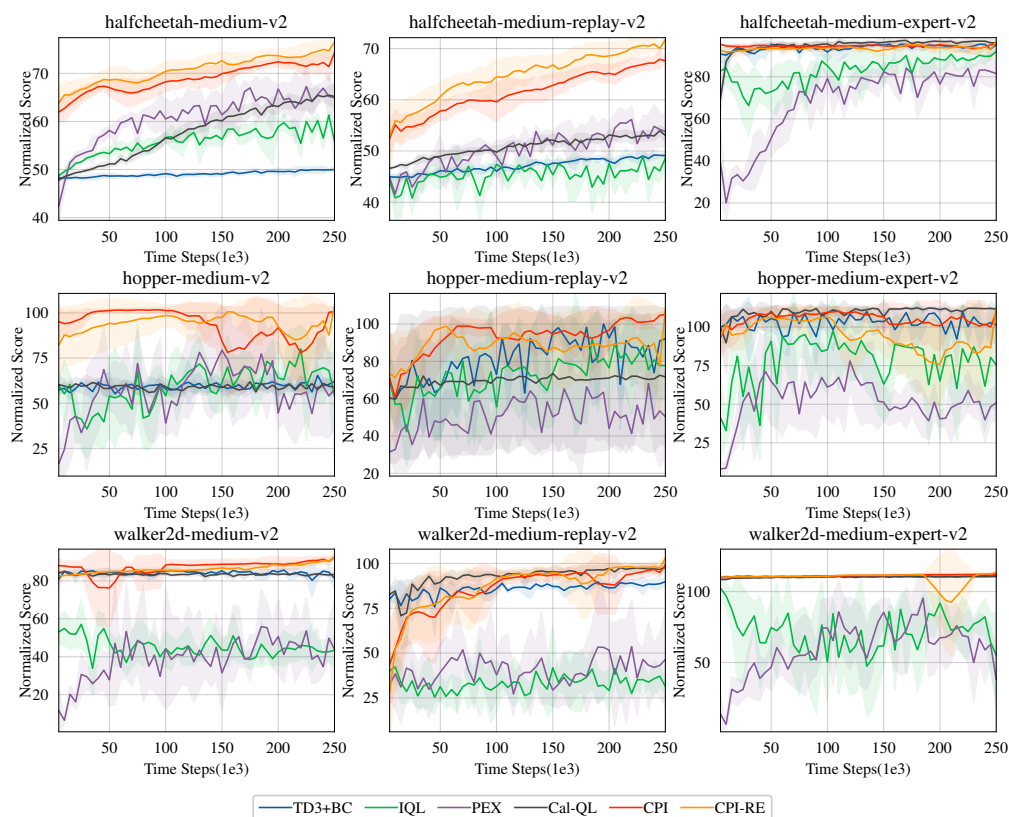


Figure 9: Performance of CPI offline training after further online fine-tuning

## E.3 CPI with minimal hyper-parameter tuning

We provide results of CPI with as little as possible hyper-parameter tuning on the same domain. For MuJoCo datasets, we set  $\tau$  to 0.05 or 1.0,  $\lambda$  to 0.7. For Antmaze, we set  $\tau$  to 0.1 and  $\lambda$  to 0.5. For Antmaze, we set  $\tau$  to 200.0 and  $\lambda$  to 0.7. The detailed results (average across five seeds) and setting are provided in the following table 5. The results show that compared with IQL, CQL and TD3BC, the overall performance of CPI with as little as possible hyper-parameter tuning is still significantly better.

## E.4 Effects of various techniques used in CPI

Here we ablate the techniques including state normalization, reward transformation used in CPI.

In Antmaze, no state norm is used for CPI and TD3BC. Table 6 shows introducing reward transformation indeed help TD3BC on certain antmaze datasets, but the overall performance is not improved and still falls behind CPI.

In Mujoco, no reward transformation is used for CPI and TD3BC. Table 7 ablates the effectiveness of state normalization. With and without state normalization, CPI can still outperform TD3BC, demonstrating the effectiveness of the iteratively policy refinement.

## E.5 In-depth comparison of TD3+BC and CPI

According to the experimental part of the ablation study and the analysis in the [34],  $\alpha$  is an important parameter for controlling the constraint strength. TD3+BC is set  $\alpha$  to a constant value of 2.5 for each dataset, whereas



Table 5: The performance of CPI and CPI-RE with as little as possible hyper-parameter (hp) tuning.

Dataset	$\tau$	$\lambda$	TD3+BC	CQL	IQL	CPI with minimal hp tuning	CPI
halfcheetah-random	0.05	0.7	11.0	31.3	13.7	29.7 $\pm$ 1.1	29.7 $\pm$ 1.1
halfcheetah-medium	0.05	0.7	48.3	46.9	47.4	64.4 $\pm$ 1.3	64.4 $\pm$ 1.3
halfcheetah-medium-replay	0.05	0.7	44.6	45.3	44.2	54.6 $\pm$ 1.3	54.6 $\pm$ 1.3
halfcheetah-medium-expert	1.0	0.7	90.7	95.0	86.7	89.7 $\pm$ 1.5	94.7 $\pm$ 1.1
halfcheetah-expert	1.0	0.7	96.7	97.3	94.9	98.1 $\pm$ 0.6	96.5 $\pm$ 0.2
hopper-random	0.05	0.7	8.5	5.3	8.4	29.5 $\pm$ 3.8	29.5 $\pm$ 3.7
hopper-medium	1.0	0.7	59.3	61.9	66.3	61.4 $\pm$ 1.9	98.5 $\pm$ 3.0
hopper-medium-replay	0.05	0.7	60.9	86.3	94.7	99.3 $\pm$ 6.8	101.7 $\pm$ 1.6
hopper-medium-expert	1.0	0.7	98.0	96.9	91.5	106.4 $\pm$ 4.3	106.4 $\pm$ 4.3
hopper-expert	1.0	0.7	107.8	106.5	108.8	112.2 $\pm$ 0.5	112.2 $\pm$ 0.5
walker2d-random	1.0	0.7	1.6	5.4	5.9	3.9 $\pm$ 2.2	5.9 $\pm$ 1.7
walker2d-medium	1.0	0.7	83.7	79.5	78.3	85.8 $\pm$ 0.8	85.8 $\pm$ 0.8
walker2d-medium-replay	1.0	0.7	81.8	76.8	73.9	89.1 $\pm$ 3.0	91.8 $\pm$ 2.9
walker2d-medium-expert	1.0	0.7	110.1	109.1	109.6	110.9 $\pm$ 0.4	110.9 $\pm$ 0.4
walker2d-expert	1.0	0.7	110.2	109.3	109.7	110.6 $\pm$ 0.1	110.6 $\pm$ 0.1
mujoco-total	-	-	1013.2	1052.8	1034.0	1145.6	1193.2
antmaze-umaze	0.1	0.5	78.6	74.0	87.5	96.6 $\pm$ 3.2	98.8 $\pm$ 1.1
antmaze-umaze-diverse	0.1	0.5	71.4	84.0	62.2	88.6 $\pm$ 5.7	88.6 $\pm$ 5.7
antmaze-medium-play	0.1	0.5	3.0	61.1	71.2	82.4 $\pm$ 5.8	82.4 $\pm$ 5.8
antmaze-medium-diverse	0.1	0.5	10.6	53.7	70.0	78.8 $\pm$ 8.9	80.4 $\pm$ 8.9
antmaze-large-play	0.1	0.5	0.0	15.8	39.6	3.3 $\pm$ 5.7	20.6 $\pm$ 16.3
antmaze-large-diverse	0.1	0.5	0.2	14.9	47.5	23.0 $\pm$ 4.3	45.2 $\pm$ 6.9
antmaze-total	-	-	163.8	303.6	378.0	372.7	416.0
pen-human	200.0	0.7	-1.9	35.2	71.5	80.1 $\pm$ 16.9	80.1 $\pm$ 16.9
pen-cloned	200.0	0.7	9.6	27.2	37.3	69.7 $\pm$ 17.0	71.8 $\pm$ 35.2
adroit-total	-	-	7.7	62.4	108.8	149.8	151.9

Table 6: Ablation of reward transformation.

datasets	TD3BC with reward transformation	TD3BC	CPI
antmaze-umaze	87.8 $\pm$ 5.2	78.6 $\pm$ 33.3	98.8 $\pm$ 1.1
antmaze-umaze-diverse	69.8 $\pm$ 11.1	71.4 $\pm$ 20.7	88.6 $\pm$ 5.7
antmaze-medium-play	0.0 $\pm$ 0.0	10.6 $\pm$ 10.1	82.4 $\pm$ 5.8
antmaze-medium-diverse	0.6 $\pm$ 1.9	3.0 $\pm$ 4.8	80.4 $\pm$ 8.9
antmaze-large-play	0.0 $\pm$ 0.0	0.2 $\pm$ 0.4	20.6 $\pm$ 16.3
antmaze-large-diverse	0.2 $\pm$ 0.4	0.0 $\pm$ 0.0	45.2 $\pm$ 6.9
Total	158.4	163.8	416.0

Table 7: Ablation of state normalization.

datasets	CPI w/o state norm	CPI	TD3BC w/o state norm	TD3BC
halfcheetah-medium	63.5 $\pm$ 3.6	64.4 $\pm$ 1.3	48.1 $\pm$ 0.4	48.3 $\pm$ 0.3
halfcheetah-medium-replay	54.4 $\pm$ 4.5	54.6 $\pm$ 1.3	44.0 $\pm$ 0.5	44.6 $\pm$ 0.5
halfcheetah-medium-expert	94.0 $\pm$ 1.7	94.7 $\pm$ 1.1	89.4 $\pm$ 5.1	90.7 $\pm$ 4.3
halfcheetah-expert	96.4 $\pm$ 0.7	96.5 $\pm$ 0.2	96.1 $\pm$ 1.0	96.7 $\pm$ 1.1
hopper-medium	94.2 $\pm$ 9.7	98.5 $\pm$ 3.0	59.5 $\pm$ 1.2	59.3 $\pm$ 4.2
hopper-medium-replay	100.7 $\pm$ 3.2	101.7 $\pm$ 1.6	56.7 $\pm$ 23.6	60.9 $\pm$ 18.8
hopper-medium-expert	104.7 $\pm$ 6.7	106.4 $\pm$ 4.3	98.4 $\pm$ 4.9	98.0 $\pm$ 9.4
hopper-expert	112.1 $\pm$ 0.3	112.2 $\pm$ 0.5	110.7 $\pm$ 1.7	107.8 $\pm$ 7.0
walker2d-medium	85.3 $\pm$ 0.5	85.8 $\pm$ 0.8	82.3 $\pm$ 3.3	83.7 $\pm$ 2.1
walker2d-medium-replay	89.4 $\pm$ 1.5	91.8 $\pm$ 2.9	83.0 $\pm$ 3.2	81.8 $\pm$ 5.5
walker2d-medium-expert	111.9 $\pm$ 0.3	110.9 $\pm$ 0.4	110.4 $\pm$ 0.5	110.1 $\pm$ 0.5
walker2d-expert	110.5 $\pm$ 0.1	110.6 $\pm$ 0.1	110.2 $\pm$ 0.3	110.2 $\pm$ 0.3
Total	1117.1	1128.1	988.8	992.1



CPI chooses the appropriate  $\tau$  from a set of  $\tau$  alternatives. We note that the hyperparameter that plays a role in regulating Q and regularization in CPI is  $\tau$ , which can essentially be understood as the reciprocal of  $\alpha$  in TD3+BC. Therefore, for the convenience of comparison, we rationalize the reciprocal of  $\tau$  as the parameter  $\alpha$ . In this section, we set the  $\alpha$  of TD3+BC to be consistent with that of CPI in order to show that the performance improvement of CPI mainly comes from amalgamating the benefits of both behavior-regularized and in-sample algorithms. Further, we also compare CPI with TD3+BC with dynamically changed  $\alpha$  [48] and TD3+BC with swept best  $\alpha$  in the ranges 0.0001, 0.05, 0.25, 2.5, 25, 36, 50, 100, which improves TD3+BC by a large margin, to show the superiority of CPI. The selection of parameters is shown in Table 4.

The results for TD3+BC (vanilla), TD3+BC (same  $\alpha$  with CPI), TD3+BC (swept best  $\alpha$ ), TD3+BC with dynamically changed  $\alpha$  and CPI are shown in Table 8. Comparing the variants of TD3+BC with different  $\alpha$  choices, it can be found that changing  $\alpha$  can indeed improve the performance of TD3+BC. However, compared with TD3+BC (same  $\alpha$ ) and TD3+BC (swept best  $\alpha$ ), the performance of CPI is significantly better, which proves the effectiveness of the mechanism for iterative refinement of policy for behavior regularization in CPI.

Table 8: Comparison with TD3+BC and its variants.

Dataset	TD3+BC (vanilla)	TD3+BC (same $\alpha$ with CPI)	TD3+BC (swept best $\alpha$ )	TD3+BC (dynamic $\alpha$ )	CPI	CPI-RE
halfcheetah-medium	48.3	58.8	58.8	55.3	64.4	65.9
hopper-medium	59.3	69.0	69.0	100.1	98.5	97.9
waker2d-medium	83.7	79.8	83.7	89.1	85.8	86.3
halfcheetah-medium-replay	44.6	51.2	51.2	48.7	54.6	55.9
hopper-medium-replay	60.9	88.5	88.5	100.5	101.7	103.2
waker2d-medium-replay	81.8	83.0	83.0	87.9	91.8	93.8
halfcheetah-medium-expert	90.7	92.3	92.3	91.9	94.7	95.6
hopper-medium-expert	98.0	76.1	98.0	103.9	106.4	110.1
waker2d-medium-expert	110.1	109.4	110.1	112.7	110.9	111.2
halfcheetah-expert	96.7	94.5	96.7	97.5	96.5	97.4
hopper-expert	107.8	110.8	110.8	112.4	112.2	112.3
waker2d-expert	110.2	109.3	110.2	113.0	110.6	111.2
Above Total	992.1	1022.7	1052.3	1113.0	1128.1	1140.8
halfcheetah-random	11.0	23.2	23.2	-	29.7	30.7
hopper-random	8.5	28.8	28.8	-	29.5	30.4
waker2d-random	1.6	4.4	4.4	-	5.9	5.5
Gym Total	1013.2	1079.1	1108.7	-	1193.3	1207.4
antmaze-umaze	78.6	93.8	93.8	-	98.8	99.2
antmaze-umaze-diverse	71.4	73.2	73.2	-	88.6	92.6
antmaze-medium-play	3.0	13.0	35.7	-	82.4	84.8
antmaze-medium-diverse	10.6	8.0	17.5	-	80.4	80.6
antmaze-large-play	0.0	0.0	0.0	-	20.6	33.6
antmaze-large-diverse	0.2	0.0	0.2	-	45.2	48.0
Antmaze Total	163.8	188.0	220.4	-	416.0	438.8

## E.6 Applying idea of CPI to SAC-BC

We implement SAC-BC similar to TD3-BC and integrate CPI into SAC-BC to see whether the idea of CPI could improve SAC-BC, which adding behavior cloning term to SAC algorithm. As we show in Table 9, on these six datasets, CPI could bring benefits to SAC-BC.

Table 9: Results of SAC-BC and SAC-BC with Policy Iteration.

Datasets	SAC-BC	SAC-BC with Policy Iteration
halfcheetah-medium	42.1 $\pm$ 0.8	51.4 $\pm$ 0.8
halfcheetah-medium-replay	38.0 $\pm$ 2.8	51.0 $\pm$ 1.0
hopper-medium	35.9 $\pm$ 1.4	40.5 $\pm$ 8.1
hopper-medium-replay	32.2 $\pm$ 7.9	36.4 $\pm$ 4.1
waker2d-medium	78.8 $\pm$ 4.2	59.6 $\pm$ 3.0
waker2d-medium-replay	75.8 $\pm$ 3.2	84.3 $\pm$ 3.8
Total	302.8	323.2

## E.7 Comparisons with more baselines

We additionally provide comparisons of our methods and more recently proposed baselines here.



Table 10: Comparison with more baselines

datasets	ODICE[49]	f-DVL[50]	SPOT[34]	CPED[44]	CPI	CPI-RE
halfcheetah-medium	47.4	47.5	58.4	61.8	64.4	65.9
hopper-medium	86.1	64.1	86.0	100.1	98.5	97.9
walker2d-medium	84.9	81.5	86.4	90.2	85.8	86.3
halfcheetah-medium-replay	44.0	44.7	52.2	55.8	54.6	55.9
hopper-medium-replay	99.9	98.0	100.2	98.1	101.7	103.2
walker2d-medium-replay	83.6	68.7	91.6	91.9	91.8	93.8
halfcheetah-medium-expert	93.2	91.2	86.9	85.4	94.7	95.6
hopper-medium-expert	110.8	93.3	99.3	95.3	106.4	110.1
walker2d-medium-expert	110.8	109.6	112.0	113.0	110.9	111.2
Gym-MuJoCo Total	760.7	698.6	773.0	791.7	808.8	819.9
antmaze-umaze	94.1	87.7	93.5	96.8	98.8	99.2
antmaze-umaze-diverse	79.5	48.4	40.7	55.6	88.6	92.6
antmaze-medium-play	86.0	71.0	74.7	85.1	82.4	84.8
antmaze-medium-diverse	82.7	60.2	79.1	72.1	80.4	80.6
antmaze-large-play	55.9	41.7	35.3	34.9	20.6	33.6
antmaze-large-diverse	54.0	39.3	36.3	32.3	45.2	48.0
Antmaze Total	452.2	348.3	359.6	376.8	416.0	438.8
Total	1212.9	1046.9	1132.6	1168.5	1224.8	1258.7

## E.8 Learning curves of CPI

The learning curves of CPI on all tasks are shown in Figure 10.

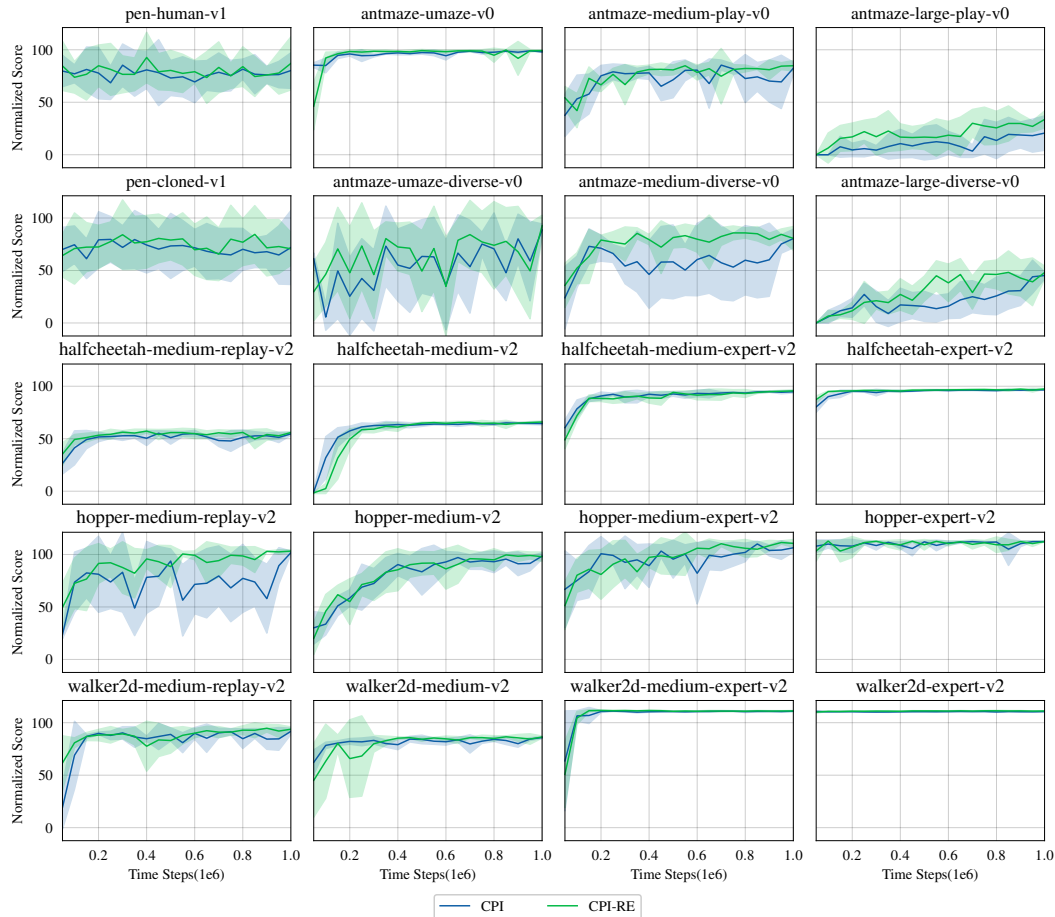


Figure 10: Learning curves of CPI and CPI-RE for Mujoco, Antmaze and Pen, evaluated every 5e3 steps.



## NeurIPS Paper Checklist

### 1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [Yes]

Justification: We proposed a novel offline rl algorithm, the core concept behind this approach hinges on the iterative refinement of the reference policy utilized for behavior regularization. We try our best to accurately describe our contributions in the Abstract and Introduction (Section 1). We also reiterated our contribution in the last paragraph of Introduction section

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

### 2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: See the conclusion section.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

### 3. Theory Assumptions and Proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: See appendix B.

Guidelines:

- The answer NA means that the paper does not include theoretical results.



- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

#### 4. Experimental Result Reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide code link and describe the hyperparameters setting in appendix D.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

#### 5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide code link and description of datasets in the experimental section and appendix D.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).



- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

## 6. Experimental Setting/Details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We describe experimental details in appendix D.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

## 7. Experiment Statistical Significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Results in our paper are depicted with 95% confidence intervals, represented by shaded areas in figures and expressed as standard deviations in tables.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

## 8. Experiments Compute Resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: See appendix D.

Guidelines:

- The answer NA means that the paper does not include experiments.



- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

#### 9. Code Of Ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [NA]

Justification: None

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

#### 10. Broader Impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: None

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

#### 11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: None

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.



- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

## 12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [NA]

Justification: None

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, [paperswithcode.com/datasets](https://paperswithcode.com/datasets) has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

## 13. New Assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: None

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

## 14. Crowdsourcing and Research with Human Subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: None

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

## 15. Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?



Answer: [NA]

Justification: None

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.