FLOW Q-TRACE: MULTI-STEP CRITIC LEARNING WITH DENSITY-RATIO-BASED OOD ACTION HANDLING

Anonymous authors
Paper under double-blind review

ABSTRACT

In this paper, we consider multi-step critic learning for offline reinforcement learning (RL), moving beyond the widely-used one-step temporal difference methods. Although multi-step learning can offer more accurate value estimation, it has been underutilized in offline RL settings due to two challenges: the difficulty of correcting multi-step off-policyness via importance sampling, and value overestimation caused by bootstrapping from out-of-distribution (OOD) actions. We address these challenges by introducing a conditional density ratio estimator and develop Q-trace, an offline multi-step critic learning algorithm with several desirable properties. In our framework, the learned density ratio estimator is employed not only for off-policyness correction but also for effective OOD action handling through critic penalization and rejection sampling, exploiting its capacity to identify OOD actions in a pointwise manner. By integrating Q-trace with expressive flow-based policies and our density-ratio-based OOD handling, we propose Flow Q-trace (FQT), which demonstrates strong performance across diverse offline RL tasks.

1 INTRODUCTION

Accurate critic value estimation is central to reinforcement learning (RL), as it provides the learning signal for actor policy improvement (Sutton & Barto, 2018; Levine et al., 2020). In online RL, where additional interactions with the environment are available, critic learning benefits not only from bootstrapping-based updates (Mnih et al., 2013; Van Hasselt et al., 2016; Fujimoto et al., 2018) but also from continued data collection (Haarnoja et al., 2018), which progressively enlarges the state-action coverage of the replay buffer. Nevertheless, standard one-step bootstrapping for critic learning remains susceptible to biases induced by compounding function approximation errors. Consequently, multi-step value learning approaches have been studied as a means to balance the low variance of one-step bootstrapping with the reduced bias of Monte Carlo returns. By interpolating between these two extremes, multi-step methods provide more reliable target values for critic learning (Precup et al., 2000; Harutyunyan et al., 2016; Munos et al., 2016; Peng & Williams, 1994; Wang et al., 2017; Rowland et al., 2020; Espeholt et al., 2018; Schulman et al., 2015).

Despite its advantages, multi-step value estimation has remained underexplored in offline RL due to the inherent off-policyness of the data and the difficulty of correcting it in multi-step settings. Specifically, in one-step temporal difference (TD), the action-value target for a target policy π is given by $Q_{\text{target}}(s,a) = r(s,a) + \gamma \mathbb{E}_{a' \sim \pi}[Q(s',a')]$ for a transition $(s, a, r(s,a), s')$. This formulation does not require off-policy correction, even if the action a was drawn from a behavior policy, because a is given for action-value computation. In the multi-step case, however, subsequent rewards depend on actions sequentially drawn from the behavior policy, necessitating off-policy correction to estimate the target value for π. The most common approach is importance sampling (IS), which requires knowledge of the ratio between the target and behavior densities. In offline RL, however, this ratio is typically inaccessible due to the unavailability of the behavior policy density. This challenge hinders the adoption of multi-step estimation in offline settings, where one-step TD remains the predominant approach. Furthermore, offline RL suffers from the value overestimation of out-of-distribution (OOD) actions, resulting from the limited coverage of the dataset and the lack of value correction mechanisms through environmental interactions (Levine et al., 2020; Lange et al., 2012; Fujimoto et al., 2019). This issue further complicates the application of multi-step value estimation to offline settings.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

In this paper, we consider and tackle multi-step value estimation with OOD action handling in offline settings, going beyond the predominant one-step TD learning to exploit the multifaceted benefits of multi-step estimation. To this end, we adopt Retrace(λ) (Munos et al., 2016), a robust online off-policy multi-step action-value estimator, as our backbone and incorporate several key components to enable stable multi-step learning in offline settings. Our key idea to overcome the unavailability of the importance sampling ratio is to introduce a conditional density ratio estimator. This estimator can be learned using a binary cross-entropy loss solely with samples from the behavior and target policies, bypassing the need for explicit density values. The resulting critic learning algorithm, Q-trace, uses this estimator for off-policyness correction and enables a flexible bias-variance trade-off. However, we show that the Retrace(λ) target still bootstraps from potential OOD actions of the target policy in offline RL. Exploiting the ability of this density ratio estimator to identify OOD action in a pointwise manner, we introduce two OOD action handling techniques: pointwise critic penalization required for stable multi-step value accumulation without divergence, and a novel rejection sampling for filtering out OOD candidate actions. We demonstrate that FQT achieves superior performance across diverse offline RL tasks, and that the gain from multi-step critic learning becomes more pronounced when it is combined with our OOD handling techniques.

The contributions of this paper are summarized as follows:

• We propose Q-trace, an offline multi-step action value estimator with the capability of bias-variance trade-off beyond the predominant one-step TD learning for offline RL.
• We propose an effective way based on a conditional density ratio function to compute the importance sampling ratio required for off-policy correction, only with samples from behavior and target policies, without requiring knowledge of the behavior and target probability densities.
• We propose new effective OOD action handling techniques by exploiting the capacity of the same density ratio function to test whether an action is OOD or not for a given state.
• We propose FQT by integrating Q-trace into flow-based policies and density-ratio-based OOD action handling techniques, demonstrating the performance gains of moving beyond predominant one-step critic learning in offline RL.

2 PRELIMINARIES

Reinforcement Learning A Markov Decision Process (MDP) is defined by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, d_0, r, \gamma)$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ is the action space, $P\left(s'|s,a\right) : \mathcal{S}\times\mathcal{A} \mapsto \Delta\left(\mathcal{S}\right)$ is the transition dynamics, d_0 is the initial state distribution, $r(s,a) : \mathcal{S}\times\mathcal{A} \mapsto \mathbb{R}$ is the reward function, and γ is the discount factor (Puterman, 1990). The goal of RL is to find a policy $\pi(a|s) : \mathcal{S} \mapsto \Delta\left(\mathcal{A}\right)$ that maximizes the expected discounted return $J(\pi) = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)|d_0,P\right]$. For a policy π, the action value function is defined as $Q^\pi(s,a) = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)|s_0=s,a_0=a\right]$, which is typically used as a critic for actor, i.e., policy improvement.

A common approach to estimate Q^π is to iteratively apply the Bellman operator to $Q(s,a)$, i.e., $\mathcal{T}^\pi Q(s,a) = r(s,a) + \gamma\mathbb{E}_{s'\sim P(\cdot|s,a),a'\sim\pi(\cdot|s')}\left[Q(s',a')\right]$, which corresponds to one-step bootstrapping. Alternatively, Monte Carlo (MC) returns can be used to learn the value function based on the law of large numbers without bootstrapping. However, each of these two methods has its own limitation. Specifically, one-step bootstrapping suffers from biases due to function approximation and bootstrapping errors. On the other hand, MC-based estimation is unbiased but exhibits high variance. Both excessive bias and high variance can adversely affect policy improvement, necessitating a careful bias-variance trade-off (Sutton & Barto, 2018).

Multi-Step Value Learning Multi-step value learning is an effective approach to address the aforementioned bias-variance trade-off in value estimation (Harutyunyan et al., 2016; Wang et al., 2017; Rowland et al., 2020; Espeholt et al., 2018; Schulman et al., 2015). In particular, several off-policy RL approaches have been proposed to learn value functions from trajectories generated by a behavior policy β different from the target policy π under evaluation.

Uncorrected value estimates: A simple way to off-policy multi-step value estimation is to directly use trajectories generated by the behavior policy β to construct *uncorrected* n-step returns, which is defined by $\mathcal{T}_n^{\pi,\beta}Q(s_t,a_t) = \mathbb{E}_\beta\left[\sum_{h=0}^{n-1}\gamma^h r_{t+h}\right] + \gamma^n\mathbb{E}_\pi\left[Q(s_{t+n},a_{t+n})\right]$. The associated λ-return, $\mathcal{T}_\lambda^{\pi,\beta}Q(s_t,a_t) = (1-\lambda)\sum_{n\geq 1}\lambda^{n-1}\mathcal{T}_n^{\pi,\beta}Q(s_t,a_t)$, can be used as the target for value learning

(Peng & Williams, 1994). Since this target does not account for the discrepancy between the behavior and target policies, the resulting operator converges to the biased action value function $Q^{\lambda\beta+(1-\lambda)\pi}$ (Harutyunyan et al., 2016; Kozuno et al., 2021). Note that in the on-policy case of $\beta = \pi$, the learned value function reduces to desired Q^π, whereas in the off-policy case the learned value function becomes biased as the target policy π deviates from the behavior policy β.

Per-decision importance sampling correction: The most common approach to correct the off-policyness in multi-step value estimation is to use importance sampling. The return $G_t = \sum_{h\geq 0}\gamma^h r_{t+h}$ is a function of the whole trajectory $\tau = (s_t, a_t, r_t, s_{t+1}, \cdots)$, and the importance sampling yields its expectation under π as

$$\mathbb{E}_\pi\left[G_t\right] \overset{(a)}{=} \mathbb{E}_{\tau\sim\beta}\left[\left(\sum_{h\geq 0}\gamma^h r_{t+h}\right)\frac{d^\pi(\tau)}{d^\beta(\tau)}\right] \overset{(b)}{=} \mathbb{E}_\beta\left[\sum_{h\geq 0}\gamma^h\left(\prod_{k=1}^{h}\frac{\pi(a_{t+k}|s_{t+k})}{\beta(a_{t+k}|s_{t+k})}\right)r_{t+h}\right], \quad (1)$$

where $d^\pi(\tau)$ and $d^\beta(\tau)$ are the state-action distributions of π and β, respectively. Step (a) is by simply applying the law of expectations and importance sampling. Per-decision correction converts this expectation into the form of the last term in eq. (1) with Step (b) based on the linearity of expectation and the fact that r_{t+h} does not depend on the trajectory segment after time step h (Precup et al., 2000). This per-decision correction forms the basis of current multi-step off-policy correction methods based on importance sampling (Harutyunyan et al., 2016; Munos et al., 2016; Espeholt et al., 2018). In the case of episodic MC with episode length T, the above per-decision multi-step target can easily be implemented by applying the following recursion backward in time from the end of the episode, i.e., $t = T-1, T-2, \cdots$ with initial $Q_{\text{target}} = 0$:

$$Q_{\text{target}} \;\leftarrow\; r_t + \gamma\frac{\pi(a_{t+1}|s_{t+1})}{\beta(a_{t+1}|s_{t+1})}Q_{\text{target}}. \quad (2)$$

It is straightforward to verify the backward recursion (2) yields the last term in eq. (1). Thus, backward recursion in time is typical in off-policy multi-step value learning.

Offline RL and Flow-Based Policies Offline RL is a paradigm where policy learning is conducted using a static dataset $\mathcal{D} = \{(s, a, r, s')\}$ collected by unknown behavior policies β, without further environmental interactions. The central challenge in offline RL is the value overestimation of out-of-distribution (OOD) actions, i.e., those outside the support of β (Fujimoto et al., 2019; Kumar et al., 2019). This phenomenon stems from the maximization bias in one-step Q-learning combined with the linear extrapolation behavior of MLPs used for Q function approximators outside the training distribution (Xu et al., 2021; Yue et al., 2023; Kim et al., 2025). Furthermore, because offline settings lack a mechanism to correct these erroneous values with real environmental samples, this overestimation persists and degrades performance. To mitigate this issue, many offline RL methods optimize the target policy π under an explicit or implicit constraint relative to the behavior policy (Fujimoto et al., 2019; Peng et al., 2019; Wu et al., 2019; Kumar et al., 2020). Practically, this is achieved via a behavior-regularized actor objective (Fujimoto & Gu, 2021; Wu et al., 2022):

$$\max_\pi \;\left\{\mathbb{E}_{s\sim\mathcal{D},a\sim\pi}\left[Q(s,a)\right] - \alpha\,\mathbb{E}_{s\sim\mathcal{D}}\left[D\left(\pi(\cdot|s),\beta(\cdot|s)\right)\right]\right\}, \quad (3)$$

where D is a divergence measure and $\alpha > 0$ is a regularization coefficient.

Flow-based policies: To model the behavior distribution using an expressive policy class, FQL (Park et al., 2025) trains a conditional flow matching model (Lipman et al., 2022) on the actions in the dataset. Specifically, a velocity vector field v_ψ is learned by minimizing the following objective:

$$\min_\psi \;\mathbb{E}_{(s,a)\sim\mathcal{D},z\sim\mathcal{N}(0,I),u\sim\text{Unif}([0,1])}\left[\|v_\psi(\tilde{a}_u; s, u) - (a - z)\|_2^2\right], \quad (4)$$

where $\tilde{a}_u = (1-u)z + ua$. The learned model induces a behavior-cloning (BC) flow policy $\beta_\psi(\cdot|s)$ for the behavior policy β. Following the behavior-regularized actor principle, the flow-based actor π_θ is then optimized as follows (Park et al., 2025):

$$\max_{\pi_\theta}\left\{\mathbb{E}_{s\sim\mathcal{D},a\sim\pi_\theta}\left[Q_\phi(s,a)\right] - \alpha\,\mathbb{E}_{s\sim\mathcal{D},z\sim\mathcal{N}(0,I)}\left[\|a_\theta(s,z) - a_\psi(s,z)\|_2^2\right]\right\}, \quad (5)$$

where $a_\theta(s,z)$ and $a_\psi(s,z)$ denote the actions generated by π_θ and β_ψ for a given (s,z), respectively.

3 PROPOSED METHOD: FLOW Q-TRACE

3.1 Q-TRACE VIA CONDITIONAL DENSITY RATIO ESTIMATION

Motivation To motivate our development, consider the following TD loss for one-step Q learning for offline critic learning (Fujimoto & Gu, 2021; Wang et al., 2022; Park et al., 2025), where $\mathcal{D}$ is collected by the behavior policy β:

$$L_{TD}(\phi) = \mathbb{E}_{(s,a,s')\sim\mathcal{D}}[(r(s,a) + \gamma\mathbb{E}_{a'\sim\pi}[Q_{\bar{\phi}}(s',a')] - Q_\phi(s,a))^2], \tag{6}$$

where $Q_\phi(s,a)$ is a parameterized Q-function and $Q_{\bar{\phi}}(s,a)$ is a target network. The bootstrapping expectation term $\mathbb{E}_{a'\sim\pi}[Q_{\bar{\phi}}(s',a')]$ is replaced with the sample mean with sample(s) $a' \sim \pi(\cdot|s')$ or with $\max_{a'}[Q_{\bar{\phi}}(s',a')]$ in discrete cases. Note that off-policy correction is not required in eq. (6) since a' directly follows the target policy. However, the impact of a potential OOD action a' used as the bootstrapping index invoking an inaccurate value $Q_{\bar{\phi}}(s',a')$ is critical because its value is added just with one step discount. In the n-step TD case, on the other hand, we have

$$Q_{\text{target}} = \text{Correct}[r(s_0,a_0)+\gamma r(s_1,a_1)+\cdots+\gamma^{n-1}r(s_{n-1},a_{n-1})]+\gamma^n\mathbb{E}_{a_n\sim\pi}[Q_{\bar{\phi}}(s_n,a_n)], \tag{7}$$

where $(s_0,a_0,s_1,\cdots,s_{n-1},a_{n-1},s_n) \sim \beta$ and $\text{Correct}[\cdot]$ means off-policy correction (based on the per-decision IS correction). Since $a_1,\cdots,a_{n-1}$ are drawn from β (even though a_0 is considered given for Q-value computation), off-policy correction is now required. However, the impact of inaccurate value bootstrapping with a potential OOD action a_n is reduced by n-step discount in this case and we have a significant portion of sample reward sum in the target. Thus, multi-step value estimation provides not only reduced biases but also an opportunity to mitigate the impact of inaccurate OOD action values on bootstrapping.

However, there are two key issues in off-policy multi-step value estimation. One is the breakdown of reward accumulation from behind when the importance sampling ratio becomes near zero for some step t, as seen in eq. (2). The other is the per-decision correction term $\prod_{k=1}^{h}\frac{\pi(a_{t+k}|s_{t+k})}{\beta(a_{t+k}|s_{t+k})}$ in eq. (1) can have a large varying range due to the multiplicative nature, possibly yielding large variance in estimation. Retrace(λ) (Munos et al., 2016) effectively solves these two issues by applying the following backward-in-time recursion for target value computation:

$$Q_{\text{target}}(s_t,a_t) = r_t + \gamma\left\{\mathbb{E}_{a'_{t+1}\sim\pi}\left[Q(s_{t+1},a'_{t+1})\right] + c_{t+1}\left(Q_{\text{target}}(s_{t+1},a_{t+1}) - Q(s_{t+1},a_{t+1})\right)\right\}, \tag{8}$$

where $c_{t+k}(s,a) = \lambda\min\left(1,\ \frac{\pi(a_{t+k}|s_{t+k})}{\beta(a_{t+k}|s_{t+k})}\right)$ with $\prod_{k=1}^{0} c_{t+k} = 1$, and $\lambda \in [0,1]$ is a trace-decay parameter that controls the bias-variance trade-off. The key idea is to construct the trace coefficient by clipping the importance sampling below one and to mix bootstrapping and Monte-Carlo reward accumulation. This clipping controls the potentially large variance induced by multiplicative importance weights, and, even when $c_{t+1} = 0$, the backward reward accumulation still continues with the bootstrapping term $\mathbb{E}_{a'_{t+1}\sim\pi}[Q(s_{t+1},a'_{t+1})]$. Furthermore, one can define the following Retrace(λ) operator (Munos et al., 2016), which theoretically represents the Retrace(λ) target without recursion form:

$$\mathcal{R}_\lambda^{\pi,\beta}Q(s_t,a_t) = Q(s_t,a_t) + \mathbb{E}_\beta\left[\sum_{h\geq 0}\gamma^h\left(\prod_{k=1}^{h}c_{t+k}\right)\left(r_{t+h} + \gamma\mathbb{E}_\pi\left[Q(s_{t+h+1},\cdot)\right] - Q(s_{t+h},a_{t+h})\right)\right]. \tag{9}$$

Then, this operator is a γ-contraction and its fixed point is given by Q^π independently of λ (Munos et al., 2016), providing theoretical support of using Retrace(λ). In this way, Retrace(λ) provides a strong basis for off-policy multi-step action value estimation.

Offline Importance Sampling Ratio Retrace(λ) as a multi-step value estimator provides the desired bias-variance trade-off mechanism for better target computation. Furthermore, in the offline case, the impact of a possible OOD action $a'_{t+1} \sim \pi(\cdot|s_{t+1})$ is mitigated by mixing the bootstrapping with the subsequent accumulated rewards, although this may not be as effective as the direct n-step return (7). However, this trade-off is inevitable to prevent the reward accumulation breakdown in the off-policy setting. We adopt Retrace(λ) as the basis for our offline multi-step value estimator design.

The main hurdle in constructing multi-step offline value estimator is the unavailability of the density ratio $\frac{\pi(a_t|s_t)}{\beta(a_t|s_t)}$. This ratio is essential for off-policy correction and its absence has been the primary

reason for the prevalence of one-step TD learning in offline critic acquisition. Note that learning β from a complicated dataset is difficult although there exist some works approximating β with evidence lower bound (ELBO) (Wu et al., 2022; Yeom et al., 2024). Furthermore, the density evaluation required for both target and behavior policies is challenging in expressive policies such as recent diffusion-based or flow-based policies.

To circumvent this difficulty, inspired by Sugiyama et al. (2012); Goodfellow et al. (2020); Ho & Ermon (2016); Chae et al. (2022); Han & Sung (2021); Vuong et al. (2022), we introduce a conditional density ratio that can yield the desired importance sampling ratio as follows:

$$\Gamma(s,a) := \tfrac{\pi(a|s)}{\pi(a|s)+\beta(a|s)}. \tag{10}$$

We parameterize this ratio function with parameter w as Γ_w and learn this function with the following binary cross-entropy (BCE) loss:

$$\min_w \; L(\Gamma_w) = \min_w \left\{-\mathbb{E}_{s\sim\mathcal{D}}\left[\mathbb{E}_{a\sim\pi}\left[\log \Gamma_w(s,a)\right] + \mathbb{E}_{a\sim\beta}\left[\log(1-\Gamma_w(s,a))\right]\right]\right\}. \tag{11}$$

Indeed, we can easily show that the optimal solution of eq. (11) is the desired ratio as stated in the following lemma:

Lemma 1. *The conditional density ratio estimator Γ_w trained by eq. (11) has the following closed-form of optimal solution: $\Gamma^*_w(s,a) = \frac{\pi(a|s)}{\pi(a|s)+\beta(a|s)}$.*

The key advantage of this approach is that we do not need the densities π and β but only require samples from π and β, which can easily be obtained for most policy classes, to learn Γ_w via eq. (11). Then, from the learned ratio function Γ^*_w, we can obtain the desired importance sampling ratio as

$$\frac{\pi(a|s)}{\beta(a|s)} = \frac{\Gamma^*_w(s,a)}{1-\Gamma^*_w(s,a)} \tag{12}$$

and use this to obtain the trace coefficient after clipping, i.e., $c(s,a) = \lambda \min(1, \frac{\Gamma^*_w(s,a)}{1-\Gamma^*_w(s,a)})$. We name this offline multi-step critic learning method **Q-trace**, which computes the Retrace(λ) target (9) with the importance sampling ratio (12) obtained from Γ_w trained by (11).

Now we examine how the Retrace(λ) target (9) behaves in the offline RL setting to gain insights into the offline overestimation issue.

Theorem 1. *For a given state s, let $\mathcal{A}^{\text{OOD}}_{\beta(\cdot|s)}$, $\mathcal{A}^{\text{non-clipped}}_{\beta(\cdot|s)}$, and $\mathcal{A}^{\text{clipped}}_{\beta(\cdot|s)}$ denote the following three disjoint subsets that partition the action space $\mathcal{A}$:*

$$\begin{aligned}
\mathcal{A}^{\text{OOD}}_{\beta(\cdot|s)} &= \{a|a\in\mathcal{A},\ \beta(a|s)=0\}\\
\mathcal{A}^{\text{non-clipped}}_{\beta(\cdot|s)} &= \{a|a\in\mathcal{A},\ \beta(a|s)>0,\ \tfrac{\pi(a|s)}{\beta(a|s)}\le 1\}\\
\mathcal{A}^{\text{clipped}}_{\beta(\cdot|s)} &= \{a|a\in\mathcal{A},\ \beta(a|s)>0,\ \tfrac{\pi(a|s)}{\beta(a|s)}> 1\},
\end{aligned}$$

where the latter two sets correspond to the action sets of which importance sampling ratios are not clipped and clipped by the trace coefficient $c(s,a)=\lambda\min\left(1,\pi(a|s)/\beta(a|s)\right)$, respectively. Then, the Retrace(λ) target (9) represented in a recursion form can be written as

$$\begin{aligned}
\mathcal{R}^{\pi,\beta}_\lambda Q(s_t,a_t) = r_t + \gamma\Big\{ &\int_{a\in\mathcal{A}^{\text{OOD}}_{\beta(\cdot|s_{t+1})}} \pi(a|s_{t+1})Q(s_{t+1},a)\\
&+\int_{a\in\mathcal{A}^{\text{non-clipped}}_{\beta(\cdot|s_{t+1})}} \pi(a|s_{t+1})\left((1-\lambda)\,Q(s_{t+1},a)+\lambda\mathcal{R}^{\pi,\beta}_\lambda Q(s_{t+1},a)\right)\\
&+\int_{a\in\mathcal{A}^{\text{clipped}}_{\beta(\cdot|s_{t+1})}} \pi(a|s_{t+1})\left(\left(1-\lambda\frac{\beta(a|s_{t+1})}{\pi(a|s_{t+1})}\right)Q(s_{t+1},a)+\lambda\frac{\beta(a|s_{t+1})}{\pi(a|s_{t+1})}\mathcal{R}^{\pi,\beta}_\lambda Q(s_{t+1},a)\right)\Big\}.
\end{aligned} \tag{13}$$

Theorem 1 shows that under the Retrace(λ) backward recursion, the actor actions $a\sim\pi(\cdot|s_{t+1})$ within the support of the behavior policy are learned through a convex combination of one-step bootstrapping and the next-step Retrace target accumulating the rewards from behind, as desired. With large λ close to 1, the one-step bootstrapping weakens and the accumulated rewards from behind becomes stronger, mitigating the overestimation issue from bootstrapping with $a_{t+1}\sim\pi(\cdot|s_{t+1})$. On the other hand, the actor actions outside the behavior support are updated solely via one-step bootstrapping, which is vulnerable to overestimated Q value bootstrapping. This motivates the explicit OOD action handling described in the following section.

3.2 PENALIZING OOD ACTIONS

Theorem 1 reveals that, in the offline RL setting, the Retrace(λ) target may still suffer from bootstrapping with overestimated values on potential OOD actions $a_{t+1} \sim \pi(\cdot|s_{t+1})$. To address this issue, one can use the behavior-regularized actor optimization objective (3) in general or (5) in the case of flow-based policies. However, recent studies show that the behavior-regularized actor design (5) alone cannot fully eliminate OOD action sampling (Chae et al., 2026). Therefore, in offline RL, it is desirable to train the critic so that such OOD actions are penalized, thereby discouraging the actor from selecting such actions. Our key idea for such penalization is again to exploit the density ratio function $\Gamma_w^*(s,a)$, as explained in the following lemma.

Lemma 2. *For any (s,a) and $\pi(a|s)$, $\Gamma_w^*(s,a)$ monotonically increases to one as $\beta(a|s) \downarrow 0$, and for an OOD action a with $\beta(a|s) = 0$, we have $\Gamma_w^*(s,a) = \frac{\pi(a|s)}{\pi(a|s)+\beta(a|s)} = \frac{\pi(a|s)}{\pi(a|s)} = 1$.*

Lemma 2 states that we can precisely determine whether an action a is OOD for a given state s or not, based on $\Gamma_w^*(s,a)$. We only need to test if $\Gamma_w^*(s,a) \gtrless 1-\epsilon =: \zeta$ for some small $\epsilon > 0$. Since the learned Γ_w^* is not perfect, we allow a sufficient margin for ϵ. Thus, our ratio function Γ_w^* provides not only the importance sampling ratio required for off-policy correction but also an effective mechanism to distinguish OOD actions for a given state in a pointwise manner!

One way to correct the overestimation problem at OOD actions is to apply penalization to OOD action values, which we call Penalizing OOD Actions (**POA**). To this end, our OOD test based on Γ_w^* can be incorporated directly into the critic learning loss as

$$\min_{Q_\phi} \mathbb{E}_{(s_t,a_t)\sim\mathcal{D}}\Big[\eta\, \mathbb{E}_{a\sim\pi(\cdot|s_t)}\left[1_{\{\Gamma_w^*(s_t,a)\geq\zeta\}} Q_\phi(s_t,a)\right] + \Big(Q_\phi(s_t,a_t) - \mathcal{R}_\lambda^{\pi,\beta} Q(s_t,a_t)\Big)^2\Big], \quad (14)$$

where $\eta\ (>0)$ controls the strength of penalization, and $1_{\{\Gamma_w^*(s,a)\geq\zeta\}}$ is the indicator function, which is 1 if $\Gamma_w^*(s,a) \geq \zeta$, or 0 otherwise. In (14), s_t is sampled from the dataset. For in-distribution action a_t, the Q target is obtained as the Retrace(λ) target. In addition, we sample a from $\pi(\cdot|s_t)$. When a is OOD for given s_t, which can be tested easily by $1_{\{\Gamma_w^*(s_t,a)\geq\zeta\}}$, we conditionally add $Q_\phi(s_t,a)$ with weighting factor η in the loss so that the Q value at OOD a is suppressed. In this way, the bootstrapping part of the Retrace(λ) backward recursion (8) is less affected by potentially overestimated Q values for potential OOD actions from π, and hence the Retrace(λ) backward recursion flows stably without explosion. Hence, the critic objective (14) corresponds to Q-trace equipped with POA. Regarding the critic learning, we have the following theoretical results:

Theorem 2. *Let β and π be the behavior and target actor policies, and Q be the action value function of π. In tabular settings without function approximation, the loss (14) yields the following operator:*

$$\mathcal{R}_{POA}^{\pi,\beta} Q(s,a) = \begin{cases} \mathcal{R}_\lambda^{\pi,\beta} Q(s,a) & \text{if } \Gamma_w^*(s,a) < \zeta \\ \mathcal{R}_\lambda^{\pi,\beta} Q(s,a) - \frac{\eta}{2}\frac{\pi(a|s)}{\beta(a|s)} & \text{if } \Gamma_w^*(s,a) \geq \zeta \end{cases} \quad (15)$$

Definition 1. *The support-constrained policy class Π is defined as $\Pi = \{\pi | \pi(a|s) = 0$ whenever $\beta(a|s) = 0\}$, where β is the behavior policy.*

Theorem 3. *Any policy π trained by a behavior-regularized actor objective (3) and the critic loss (14) is a support-constrained policy, i.e., $\pi \in \Pi$. In other words, π does not sample actions outside the support of behavior policy β.*

Theorem 3 shows that the proposed critic loss (14) combined with the behavior-regularized actor design (5) helps mitigate the overestimation issue on OOD actions, which is the central challenges in offline RL. The proofs of theoretical analyses in this section are in Appendix C.

3.3 DENSITY RATIO BASED REJECTION SAMPLING AND PRACTICAL IMPLEMENTATION

Density-ratio-based rejection sampling We leverage our density estimator at test (inference) time, which we call Density-ratio-based Rejection Sampling (**DRS**), as described in Lines 16-28 of Algorithm 1. Similar to many generative policies, we sample multiple candidate actions but apply rejection sampling based on the density estimator to ensure robust action selection. It is important to note that in general offline RL, increasing the sample size to select the action with maximum learned

Algorithm 1 Flow Q-trace (FQT)

Input: Target policy π_θ, BC flow policy β_ψ, critic network Q_ϕ, target network $Q_{\bar{\phi}}$, conditional density ratio network Γ_w, bias-variance trade-off parameter λ, OOD identification threshold ζ, dataset $\mathcal{D}$

Training:
for each iteration **do**
  Sample $(s_t, a_t, r_t, s_{t+1}, \ldots, r_{t+l-1}, s_{t+l}) \sim \mathcal{D}$ (drawn from β) and $a' \sim \pi(\cdot|s_{t+l})$
  $G = r_{t+l-1} + \gamma Q_{\bar{\phi}}(s_{t+l}, a')$ (pure one-step bootstrap for last transitions)
  for $i = l-1$ to 1 **do** (Backward recursion)
    $\rho_{t+i} \leftarrow \frac{\Gamma_w(s_{t+i}, a_{t+i})}{1-\Gamma_w(s_{t+i}, a_{t+i})}$
    $c_{t+i} \leftarrow \lambda \min(1, \rho_{t+i})$
    Sample $a' \sim \pi(\cdot|s_{t+i})$
    $G \leftarrow r_{t+i-1} + \gamma \left\{ Q_{\bar{\phi}}(s_{t+i}, a') + c_{t+i} \left(G - Q_{\bar{\phi}}(s_{t+i}, a_{t+i}) \right) \right\}$
  end for
  Update Q_ϕ: $\min_\phi \ \mathbb{E}_{(s_t,a_t)\sim\mathcal{D}} \Big[\eta \, \mathbb{E}_{a\sim\pi_\theta(\cdot|s_t)} \left[1_{\{\Gamma_w(s_t,a)\geq\zeta\}} Q_\phi(s_t, a) \right] + \left(G - Q_\phi(s_t, a_t) \right)^2 \Big]$
  Update π_θ and β_ψ with eqs. (5) and (4), respectively.
  Update $Q_{\bar{\phi}}$: $\bar{\phi} \leftarrow \kappa\phi + (1-\kappa)\bar{\phi}$ with some $\kappa \in (0, 1)$
  Update Γ_w: $\min_w \{ -\mathbb{E}_{s\sim\mathcal{D}} [\mathbb{E}_{a\sim\pi} [\log \Gamma_w(s, a)] + \mathbb{E}_{a\sim\mathcal{D}} [\log(1 - \Gamma_w(s, a))]] \}$
end for

Execution:
Sample initial state $s_0 \sim d_0$
for $t = 0, 1, 2, \ldots$ **do**
  Sample candidate actions $\{a_t^{(i)}\}_{i=1}^N \sim \pi_\theta(\cdot|s_t)$ and compute $\Gamma_t^{(i)} = \Gamma_w(s_t, a_t^{(i)})$ for $i = 1, \ldots, N$
  Define $\mathcal{I}_{\text{ID}} = \{i \in \{1, \ldots, N\} \mid \Gamma_t^{(i)} < \zeta\}$
  if $\mathcal{I}_{\text{ID}} \neq \emptyset$ **then**
    Select $i^\star = \arg\max_{i\in\mathcal{I}_{\text{ID}}} Q_\phi(s_t, a_t^{(i)})$
  else
    Select $i^\star = \arg\min_{i\in\{1,\ldots,N\}} \Gamma_t^{(i)}$ (i.e., choose the action closest to the in-distribution region)
  end if
  Set $a_t = a_t^{(i^\star)}$
  Sample $s_{t+1} \sim P(\cdot|s_t, a_t)$
end for

Q-value does not necessarily improve performance. This is because a larger sample size increases the probability of selecting an OOD action, particularly when Q-values for OOD regions are not sufficiently penalized. By incorporating rejection sampling based on the density ratio, we provide an additional safeguard against OOD actions, complementing our Q-trace with OOD penalization.

Practical implementation Note that Q-trace can be integrated as a critic to any offline actor via behavior-regularized actor optimization. We present **Flow Q-trace (FQT)**, our offline actor-critic RL algorithm combining Q-trace with flow-based policies and our OOD action handling techniques (POA and DRS). Its implementation is provided in Algorithm 1. In summary, we adopt the one-step flow policy as the actor and train it with a behavior-regularized actor objective (5) (Park et al., 2025), where the BC flow policy for the behavior-regularized actor is learned via the conditional flow matching objective (4). The critic network is trained with the objective (14), which augments the multi-step critic learning with explicit critic penalization on OOD actions. The density ratio estimator Γ_w required for multi-step off-policy correction and OOD penalization is trained using the loss (11).

4 Related Work

A central challenge in offline RL is to prevent the actor from attending erroneous critic estimates on unsupported actions outside the support of the dataset. To prevent this, offline RL methods commonly adopt behavior-regularized actor-critic learning to mitigate value overestimation on OOD actions by regularizing the actor toward the behavior policy (Fujimoto & Gu, 2021), learning a conservative critic (Kumar et al., 2020), or combining both (Tarasov et al., 2023). Recent works adopt expressive flow-based policies to better capture multi-modal behavior distributions. FQL (Park et al., 2025), which optimizes a one-step flow policy with behavior-regularized Q-value maximization, and FAC (Chae et al., 2026), which further incorporates conservative critic learning through behavior

Table 1: **Offline RL results** on 50 singletasks of the OGBench. For each task category, the results are averaged across its 5 singletasks, over 8 seeds, with ± indicating the standard deviation. We highlight the best and second-best results. Full results are in Table 3.

Task Category	BC	FQL	FAC	QC-FQL	FQT
antmaze-large-navigate	11	89±3	**90**±1	35±4	**91**±1
antmaze-giant-navigate	0	10±5	**19**±6	1±1	**23**±3
humanoidmaze-medium-navigate	2	**54**±6	**67**±4	14±3	**54**±15
humanoidmaze-large-navigate	1	4±1	**6**±4	0±0	**28**±10
antsoccer-arena-navigate	1	**60**±2	**63**±2	22±2	58±3
cube-single-play	5	96±1	**98**±1	**98**±1	**98**±1
cube-double-play	2	30±3	31±4	**43**±6	**49**±4
scene-play	5	68±6	**69**±8	60±3	**74**±7
puzzle-3x3-play	2	93±4	**99**±1	98±1	**99**±1
puzzle-4x4-play	0	22±1	31±6	**35**±2	**47**±3
Average	3	53	**57**	40	**62**

density estimation. However, their critics are still primarily trained with one-step TD learning. Beyond one-step critic learning, QC (Li et al., 2025) introduces Q-chunking, in which the critic evaluates fixed-length temporally extended action sequences. Our method instead enables multi-step critic learning with expressive policies by estimating conditional density ratios for the Retrace(λ) target (Munos et al., 2016), which preserves the target-policy fixed point in off-policy setting and, with the same estimator, identifies OOD actions in a pointwise manner for penalization and rejection.

We provide a more detailed discussion of related work in Appendix B.

5 Experiment

5.1 Experiment Setup

Benchmark We use OGBench task suite (Park et al., 2024) as our benchmark, and consider the reward-based singletask variants provided by OGBench for all tasks. Specifically, we evaluate on 10 environments, consisting of 5 `navigation` environments and 5 `manipulation` environments, each of which provides 5 singletask variants. The benchmark details are provided in Appendix E.1.

Methods For offline RL experiments, we compare our methods against four baselines: behavior cloning (BC), the flow-policy-based one-step RL methods FQL (Park et al., 2025) and FAC (Chae et al., 2026), and the flow-policy-based n-step RL method QC-FQL (Li et al., 2025). To ensure a controlled comparison with baselines, we tune the discount factor over $\gamma \in \{0.99, 0.995\}$ for all flow-policy-based methods. Detailed hyperparameter configurations are provided in Appendix E.3.

Evaluation For offline RL experiments, all methods are trained for 1M gradient steps. Following the evaluation protocol of OGBench (Park et al., 2024), we report the performance in Table 1 and 2. For each task, we report the mean and standard deviation over 8 random seeds. More details are provided in Appendix E.2.

5.2 Offline RL results

Table 1 summarizes the results over 50 OGBench singletasks under the tuned discount-factor settings. FQT achieves the best overall performance and shows strong performance across both navigation and manipulation tasks. Compared with the one-step baselines FQL and FAC, FQT provides clear gains on challenging tasks such as `antmaze-giant`, `humanoidmaze-large`, `cube-double`, and `puzzle-4x4`, while remaining competitive on the remaining task categories. Compared with the fixed-length n-step baseline QC-FQL, FQT shows substantial improvements across the navigation tasks and strong overall performance on manipulation tasks. These results demonstrate the effectiveness of combining corrected multi-step critic learning with density-ratio-based OOD action handling via POA and DRS. We next disentangle the contributions of the discount factor, multi-step target, POA, and DRS through component analysis.

Table 2: **Component analysis** on 50 singletasks of OGBench. For each task category, the results are averaged over 8 seeds, with ± indicating the standard deviation. For the four task categories where our tuned discount factor γ differs from that of the original FQL (Park et al., 2025), we report results under both values of γ. The last two columns report the average scores under the original FQL's and tuned discount-factor settings. Full results are provided in Tables 3 and 4.

Method	antmaze large		antmaze giant	humanoidmaze medium	humanoidmaze large	antsoccer arena	cube single	cube double	scene		puzzle 3x3		puzzle 4x4		Average	
γ	0.99	0.995	0.995	0.995	0.995	0.995	0.99	0.99	0.99	0.995	0.99	0.995	0.99	0.995	FQL's	Tuned
FQL	78±2	89±3	10±5	54±6	4±1	**60**±2	96±1	30±3	58±3	68±6	29±2	93±4	19±1	22±1	44	53
+POA	71±1	89±3	15±2	56±13	12±4	59±2	**98**±1	30±5	57±1	70±7	92±3	98±2	20±1	22±2	51	55
+DRS	**82**±3	88±4	6±4	53±5	3±2	55±2	95±2	41±4	**59**±1	68±6	29±3	98±1	36±3	41±2	46	55
+POA&DRS	76±4	89±3	14±2	**58**±13	**19**±10	55±2	96±1	41±3	**59**±1	69±8	**99**±1	**99**±1	**37**±2	**44**±1	**55**	**58**
QT-FQL	81±1	**91**±1	9±7	**57**±8	5±2	**60**±4	96±1	30±3	58±2	**74**±4	30±2	93±2	19±2	21±2	45	54
+POA	77±1	**91**±1	**20**±3	54±13	13±4	**60**±3	**99**±1	30±2	57±1	73±5	98±1	**99**±1	20±1	23±2	53	56
+DRS	**82**±2	90±1	9±6	**57**±9	5±2	58±5	97±1	**49**±3	**59**±1	**74**±4	32±3	98±2	36±2	42±2	49	**58**
+POA&DRS	81±2	**91**±1	**23**±3	54±15	**28**±10	58±3	**98**±1	**49**±4	**59**±1	**74**±7	**100**±0	**99**±1	**38**±3	**47**±3	**59**	**62**

5.3 Component Analysis

Table 2 shows the contributions of the multi-step critic target, POA, and DRS using FQL and QT-FQL as the one-step and multi-step counterparts, respectively. Specifically, the multi-step component corresponds to the Retrace(λ) target (9), where the importance-sampling ratio is estimated using Γ_w; POA corresponds to the critic penalization term $\mathbb{E}_{a\sim\pi_\theta}\left[\mathbf{1}_{\{\Gamma_w(s,a)\geq\zeta\}}Q_\phi(s,a)\right]$ in the objective (14); and DRS corresponds to our action-selection procedure in Lines 16-28 of Algorithm 1. QT-FQL differs from FQL by replacing the one-step TD target with the multi-step target, while FQT corresponds to QT-FQL equipped with both POA and DRS. We additionally apply POA and DRS components to FQL to assess their effects. We report performance under both the original FQL's discount-factor setting and our tuned discount-factor setting; they differ only on `antmaze-large`, `scene`, `puzzle-3x3`, and `puzzle-4x4`, where the tuned setting uses $\gamma = 0.995$ instead of $\gamma = 0.99$. The tuned discount-factor setting yields better overall performance, while showing similar performance patterns across both settings.

Effects of POA and DRS POA and DRS provide distinct gains across task categories for both FQL and QT-FQL. POA is particularly beneficial on challenging navigation tasks such as `antmaze-giant` and `humanoidmaze-large`, but also provides substantial improvement on `puzzle-3x3`, especially with $\gamma = 0.99$, from 29 to 92 for FQL and from 30 to 98 for QT-FQL. DRS provides particularly clear gains on `cube-double` and `puzzle-4x4`. Using POA and DRS together yields stronger overall performance for both FQL and QT-FQL.

Multi-step critic learning benefits from OOD action handling Comparing FQL and QT-FQL, replacing the one-step TD target with the Retrace(λ) target alone yields a modest improvement in overall performance. This is consistent with Theorem 1: without OOD action handling, the Retrace(λ) target still bootstraps from potentially overestimated values of OOD actions sampled from the learned policy. In contrast, when POA and DRS are applied, FQT outperforms FQL+POA&DRS, improving the overall score from 55 to 59 and from 58 to 62 under the FQL's and tuned settings, respectively.

Additional analyses of Γ_w and FQT are provided in Appendices D.2 and D.3.

6 Conclusion

We have proposed Flow Q-trace, an offline RL algorithm that enables corrected multi-step critic learning with expressive flow-based policies and effectively addresses value overestimation on OOD actions. The key idea is to learn a conditional density ratio estimator, which provides the actor-to-behavior density ratio for off-policy correction and can identify OOD actions in a pointwise manner. By exploiting this property, FQT incorporates OOD-action-aware critic penalization and density-ratio-based rejection sampling, thereby suppressing overestimated OOD values and promoting the selection of in-distribution actions. Across diverse offline RL tasks on OGBench, FQT achieves strong overall performance, and the results demonstrate the superiority of combining corrected multi-step critic learning with our pointwise OOD-action handling in offline RL.

AI USE STATEMENT

In this work, we have not used generative AI tools for any tasks with required disclosure, including theoretical development, mathematical derivations and proofs, method design, experimental design, or result interpretation. Additionally, we used generative AI tools only for tasks with recommended disclosure, specifically to polish and refine the writing for clarity, readability, and grammatical correctness. We have reviewed all AI-assisted work, and every suggested edit was manually checked by the authors against the original content before being incorporated into the manuscript. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work does not involve human subjects or any personal information. All experiments use only publicly available datasets from OGBench (Park et al., 2024), whose original work is properly cited, and are conducted in simulated environments.

REPRODUCIBILITY STATEMENT

The implementation of our method is described in Algorithm 1. Detailed experimental settings and hyperparameters are provided in Appendices E.2 and E.3, respectively, and full task-level results over 8 random seeds are reported in Appendix D.1.

REFERENCES

Bhavya Kumar Agrawalla, Michal Nauman, Khush Agrawal, and Aviral Kumar. floq: Training critics via flow-matching for scaling compute in value-based RL. In *The Fourteenth International Conference on Learning Representations*, 2026.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/jax-ml/jax`.

Jongseong Chae, Seungyul Han, Whiyoung Jung, Myungsik Cho, Sungho Choi, and Youngchul Sung. Robust imitation learning against variations in environment dynamics. In *International Conference on Machine Learning*, pp. 2828–2852. PMLR, 2022.

Jongseong Chae, Jongeui Park, Yongjae Shin, Gyeongmin Kim, Seungyul Han, and Youngchul Sung. Flow actor-critic for offline reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026.

Huayu Chen, Cheng Lu, Chengyang Ying, Hang Su, and Jun Zhu. Offline reinforcement learning via high-fidelity generative behavior modeling. *arXiv preprint arXiv:2209.14548*, 2022.

Huayu Chen, Cheng Lu, Zhengyi Wang, Hang Su, and Jun Zhu. Score regularized policy optimization through diffusion behavior. In *The Twelfth International Conference on Learning Representations*, 2024.

Qian Chen, Junqiao Zhao, Hongtu Zhou, Hang Yu, Yanping Zhao, Chen Ye, and Guang Chen. Acsac: Adaptive chunk size actor-critic with causal transformer q-network. *arXiv preprint arXiv:2605.11009*, 2026.

Shutong Ding, Ke Hu, Zhenhao Zhang, Kan Ren, Weinan Zhang, Jingyi Yu, Jingya Wang, and Ye Shi. Diffusion-based reinforcement learning via q-weighted variational policy optimization. *Advances in Neural Information Processing Systems*, 37:53945–53968, 2024.

Zihan Ding and Chi Jin. Consistency models as a rich and efficient policy class for reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2024.

Perry Dong, Chongyi Zheng, Chelsea Finn, Dorsa Sadigh, and Benjamin Eysenbach. Value flows. In *The Fourteenth International Conference on Learning Representations*, 2026.

JaeHyeok Doo, Byeongguk Jeon, Seonghyeon Ye, Kimin Lee, and Minjoon Seo. Q-flow: Stable and expressive reinforcement learning with flow-based policy. In *Forty-third International Conference on Machine Learning*, 2026.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*, pp. 1407–1416. PMLR, 2018.

Nicolas Espinosa-Dice, Yiyi Zhang, Yiding Chen, Bradley Guo, Owen Oertell, Gokul Swamy, Kianté Brantley, and Wen Sun. Scaling offline RL via efficient and expressive shortcut models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

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

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

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. Pmlr, 2018.

Seungyul Han and Youngchul Sung. Diversity actor-critic: Sample-aware entropy regularization for sample-efficient exploration. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 4041–4051. PMLR, 2021.

Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies. *arXiv preprint arXiv:2304.10573*, 2023.

Anna Harutyunyan, Marc G Bellemare, Tom Stepleton, and Rémi Munos. Q () with off-policy corrections. In *International Conference on Algorithmic Learning Theory*, pp. 305–320. Springer, 2016.

Longxiang He, Li Shen, Junbo Tan, and Xueqian Wang. Aligniql: Policy alignment in implicit q-learning through constrained optimization. *arXiv preprint arXiv:2405.18187*, 2024.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.

Anas Houssaini, Mohamad H Danesh, Amin Abyaneh, Scott Fujimoto, Hsiu-Chin Lin, and David Meger. Drift q-learning. *arXiv preprint arXiv:2606.00350*, 2026.

Bingyi Kang, Xiao Ma, Chao Du, Tianyu Pang, and Shuicheng Yan. Efficient diffusion policies for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36: 67195–67212, 2023.

Byeongchan Kim and Min hwan Oh. Peng's q(λ) for conservative value estimation in offline reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026.

Changyeon Kim, Haeone Lee, Younggyo Seo, Kimin Lee, and Yuke Zhu. DEAS: DEtached value learning with action sequence for scalable offline RL. In *The Fourteenth International Conference on Learning Representations*, 2026.

Geon-Hyeong Kim, Seokin Seo, Jongmin Lee, Wonseok Jeon, HyeongJoo Hwang, Hongseok Yang, and Kee-Eung Kim. DemoDICE: Offline imitation learning with supplementary imperfect demonstrations. In *International Conference on Learning Representations*, 2022.

Jeonghye Kim, Yongjae Shin, Whiyoung Jung, Sunghoon Hong, Deunsol Yoon, Youngchul Sung, Kanghoon Lee, and Woohyung Lim. Penalizing infeasible actions and reward scaling in reinforcement learning with offline data. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, 2025.

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

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

Tadashi Kozuno, Yunhao Tang, Mark Rowland, Rémi Munos, Steven Kapturowski, Will Dabney, Michal Valko, and David Abel. Revisiting peng's q (λ) for modern reinforcement learning. In *International Conference on Machine Learning*, pp. 5794–5804. PMLR, 2021.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in neural information processing systems*, 32, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33:1179–1191, 2020.

Sascha Lange, Thomas Gabel, and Martin Riedmiller. Batch reinforcement learning. In *Reinforcement learning: State-of-the-art*, pp. 45–73. Springer, 2012.

Jongmin Lee, Wonseok Jeon, Byungjun Lee, Joelle Pineau, and Kee-Eung Kim. Optidice: Offline policy optimization via stationary distribution correction estimation. In *International Conference on Machine Learning*, pp. 6120–6130. PMLR, 2021.

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.

Qiyang Li, Zhiyuan Zhou, and Sergey Levine. Reinforcement learning with action chunking. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Qiyang Li, Seohong Park, and Sergey Levine. Decoupled q-chunking. In *The Fourteenth International Conference on Learning Representations*, 2026.

Yuanchang Liang, Xiaobo Wang, Kai Wang, Shuo Wang, Xiaojiang Peng, Haoyu Chen, David Kim Huat Chua, and Prahlad Vadakkepat. Adaptive action chunking at inference-time for vision-language-action models. In *Conference on Computer Vision and Pattern Recognition 2026*, 2026.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Cheng Lu, Huayu Chen, Jianfei Chen, Hang Su, Chongxuan Li, and Jun Zhu. Contrastive energy prediction for exact energy-guided diffusion sampling in offline reinforcement learning. In *International Conference on Machine Learning*, pp. 22825–22855. PMLR, 2023.

Jiafei Lyu, Xiaoteng Ma, Xiu Li, and Zongqing Lu. Mildly conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:1711–1724, 2022.

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: 40587–40609, 2023.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Zhancun Mu. Deflow: Decoupling manifold modeling and value maximization for offline policy extraction. *arXiv preprint arXiv:2601.10471*, 2026.

Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. Safe and efficient off-policy reinforcement learning. *Advances in neural information processing systems*, 29, 2016.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

Thanh Xuan Nguyen and Chang D. Yoo. One-step flow q-learning: Addressing the diffusion policy bottleneck in offline reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026.

Alexander Nikulin, Vladislav Kurenkov, Denis Tarasov, and Sergey Kolesnikov. Anti-exploration by random network distillation. In *International conference on machine learning*, pp. 26228–26244. PMLR, 2023.

Kwanyoung Park, Seohong Park, Youngwoon Lee, and Sergey Levine. Scalable offline model-based RL with action chunks. In *The Fourteenth International Conference on Learning Representations*, 2026a.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. *arXiv preprint arXiv:2410.20092*, 2024.

Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning. In *Forty-second International Conference on Machine Learning*, 2025.

Seohong Park, Kevin Frans, Deepinder Mann, Benjamin Eysenbach, Aviral Kumar, and Sergey Levine. Horizon reduction makes RL scalable. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026b.

Seohong Park, Aditya Oberai, Pranav Atreya, and Sergey Levine. Transitive RL: Value learning via divide and conquer. In *The Fourteenth International Conference on Learning Representations*, 2026c.

Jing Peng and Ronald J Williams. Incremental multi-step q-learning. In *Machine Learning Proceedings 1994*, pp. 226–232. Elsevier, 1994.

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.

Doina Precup, Richard S Sutton, and Satinder Singh. Eligibility traces for off-policy policy evaluation. 2000.

Martin L Puterman. Markov decision processes. *Handbooks in operations research and management science*, 2:331–434, 1990.

Mark Rowland, Will Dabney, and Rémi Munos. Adaptive trade-offs in off-policy learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 34–44. PMLR, 2020.

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

Yongjae Shin, Jongseong Chae, Seongmin Kim, Jongeui Park, and Youngchul Sung. Adaptive action chunking via multi-chunk q value estimation. *arXiv preprint arXiv:2605.10044*, 2026.

Gwanwoo Song, Kwanyoung Park, and Youngwoon Lee. Chunk-guided q-learning. In *Forty-third International Conference on Machine Learning*, 2026.

Masashi Sugiyama, Taiji Suzuki, and Takafumi Kanamori. *Density ratio estimation in machine learning*. Cambridge University Press, 2012.

Richard S Sutton and Andrew G Barto. Reinforcement learning: an introduction, 2nd edn. adaptive computation and machine learning, 2018.

Denis Tarasov and Robert K Katzschmann. Rebrac-v2: The return of the king. *arXiv preprint arXiv:2608.01205*, 2026.

Denis Tarasov, Vladislav Kurenkov, Alexander Nikulin, and Sergey Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36:11592–11620, 2023.

Franki NGUIMATSIA TIOFACK, Théotime Le Hellard, Fabian Schramm, Nicolas Perrin-Gilbert, and Justin Carpentier. Guided flow policy: Learning from high-value actions in offline reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Quan Vuong, Aviral Kumar, Sergey Levine, and Yevgen Chebotar. Dasco: Dual-generator adversarial support constrained offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:38937–38949, 2022.

Andrew Wagenmaker, Yunchu Zhang, Mitsuhiko Nakamoto, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space reinforcement learning. In *9th Annual Conference on Robot Learning*, 2025.

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.

Ziyu Wang, Victor Bapst, Nicolas Heess, Volodymyr Mnih, Remi Munos, Koray Kavukcuoglu, and Nando de Freitas. Sample efficient actor-critic with experience replay. In *International Conference on Learning Representations*, 2017.

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.

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

Keyulu Xu, Mozhi Zhang, Jingling Li, Simon S. Du, Ken-ichi Kawarabayashi, and Stephanie Jegelka. How neural networks extrapolate: From feedforward to graph neural networks. In *International Conference on Learning Representations*, 2021.

Junghyuk Yeom, Yonghyeon Jo, Jeongmo Kim, Sanghyeon Lee, and Seungyul Han. Exclusively penalized q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 37:113405–113435, 2024.

Sheng Yue, Xingyuan Hua, Ju Ren, Sen Lin, Junshan Zhang, and Yaoxue Zhang. OLLIE: Imitation learning from offline pretraining to online finetuning. In *Forty-first International Conference on Machine Learning*, 2024.

Yihao Yue, Robert Lu, Bingyi Kang, Shiji Song, and Gao Huang. Understanding, predicting and better resolving q-value divergence in offline rl. In *Advances in Neural Information Processing Systems*, volume 36, pp. 55502–55523, 2023.

Guojian Zhan, Letian Tao, Pengcheng Wang, Yixiao Wang, Yuxin Chen, Yiheng Li, Hongyang Li, Masayoshi Tomizuka, and Shengbo Eben Li. Mean flow policy with instantaneous velocity constraint for one-step action generation. In *The Fourteenth International Conference on Learning Representations*, 2026.

Shiyuan Zhang, Weitong Zhang, and Quanquan Gu. Energy-weighted flow matching for offline reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

A LIMITATIONS

Our method introduces additional components beyond standard one-step RL frameworks, including conditional density ratio estimation and multi-step critic learning, which may increase computational cost and requires additional hyperparameters. In addition, the proposed method does not incorporate an explicit exploration mechanism, so its applicability to online RL settings is not established in this work. That said, considering the success of prior multi-step RL methods in online RL, we view extending the proposed framework to the online settings as a promising direction for future work.

B EXTENDED RELATED WORK

Behavior-regularized actor-critic for offline RL A central challenge in offline RL is to prevent the actor from attending erroneous critic estimates on unsupported actions outside the support of the dataset. Prior works have addressed this issue mainly through three directions: explicit behavior-regularized actor optimization, weighted regression, and conservative critic learning. The first two can be viewed as actor-regularization approaches, which constrain the target policy to remain close to the underlying behavioral distribution through explicit support constraints under a distance constraint such as TD3+BC (Fujimoto & Gu, 2021), SPOT (Wu et al., 2022), and BEAR (Kumar et al., 2019), or value-weighted regression only on samples of the dataset such as IQL (Kostrikov et al., 2021), AWR (Peng et al., 2019), AWAC (Nair et al., 2020), and OptiDICE (Lee et al., 2021). Conservative critic learning approaches suppress the values of unsupported actions, whose probability density under the behavior policy is low or absent in the offline dataset, during critic learning such as CQL (Kumar et al., 2020), MCQ (Lyu et al., 2022), SVR (Mao et al., 2023), and SAC-RND (Nikulin et al., 2023). Several methods combine actor regularization with critic conservatism such as BRAC (Wu et al., 2019), ReBRAC (Tarasov et al., 2023), and PARS (Kim et al., 2025). Despite these differences, most existing offline RL methods still train the critic using one-step temporal-difference targets or via SARSA-like in-sample learning.

Offline RL with expressive policies As offline datasets become more diverse and often exhibit multi-modal action distributions, recent work has explored expressive policy classes based on flow, diffusion, and other generative models. In flow model based offline RL, FQL (Park et al., 2025) optimizes a one-step flow actor via the behavior regularization with explicit Q-maximization, and FAC (Chae et al., 2026) further incorporates conservative critic learning based on density estimates obtained from a flow-matching model. GFP (TIOFACK et al., 2026) introduces value-aware behavior cloning, where a multi-step flow policy models high-value dataset actions and regularizes a distilled one-step flow actor. MVP (Zhan et al., 2026) and OFQL (Nguyen & Yoo, 2026) instead learn mean velocity fields to directly generate actions in a single step. Whereas, DeFlow (Mu, 2026) preserves a multi-step flow policy and decouples value maximization into an action-refinement module, while Q-Flow (Doo et al., 2026) propagates terminal Q-values to intermediate flow states to enable stable optimization of multi-step flow policies. QIPO-OT (Zhang et al., 2025) adopts value-weighted regression for policy improvement. A separate line of research investigates the use of flow models for critic representation. Value Flows (Dong et al., 2026) and floq (Agrawalla et al., 2026) use flow models to parameterize distributional critics. Similarly, in diffusion-based offline RL, IDQL (Hansen-Estruch et al., 2023), SfBC (Chen et al., 2022) and AlignIQL (He et al., 2024) are based on value-based importance sampling to represent actor, while QGPO (Lu et al., 2023), QVPO (Ding et al., 2024) and EDP (Kang et al., 2023) are based on value-weighted regression as actor update design. Diffusion-QL (Wang et al., 2022), SRPO (Chen et al., 2024) and CAC (Ding & Jin, 2024) use explicit Q-maximization objectives via backpropagation through denoising steps. Another direction explores alternative expressive generative models for efficient action generation. SORL (Espinosa-Dice et al., 2026) uses self-consistent shortcut models to enable flexible one-step to multi-step action generation, while DriftQL (Houssaini et al., 2026) adopts a one-step drifting model that combines attraction-repulsion-based behavior regularization with value-guided policy improvement. Complementary to these advances in expressive policy modeling, our method enables practical multi-step critic learning with expressive policies, including flow-based policies.

Multi-step RL and off-policy correction Multi-step critic learning has been widely studied in online RL to balance the low variance of short-horizon bootstrapping against the reduced bias of longer-horizon returns. For the on-policy setting, GAE (Schulman et al., 2015) uses λ-returns to

reduce the variance of policy-gradient estimates. For the off-policy setting, Peng's $Q(\lambda)$ (Peng & Williams, 1994) directly uses an uncorrected λ-return from trajectories generated by the behavior policy, leading to a mixture-policy fixed point biased toward the behavior policy when the behavior and target policies differ (Kozuno et al., 2021). To instead recover the target-policy fixed point, off-policy multi-step methods introduce corrections for the mismatch between the target and behavior policies. Harutyunyan's $Q(\lambda)$ (Harutyunyan et al., 2016) augments the λ-return with off-policy correction terms to account for the discrepancy between the target and behavior policies. Retrace (Munos et al., 2016), V-trace (Espeholt et al., 2018), α-trace (Rowland et al., 2020), and ACER (Wang et al., 2017) incorporate per-decision importance-sampling-based corrections into multi-step value estimation (Precup et al., 2000). More recently, several works have explored approaches beyond one-step RL by leveraging multi-step objectives or reducing the effective decision horizon in offline and goal-conditioned RL. SHARSA (Park et al., 2026b) reduces the effective decision horizon through hierarchical policies, while Transitive RL (Park et al., 2026c) introduces a divide-and-conquer value learning framework to mitigate the accumulation of bootstrapping bias over long horizons. Another line of work leverages action chunking, where critics evaluate temporally extended action sequences to accelerate value propagation and reduce the effective backup horizon. QC (Li et al., 2025) formulates Q-learning over action chunks, while DQC (Li et al., 2026) decouples the policy and critic chunk lengths. DEAS (Kim et al., 2026) extends action-chunked value learning with distributional value estimation, and MAC (Park et al., 2026a) uses temporally extended actions to mitigate compounding errors in learned dynamics over long horizons. CGQ (Song et al., 2026) instead retains a single-step critic and regularizes it toward an action-chunked critic. Building on these fixed-length action-chunking approaches, subsequent works have explored adaptive chunking strategies that vary the temporal decision horizon according to the current state. ACH (Shin et al., 2026) and ACSAC (Chen et al., 2026) jointly estimate Q-values for multiple candidate chunk lengths using causal Transformer critics and dynamically select the chunk length based on the current state. Relatedly, in vision-language-action models, AAC (Liang et al., 2026) adapts the executed chunk length at inference time using action entropy as a measure of prediction uncertainty. A distinct line of work directly incorporates multi-step bootstrapped targets. CPQL (Kim & hwan Oh, 2026) incorporates Peng's $Q(\lambda)$ target into conservative value estimation based on CQL (Kumar et al., 2020). Its mixture-policy fixed point, $Q^{\lambda\beta+(1-\lambda)\pi}$, induces implicit behavior regularization, which can mitigate overly pessimistic value estimation, improve robustness to the conservatism coefficient, and provide a tighter sub-optimality guarantee than its single-step conservative counterpart. CPQL and our method therefore take distinct approaches to offline multi-step critic learning: CPQL leverages the behavior regularization induced by its mixture-policy fixed point, whereas our method adopts the Retrace(λ) target (Munos et al., 2016) and estimates the target-policy-to-behavior-policy density ratio to preserve the target-policy fixed point Q^π, even with expressive policies.

Action-selection strategies for expressive policies Expressive generative policies naturally enable multiple candidate actions to be sampled at test (inference) time, motivating value-guided action selection. A simple strategy is best-of-N (BFN), which samples multiple actions from the policy and executes the candidate with the highest estimated Q-value, as in QC (Li et al., 2025). ReBRAC-v2 (Tarasov & Katzschmann, 2026) further refines sampled actions using critic gradients before value-based action selection. Another line of work considers sampling-based policy extraction, where candidate actions from expressive behavior models are reweighted according to learned values, as in SfBC (Chen et al., 2022), IDQL (Hansen-Estruch et al., 2023), and AlignIQL (He et al., 2024). More recently, DSRL (Wagenmaker et al., 2025) learns a policy over the latent space of a pretrained expressive policy to steer generated actions toward higher-return behaviors. Our density-ratio-based rejection sampling (DRS) builds on BFN by first filtering candidate actions identified as OOD using the target-policy-to-behavior-policy density ratio and then selecting the highest Q-value action among the remaining candidates.

Conditional density ratio estimator in RL Density ratios can be estimated without explicit density modeling by training a binary classifier to distinguish samples from two distributions (Sugiyama et al., 2012; Goodfellow et al., 2020). This principle has been used in reinforcement and imitation learning. GAIL (Ho & Ermon, 2016) uses a discriminator to distinguish policy and expert state-action distributions, and related offline imitation learning methods adopt similar discriminator- or density-ratio-based formulations (Kim et al., 2022; Yue et al., 2024). In reinforcement learning, DAC (Han & Sung, 2021) uses density-ratio estimation between the current policy and replay-buffer action

distributions for sample-aware entropy regularization, while DASCO (Vuong et al., 2022) leverages discriminator scores to measure deviation from the offline dataset distribution and enforce support constraint. Our method estimates the target-policy-to-behavior-policy density ratio from action samples and uses it for both multi-step off-policy correction and pointwise OOD-action identification.

C Theoretical Analysis

C.1 Conditional Density Ratio Estimator

Lemma 1. *The conditional density ratio estimator Γ_w trained by eq. (11) has the following closed-form of optimal solution: $\Gamma_w^*(s,a) = \frac{\pi(a|s)}{\pi(a|s)+\beta(a|s)}$.*

Proof. Let $d^{\mathcal{D}}(s)$ be normalized state distribution over $\mathcal{D}$, then eq. (11) can be rewritten as:

$$\mathcal{L}(\Gamma_w) = -\int_{s\in\mathcal{S}} d^{\mathcal{D}}(s) \int_{a\in\mathcal{A}} \pi(a|s)\log\Gamma_w(s,a) + \beta(a|s)\log\left(1-\Gamma_w(s,a)\right)$$

We set the derivative of eq. (11) to zero:

$$\frac{\partial\mathcal{L}(\Gamma_w)}{\partial\Gamma_w} = -\frac{\pi(a|s)}{\Gamma_w(s,a)} + \frac{\beta(a|s)}{1-\Gamma_w(s,a)} = 0,$$

leading to the optimal solution $\Gamma_w^*(s,a)$, which is given by

$$\Gamma_w^*(s,a) = \frac{\pi(a|s)}{\pi(a|s)+\beta(a|s)}$$

Using this solution, we can have the following equation:

$$\frac{\Gamma_w^*(s,a)}{1-\Gamma_w^*(s,a)} = \frac{\pi(a|s)}{\beta(a|s)},$$

which leads to what we want to prove. □

Lemma 2. *For any (s,a) and $\pi(a|s)$, $\Gamma_w^*(s,a)$ monotonically increases to one as $\beta(a|s)\downarrow 0$, and for an OOD action a with $\beta(a|s)=0$, we have $\Gamma_w^*(s,a) = \frac{\pi(a|s)}{\pi(a|s)+\beta(a|s)} = \frac{\pi(a|s)}{\pi(a|s)} = 1$.*

Proof. For any fixed π, i.e., $\pi(a|s)$ is constant for a given state-action pair (s,a), since the probability densities $\pi(a|s)$ and $\beta(a|s)$ are non-negative, the following equation holds: $\lim_{\beta(a|s)\to 0}\Gamma_w^*(s,a) = \lim_{\beta(a|s)\to 0}\frac{\pi(a|s)}{\pi(a|s)+\beta(a|s)} = 1$. □

C.2 The Recursion of Retrace(λ)

Theorem 1. *For a given state s, let $\mathcal{A}^{\textit{OOD}}_{\beta(\cdot|s)}$, $\mathcal{A}^{\textit{non-clipped}}_{\beta(\cdot|s)}$, and $\mathcal{A}^{\textit{clipped}}_{\beta(\cdot|s)}$ denote the following three disjoint subsets that partition the action space $\mathcal{A}$:*

$$\begin{aligned}
\mathcal{A}^{\textit{OOD}}_{\beta(\cdot|s)} &= \{a|a\in\mathcal{A},\ \beta(a|s)=0\}\\
\mathcal{A}^{\textit{non-clipped}}_{\beta(\cdot|s)} &= \{a|a\in\mathcal{A},\ \beta(a|s)>0,\ \tfrac{\pi(a|s)}{\beta(a|s)}\le 1\}\\
\mathcal{A}^{\textit{clipped}}_{\beta(\cdot|s)} &= \{a|a\in\mathcal{A},\ \beta(a|s)>0,\ \tfrac{\pi(a|s)}{\beta(a|s)}> 1\},
\end{aligned}$$

where the latter two sets correspond to the action sets of which importance sampling ratios are not clipped and clipped by the trace coefficient $c(s,a) = \lambda\min\left(1,\pi(a|s)/\beta(a|s)\right)$, respectively. Then, the Retrace(λ) target (9) represented in a recursion form can be written as

$$\begin{aligned}
\mathcal{R}_\lambda^{\pi,\beta}Q(s_t,a_t) = r_t + \gamma\Big\{ &\int_{a\in\mathcal{A}^{\textit{OOD}}_{\beta(\cdot|s_{t+1})}} \pi(a|s_{t+1})Q(s_{t+1},a)\\
&+\int_{a\in\mathcal{A}^{\textit{non-clipped}}_{\beta(\cdot|s_{t+1})}} \pi(a|s_{t+1})\Big((1-\lambda)\,Q(s_{t+1},a) + \lambda\mathcal{R}_\lambda^{\pi,\beta}Q(s_{t+1},a)\Big) \qquad (13)\\
&+\int_{a\in\mathcal{A}^{\textit{clipped}}_{\beta(\cdot|s_{t+1})}} \pi(a|s_{t+1})\left(\left(1-\lambda\frac{\beta(a|s_{t+1})}{\pi(a|s_{t+1})}\right)Q(s_{t+1},a) + \lambda\frac{\beta(a|s_{t+1})}{\pi(a|s_{t+1})}\mathcal{R}_\lambda^{\pi,\beta}Q(s_{t+1},a)\right)\Big\}.
\end{aligned}$$

Proof. For a given state s, the three sets are disjoint and satisfy:

$$\mathcal{A} = \mathcal{A}^{\text{clipped}}_{\beta(\cdot|s)} \cup \mathcal{A}^{\text{non-clipped}}_{\beta(\cdot|s)} \cup \mathcal{A}^{\text{OOD}}_{\beta(\cdot|s)}$$
$$\text{supp}(\beta(\cdot|s)) = \mathcal{A}^{\text{clipped}}_{\beta(\cdot|s)} \cup \mathcal{A}^{\text{non-clipped}}_{\beta(\cdot|s)},$$

where $\text{supp}(\beta(\cdot|s))$ is the support of behavior policy β. We note that in the Retrace(λ) recursion (8), actions in $\mathcal{A}^{\text{clipped}}_{\beta(\cdot|s)}$ provide $c_{t+1} = \lambda$, and an action a in $\mathcal{A}^{\text{non-clipped}}_{\beta(\cdot|s)}$ provides $c_{t+1} = \lambda\pi(a|s)/\beta(a|s)$.

For notational simplicity, in this proof, we write $s_t, a_t, r_t, s_{t+1}, a_{t+1}$ as s, a, r, s', a', respectively. The recursion (8) can be rewritten by

$$\begin{aligned}
\mathcal{R}^{\pi,\beta}_\lambda Q(s,a) &= r + \gamma\left\{\mathbb{E}_{b\sim\pi(\cdot|s')}\left[Q(s',b)\right] + \mathbb{E}_{a'\sim\beta(\cdot|s')}\left[c_{t+1}\left(\mathcal{R}^{\pi,\beta}_\lambda Q(s',a') - Q(s',a')\right)\right]\right\} \\
&= r + \gamma\Big\{\int_{b\in\mathcal{A}} \pi(b|s')Q(s',b) + \int_{b\in\mathcal{A}^{\text{clipped}}_{\beta(\cdot|s')}} \beta(b|s')\lambda\left(\mathcal{R}^{\pi,\beta}_\lambda Q(s',b) - Q(s',b)\right) \\
&\qquad + \int_{b\in\mathcal{A}^{\text{non-clipped}}_{\beta(\cdot|s')}} \beta(b|s')\lambda\frac{\pi(b|s')}{\beta(b|s')}\left(\mathcal{R}^{\pi,\beta}_\lambda Q(s',b) - Q(s',b)\right)\Big\} \\
&= r + \gamma\Big\{\int_{b\in\mathcal{A}} \pi(b|s')Q(s',b) - \lambda\int_{b\in\mathcal{A}^{\text{clipped}}_{\beta(\cdot|s')}} \beta(b|s')Q(s',b) \\
&\quad - \lambda\int_{b\in\mathcal{A}^{\text{non-clipped}}_{\beta(\cdot|s')}} \beta(b|s')\frac{\pi(b|s')}{\beta(b|s')}Q(s',b) \\
&\quad + \lambda\int_{b\in\mathcal{A}^{\text{clipped}}_{\beta(\cdot|s')}} \beta(b|s')\mathcal{R}^{\pi,\beta}_\lambda Q(s',b) + \lambda\int_{b\in\mathcal{A}^{\text{non-clipped}}_{\beta(\cdot|s')}} \beta(b|s')\frac{\pi(b|s')}{\beta(b|s')}\mathcal{R}^{\pi,\beta}_\lambda Q(s',b)\Big\} \\
&= r + \gamma\Big\{\int_{b\in\mathcal{A}} \pi(b|s')Q(s',b) - \lambda\int_{b\in\mathcal{A}^{\text{clipped}}_{\beta(\cdot|s')}} \pi(b|s')\frac{\beta(b|s')}{\pi(b|s')}Q(s',b) \\
&\quad - \lambda\int_{b\in\mathcal{A}^{\text{non-clipped}}_{\beta(\cdot|s')}} \pi(b|s')Q(s',b) \\
&\quad + \lambda\int_{b\in\mathcal{A}^{\text{clipped}}_{\beta(\cdot|s')}} \pi(b|s')\frac{\beta(b|s')}{\pi(b|s')}\mathcal{R}^{\pi,\beta}_\lambda Q(s',b) + \lambda\int_{b\in\mathcal{A}^{\text{non-clipped}}_{\beta(\cdot|s')}} \pi(b|s')\mathcal{R}^{\pi,\beta}_\lambda Q(s',b)\Big\} \\
&= r + \gamma\Big\{\int_{b\in\mathcal{A}^{\text{OOD}}_{\beta(\cdot|s')}} \pi(b|s')Q(s',b) + \int_{b\in\mathcal{A}^{\text{clipped}}_{\beta(\cdot|s')}} \pi(b|s')\left(1 - \lambda\frac{\beta(b|s')}{\pi(b|s')}\right)Q(s',b) \\
&\quad + \int_{b\in\mathcal{A}^{\text{non-clipped}}_{\beta(\cdot|s')}} (1-\lambda)\,\pi(b|s')Q(s',b) \\
&\quad + \lambda\int_{b\in\mathcal{A}^{\text{clipped}}_{\beta(\cdot|s')}} \pi(b|s')\frac{\beta(b|s')}{\pi(b|s')}\mathcal{R}^{\pi,\beta}_\lambda Q(s',b) + \lambda\int_{b\in\mathcal{A}^{\text{non-clipped}}_{\beta(\cdot|s')}} \pi(b|s')\mathcal{R}^{\pi,\beta}_\lambda Q(s',b)\Big\} \\
&= r + \gamma\Big\{\int_{b\in\mathcal{A}^{\text{OOD}}_{\beta(\cdot|s')}} \pi(b|s')Q(s',b) \\
&\qquad + \int_{b\in\mathcal{A}^{\text{non-clipped}}_{\beta(\cdot|s')}} \pi(b|s')\left((1-\lambda)\,Q(s',b) + \lambda\mathcal{R}^{\pi,\beta}_\lambda Q(s',b)\right) \\
&\qquad + \int_{b\in\mathcal{A}^{\text{clipped}}_{\beta(\cdot|s')}} \pi(b|s')\left(\left(1 - \lambda\frac{\beta(b|s')}{\pi(b|s')}\right)Q(s',b) + \lambda\frac{\beta(b|s')}{\pi(b|s')}\mathcal{R}^{\pi,\beta}_\lambda Q(s',b)\right)\Big\}
\end{aligned}$$

□

C.3 PENALIZING OUT-OF-DISTRIBUTION ACTIONS OBJECTIVE

Theorem 2. *Let β and π be the behavior and target actor policies, and Q be the action value function of π. In tabular settings without function approximation, the loss (14) yields the following operator:*

$$\mathcal{R}^{\pi,\beta}_{POA}Q(s,a) = \begin{cases} \mathcal{R}^{\pi,\beta}_{\lambda}Q(s,a) & \text{if } \Gamma^*_w(s,a) < \zeta \\ \mathcal{R}^{\pi,\beta}_{\lambda}Q(s,a) - \frac{\eta}{2}\frac{\pi(a|s)}{\beta(a|s)} & \text{if } \Gamma^*_w(s,a) \geq \zeta \end{cases} \tag{15}$$

Proof. In the tabular setting, we can replace $a_t \sim \mathcal{D}$ with $a_t \sim \beta$. Then, the objective (14) can be rewritten as

$$\begin{aligned} &\mathbb{E}_{s_t\sim\mathcal{D}}\left[\eta\,\mathbb{E}_{a\sim\pi(\cdot|s_t)}\left[1_{\Gamma^*_w(s_t,a)\geq\zeta}\,Q(s_t,a)\right] + \mathbb{E}_{a_t\sim\beta}\left[\left(Q(s_t,a_t) - \mathcal{R}^{\pi,\beta}_{\lambda}Q(s_t,a_t)\right)^2\right]\right] \\ &= \mathbb{E}_{s\sim\mathcal{D}}\left[\mathbb{E}_{a\sim\beta}\left[\eta\frac{\pi(a|s)}{\beta(a|s)}1_{\Gamma^*_w(s,a)\geq\zeta}\,Q(s,a) + \left(Q(s,a) - \mathcal{R}^{\pi,\beta}_{\lambda}Q(s,a)\right)^2\right]\right] \\ &= \mathbb{E}_{s\sim\mathcal{D},a\sim\beta}\left[\eta\,\frac{1_{\Gamma^*_w(s,a)\geq\zeta}\,\pi(a|s)}{\beta(a|s)}\,Q(s,a) + \left(Q(s,a) - \mathcal{R}^{\pi,\beta}_{\lambda}Q(s,a)\right)^2\right] \end{aligned} \tag{16}$$

We set the derivative of (16) with respect to Q to zero

$$\eta\,\frac{1_{\Gamma^*_w(s,a)\geq\zeta}\,\pi(a|s)}{\beta(a|s)} + 2\left(Q(s,a) - \mathcal{R}^{\pi,\beta}_{\lambda}Q(s,a)\right) = 0,$$

leading to the optimal solution $Q^*(s,a)$, which is given by

$$\mathcal{R}^{\pi,\beta}_{\text{POA}}Q(s,a) = Q^*(s,a) = \mathcal{R}^{\pi,\beta}_{\lambda}Q(s,a) - \frac{\eta}{2}\frac{1_{\Gamma^*_w(s,a)\geq\zeta}\,\pi(a|s)}{\beta(a|s)}$$

Now, we consider two cases:

i) When $\Gamma^*_w(s,a) < \zeta$, i.e., $1_{\Gamma^*_w(s,a)\geq\zeta} = 0$, $\mathcal{R}^{\pi,\beta}_{\text{POA}}Q(s,a) = Q^*(s,a) = \mathcal{R}^{\pi,\beta}_{\lambda}Q(s,a)$.

ii) When $\Gamma^*_w(s,a) \geq \zeta$, $\mathcal{R}^{\pi,\beta}_{\text{POA}}Q(s,a) = \mathcal{R}^{\pi,\beta}_{\lambda}Q(s,a) - \frac{\eta\pi(a|s)}{2\beta(a|s)}$.

□

Definition 1. *The support-constrained policy class Π is defined as $\Pi = \{\pi|\pi(a|s) = 0$ whenever $\beta(a|s) = 0\}$, where β is the behavior policy.*

Theorem 3. *Any policy π trained by a behavior-regularized actor objective (3) and the critic loss (14) is a support-constrained policy, i.e., $\pi \in \Pi$. In other words, π does not sample actions outside the support of behavior policy β.*

Proof. From Theorem 2, for any policy action a outside the support of β, i.e., $\beta(a|s) = 0$ and $\Gamma^*_w(s,a) = 1 \geq \zeta$,

$$\mathcal{R}^{\pi,\beta}_{\text{POA}}Q(s,a) = \mathcal{R}^{\pi,\beta}_{\lambda}Q(s,a) - \frac{\eta\pi(a|s)}{2\beta(a|s)} = -\infty.$$

When the actor π is trained by the behavior-regularized actor objective (3):

$$\begin{aligned} &\mathbb{E}_{s\sim\mathcal{D},a\sim\pi}\left[Q(s,a)\right] - \alpha\,\mathbb{E}_{s\sim\mathcal{D}}\left[D\left(\pi(\cdot|s),\beta(\cdot|s)\right)\right] \\ &= \mathbb{E}_{s\sim\mathcal{D}}\left[\int_{a\notin\text{supp}(\beta)}\pi(a|s)\cdot(-\infty) + \int_{a\in\text{supp}(\beta)}\pi(a|s)Q(s,a) - \alpha\,D\left(\pi(\cdot|s),\beta(\cdot|s)\right)\right], \end{aligned}$$

where $\text{supp}(\beta) = \{a|\beta(a|s) > 0\}$ is the support set of behavior policy. For given state s, the regularization term $-D\left(\pi(\cdot|s),\beta(\cdot|s)\right) < \infty$ is upper bounded. Thus, any policy π maximizing the loss (3) assigns non-zero probability mass only to the in-support actions. In other words, π is a support-constrained policy. □

D ADDITIONAL RESULTS

Table 3: **Full task-level offline RL and component-analysis results on OGBench under the tuned discount-factor settings.** The corresponding hyperparameters are provided in Table 6. For the FQL and QT-FQL variants, we report results under the tuned discount-factor settings used in Tables 1 and 2. For each singletask, results are averaged over 8 seeds, with ± indicating the standard deviation. (∗) indicates the default task in each task category. We highlight the best and second-best results.

	Baselines			FQL Variants				QT-FQL Variants			
Task	**BC**	**FAC**	**QC-FQL**	**FQL**	**+POA**	**+DRS**	**+POA&DRS**	**QT-FQL**	**+POA**	**+DRS**	**FQT**
antmaze-large-navigate-singletask-task1-v0 (∗)	0	**93**±2	44±6	**93**±2	92±1	91±1	91±3	**93**±4	**93**±3	91±3	90±4
antmaze-large-navigate-singletask-task2-v0	6	86±4	11±6	76±19	80±9	77±20	80±9	84±4	85±3	**88**±3	**89**±3
antmaze-large-navigate-singletask-task3-v0	29	94±3	61±5	93±2	93±4	94±1	**96**±2	**95**±2	**95**±3	**95**±1	**95**±3
antmaze-large-navigate-singletask-task4-v0	8	82±7	13±7	89±2	**90**±2	86±2	86±3	89±2	**90**±3	87±4	86±3
antmaze-large-navigate-singletask-task5-v0	10	**93**±2	48±16	92±2	91±3	90±4	92±1	92±2	**94**±2	91±3	**93**±2
antmaze-giant-navigate-singletask-task1-v0 (∗)	0	2±2	0±0	10±8	3±3	9±9	3±2	**13**±9	5±3	**19**±13	6±5
antmaze-giant-navigate-singletask-task2-v0	0	**31**±11	0±0	**14**±9	6±3	1±1	3±3	**14**±14	12±5	6±6	11±5
antmaze-giant-navigate-singletask-task3-v0	0	0±0	0±0	0±0	0±0	0±0	**2**±2	0±0	1±1	0±0	**2**±2
antmaze-giant-navigate-singletask-task4-v0	0	27±10	0±0	3±6	21±9	2±5	26±12	12±20	**30**±12	13±25	**40**±14
antmaze-giant-navigate-singletask-task5-v0	1	35±21	3±2	25±27	44±9	18±21	35±7	7±17	**51**±7	8±20	**54**±9
humanoidmaze-medium-navigate-singletask-task1-v0 (∗)	1	**59**±17	0±1	26±17	38±24	28±18	40±30	17±14	47±25	21±17	**52**±27
humanoidmaze-medium-navigate-singletask-task2-v0	1	89±5	6±1	**96**±3	76±24	**97**±2	74±26	86±21	70±32	88±21	70±33
humanoidmaze-medium-navigate-singletask-task3-v0	6	74±21	34±13	46±33	33±31	36±32	26±33	**79**±22	43±33	**75**±25	36±34
humanoidmaze-medium-navigate-singletask-task4-v0	0	17±16	0±0	14±15	**41**±14	14±17	**58**±25	5±6	19±19	2±3	21±25
humanoidmaze-medium-navigate-singletask-task5-v0	2	97±3	29±6	88±23	91±15	91±18	91±19	**98**±1	89±16	**99**±1	89±17
humanoidmaze-large-navigate-singletask-task1-v0 (∗)	0	6±7	0±0	3±2	17±8	3±3	**28**±13	3±3	15±7	5±3	**32**±21
humanoidmaze-large-navigate-singletask-task2-v0	0	0±0	0±0	0±0	0±0	0±0	0±0	0±0	0±0	0±0	0±0
humanoidmaze-large-navigate-singletask-task3-v0	1	11±7	0±0	14±5	19±9	12±9	**30**±21	20±7	26±11	18±9	**52**±25
humanoidmaze-large-navigate-singletask-task4-v0	1	6±6	0±0	1±1	8±7	0±0	10±18	1±2	**14**±8	0±0	**33**±23
humanoidmaze-large-navigate-singletask-task5-v0	0	8±11	0±0	1±2	14±12	1±3	**27**±24	2±2	13±10	3±5	**24**±21
antsoccer-arena-navigate-singletask-task1-v0	2	78±3	46±6	**80**±2	**80**±4	77±2	74±4	77±4	77±5	75±6	73±3
antsoccer-arena-navigate-singletask-task2-v0	2	**93**±1	20±3	91±1	**92**±2	85±4	86±4	89±2	88±2	84±4	85±5
antsoccer-arena-navigate-singletask-task3-v0	0	**54**±3	23±3	53±5	49±4	53±4	48±11	52±6	52±4	50±5	**56**±3
antsoccer-arena-navigate-singletask-task4-v0 (∗)	1	**44**±8	14±3	**43**±6	42±8	35±5	39±7	**43**±6	**43**±4	38±6	**43**±6
antsoccer-arena-navigate-singletask-task5-v0	0	**45**±8	7±2	35±9	34±6	26±10	29±7	**42**±9	39±9	40±10	36±10
cube-single-play-singletask-task1-v0	10	**99**±1	**98**±1	97±2	**98**±2	95±2	96±2	**98**±1	**98**±1	**98**±2	**98**±2
cube-single-play-singletask-task2-v0 (∗)	3	**99**±1	**100**±0	98±1	**99**±1	95±2	96±3	97±2	**99**±2	97±1	97±2
cube-single-play-singletask-task3-v0	9	99±1	**100**±0	99±1	99±1	97±1	98±2	99±1	**100**±0	99±1	99±1
cube-single-play-singletask-task4-v0	2	97±2	**98**±1	94±3	**98**±1	95±4	96±2	95±3	**98**±1	96±2	97±1
cube-single-play-singletask-task5-v0	3	95±2	95±2	93±2	95±2	91±3	93±2	92±3	**98**±1	95±2	**97**±2
cube-double-play-singletask-task1-v0	8	51±10	71±4	64±5	63±7	63±5	55±9	67±5	**72**±3	**72**±7	71±7
cube-double-play-singletask-task2-v0 (∗)	0	34±8	50±12	37±6	37±8	50±7	50±6	41±8	39±6	**62**±5	**64**±6
cube-double-play-singletask-task3-v0	0	26±7	43±13	23±6	23±6	45±8	51±7	28±8	22±4	**55**±6	**54**±8
cube-double-play-singletask-task4-v0	0	2±1	**13**±3	6±2	5±2	9±3	8±4	6±3	5±3	**12**±4	**12**±5
cube-double-play-singletask-task5-v0	0	42±4	38±11	18±10	21±15	37±15	40±18	9±5	10±4	**47**±10	**43**±10
scene-play-singletask-task1-v0	19	100±0	100±0	100±0	100±0	100±0	100±0	100±0	100±0	100±0	100±0
scene-play-singletask-task2-v0 (∗)	1	**99**±1	**100**±0	93±5	96±3	96±3	97±2	95±4	96±2	98±3	98±2
scene-play-singletask-task3-v0	1	95±5	85±6	95±4	96±2	95±5	94±3	96±1	**97**±1	**97**±1	96±3
scene-play-singletask-task4-v0	2	49±41	14±13	53±31	57±31	49±34	53±39	**77**±18	71±26	**75**±20	74±30
scene-play-singletask-task5-v0	0	**1**±1	**1**±2	0±0	**2**±1	0±0	0±0	**1**±0	**1**±2	0±0	0±0
puzzle-3x3-play-singletask-task1-v0	5	**100**±0	**100**±0	**100**±0	**100**±0	**100**±0	**100**±0	98±2	99±1	99±1	**100**±0
puzzle-3x3-play-singletask-task2-v0	1	98±2	**100**±0	89±12	**100**±0	99±1	**100**±0	91±3	99±1	99±1	**100**±0
puzzle-3x3-play-singletask-task3-v0	1	**98**±1	96±1	85±4	93±9	96±2	96±7	85±5	**98**±2	96±4	**98**±2
puzzle-3x3-play-singletask-task4-v0 (∗)	1	**100**±0	95±2	94±5	**100**±0	97±4	99±2	96±3	**100**±0	98±2	**100**±0
puzzle-3x3-play-singletask-task5-v0	1	**100**±0	97±2	98±2	**99**±1	**99**±1	**99**±2	98±1	**99**±1	97±3	**99**±1
puzzle-4x4-play-singletask-task1-v0	1	66±9	59±8	41±4	46±7	**80**±5	**84**±4	39±4	46±5	72±6	**80**±6
puzzle-4x4-play-singletask-task2-v0	0	6±6	**21**±3	20±4	14±2	9±5	**21**±5	16±3	18±3	18±2	20±4
puzzle-4x4-play-singletask-task3-v0	0	53±19	55±2	26±3	26±6	**86**±3	**78**±6	23±4	25±4	77±4	76±6
puzzle-4x4-play-singletask-task4-v0 (∗)	0	**30**±10	23±3	12±2	13±4	23±6	25±5	15±3	15±3	27±4	**28**±7
puzzle-4x4-play-singletask-task5-v0	0	2±2	15±2	11±2	12±3	6±3	15±2	11±4	13±2	**16**±4	**30**±8

Table 4: **Additional task-level component-analysis results under the original FQL discount factor** $\gamma = 0.99$. We report results for `antmaze-large`, `scene`, `puzzle-3x3`, and `puzzle-4x4` categories, for which the original FQL's discount-factor setting differs from the tuned setting in Table 3. The corresponding hyperparameters are provided in Table 7. For each singletask, results are averaged over 8 seeds, with ± indicating the standard deviation. $(*)$ indicates the default task in each task category. We highlight the best and second-best results.

	FQL Variants				QT-FQL Variants			
Task	**FQL**	**+POA**	**+DRS**	**+POA&DRS**	**QT-FQL**	**+POA**	**+DRS**	**FQT**
antmaze-large-navigate-singletask-task1-v0 $(*)$	**76**±12	58±11	71±17	52±16	**79**±6	72±2	71±11	69±6
antmaze-large-navigate-singletask-task2-v0	60±5	49±8	**80**±4	74±5	66±4	60±7	**79**±5	78±5
antmaze-large-navigate-singletask-task3-v0	93±2	92±2	**96**±2	94±2	94±3	92±2	**95**±3	94±2
antmaze-large-navigate-singletask-task4-v0	**80**±5	**80**±4	76±8	78±5	79±2	**80**±4	79±5	**80**±3
antmaze-large-navigate-singletask-task5-v0	82±4	76±3	85±4	79±4	85±2	81±4	**87**±3	**86**±4
scene-play-singletask-task1-v0	100±0	100±0	100±0	100±0	100±0	100±0	100±0	100±0
scene-play-singletask-task2-v0 $(*)$	81±10	81±8	95±3	**97**±3	84±7	82±6	**97**±2	96±1
scene-play-singletask-task3-v0	96±2	97±2	96±3	97±1	97±2	**98**±1	**98**±2	**98**±2
scene-play-singletask-task4-v0	**13**±7	**7**±4	2±2	1±1	**7**±6	6±4	1±1	2±3
scene-play-singletask-task5-v0	0±0	0±0	0±0	0±0	0±0	0±0	0±0	0±0
puzzle-3x3-play-singletask-task1-v0	91±3	99±1	99±1	**100**±0	90±5	99±1	96±2	**100**±0
puzzle-3x3-play-singletask-task2-v0	14±4	97±2	4±2	99±1	17±3	**100**±0	14±3	**100**±0
puzzle-3x3-play-singletask-task3-v0	9±3	84±8	4±3	**98**±2	12±2	97±1	9±4	**100**±0
puzzle-3x3-play-singletask-task4-v0 $(*)$	12±5	88±6	22±10	**99**±1	13±2	96±3	21±7	**100**±0
puzzle-3x3-play-singletask-task5-v0	19±6	90±2	16±12	**99**±1	20±5	97±2	22±11	**100**±0
puzzle-4x4-play-singletask-task1-v0	38±3	38±3	**67**±8	**68**±9	34±6	40±2	66±6	65±7
puzzle-4x4-play-singletask-task2-v0	**15**±3	**15**±3	7±3	6±4	**15**±3	**15**±3	14±3	13±6
puzzle-4x4-play-singletask-task3-v0	21±3	22±3	**74**±5	**80**±7	21±4	21±4	68±7	71±5
puzzle-4x4-play-singletask-task4-v0 $(*)$	11±3	13±3	**28**±10	27±7	12±2	12±3	26±5	**29**±5
puzzle-4x4-play-singletask-task5-v0	**12**±3	10±3	4±3	5±2	11±2	10±2	10±3	**12**±3

D.1 Learning Curves

We provide learning curves for offline RL methods reported in Tables 1, 2, 3, and 4: (i) FQT against the baselines FQL, FAC and QC-FQL, (ii) the FQL variants, and (iii) the QT-FQL variants. Figure 1 summarizes each task category by averaging its 5 singletasks, and Figures 2, 3, and 4 report every singletask. In Figures 3 and 4, solid curves use the tuned discount-factor settings (in Table 3) and dashed curves the original FQL's discount-factor setting, $\gamma = 0.99$ (in Table 4); the two settings coincide except on `antmaze-large`, `scene`, `puzzle-3x3` and `puzzle-4x4`, so dashed curves appear only in those four task categories. Every curve is the mean over 8 seeds and the shaded region indicates ± one standard deviation.

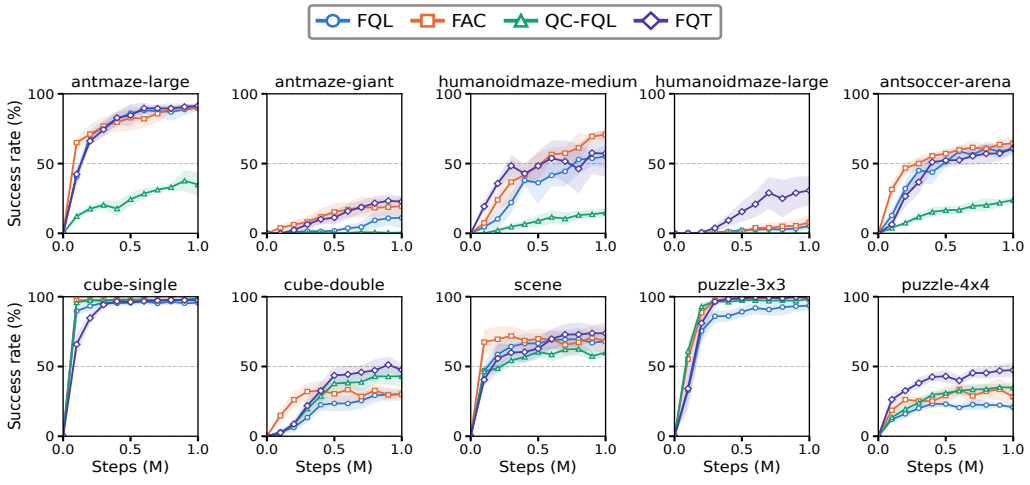


Figure 1: **Task-category learning curves** of FQT and the baselines under the tuned discount-factor settings, averaged over the 5 singletasks of each category within a seed and then over 8 seeds (cf. Table 1).

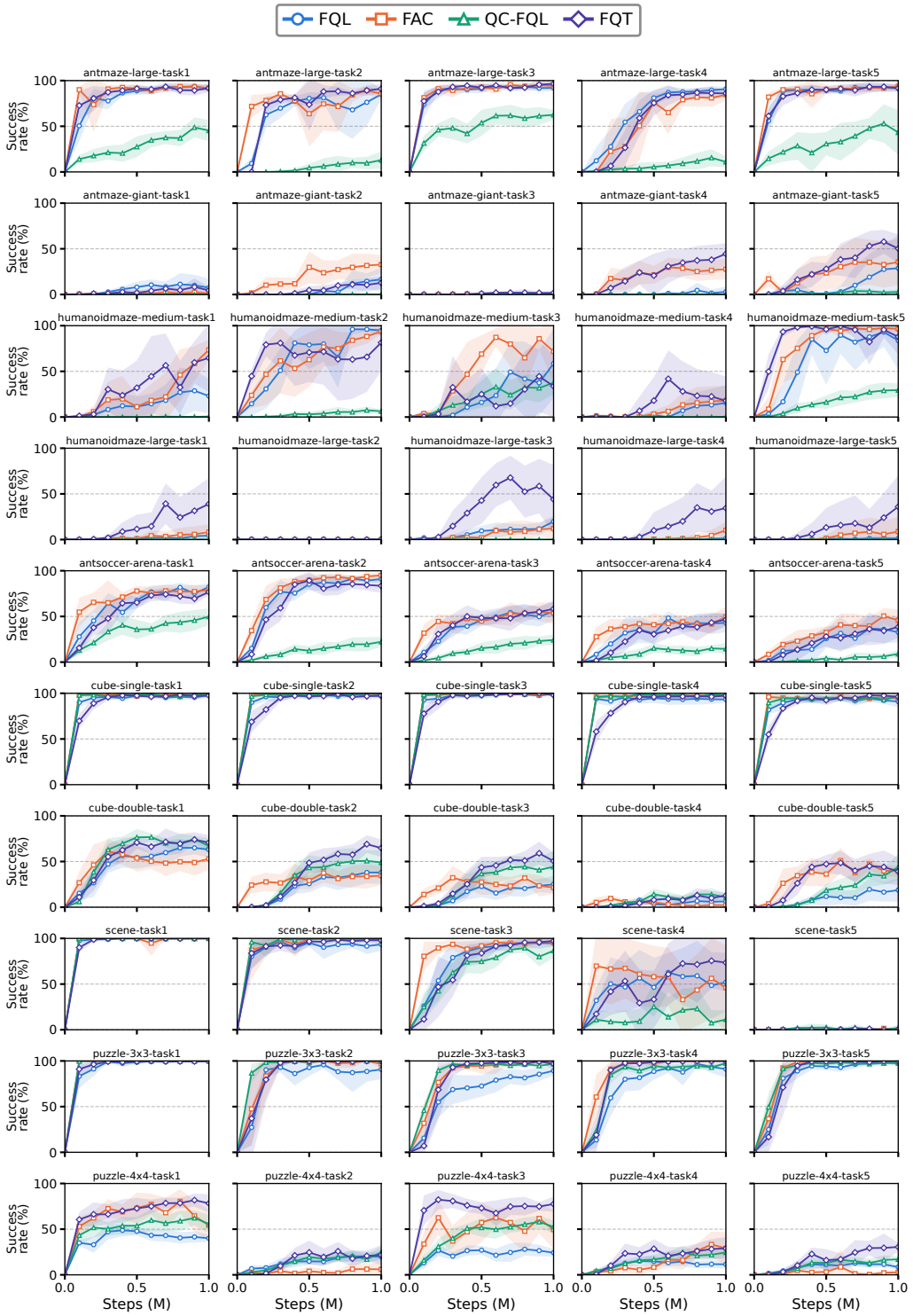


Figure 2: **Task-level learning curves of FQT and the baselines** under the tuned discount-factor settings, corresponding to the *FQT* and *baselines* columns of Table 3.

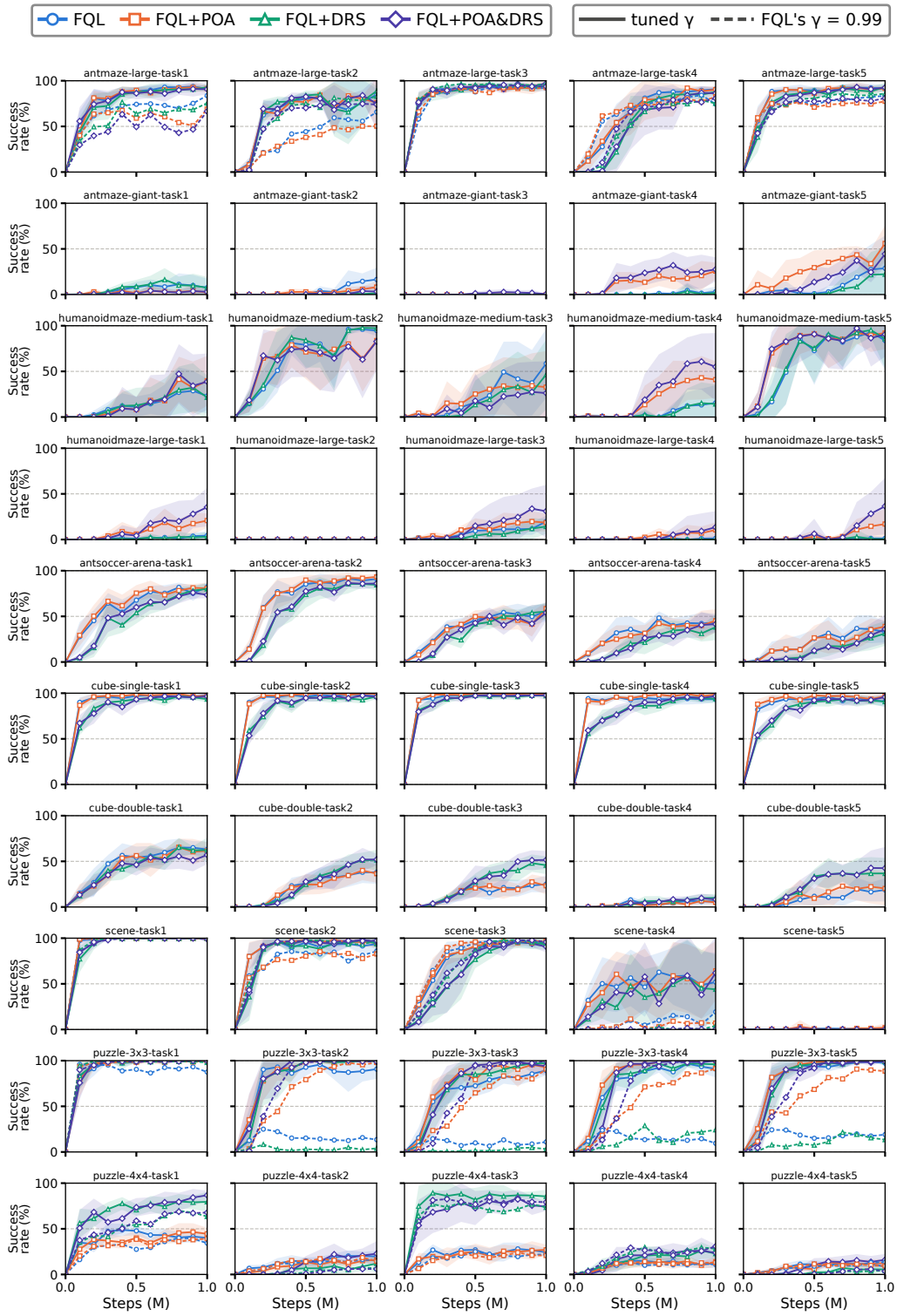


Figure 3: **Task-level learning curves of the FQL variants.** Solid curves use the tuned discount-factor setting (*FQL Variants* columns of Table 3) and dashed curves the original FQL's discount-factor setting (*FQL Variants* columns of Table 4), which differs only on `antmaze-large`, `scene`, `puzzle-3x3` and `puzzle-4x4`.

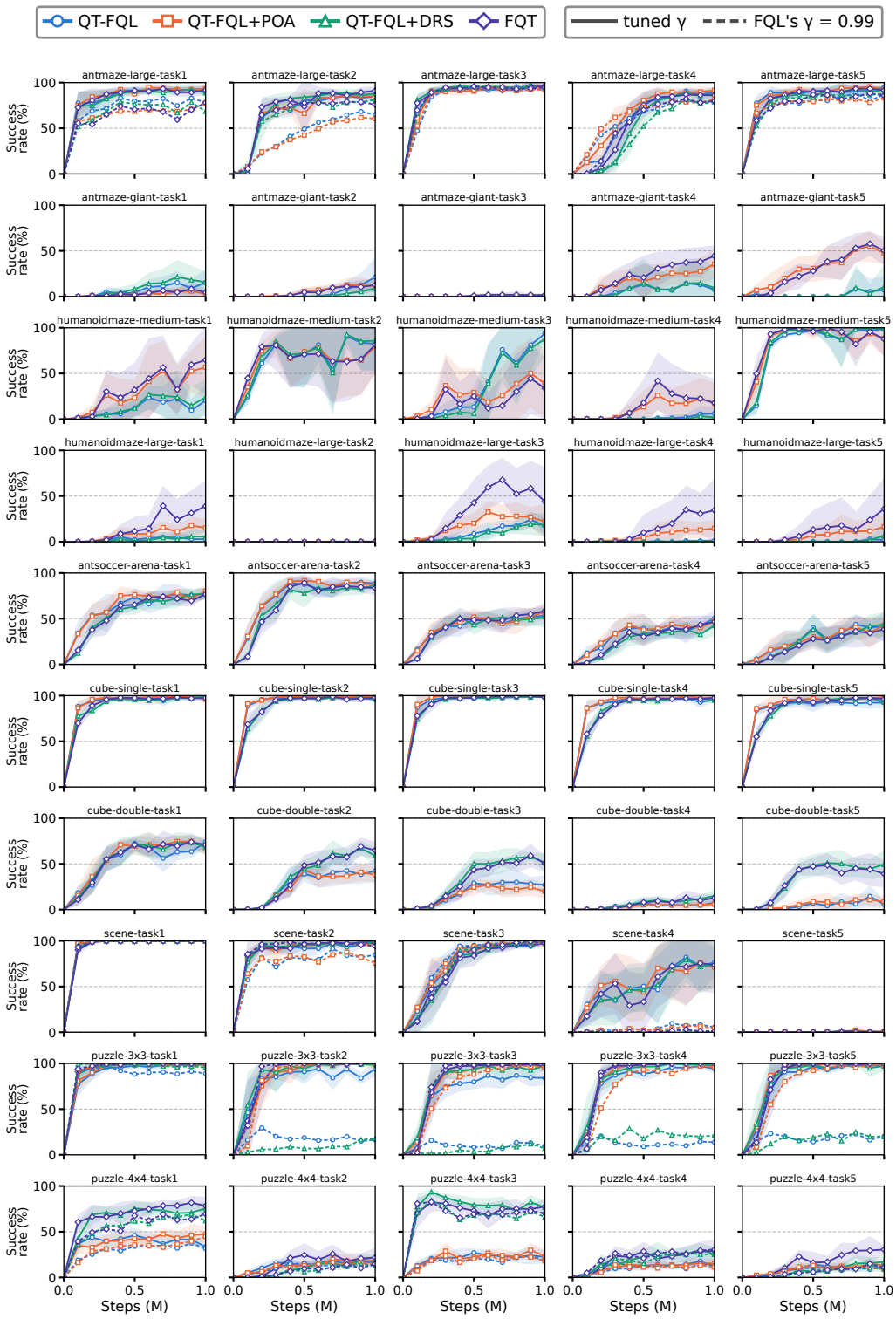
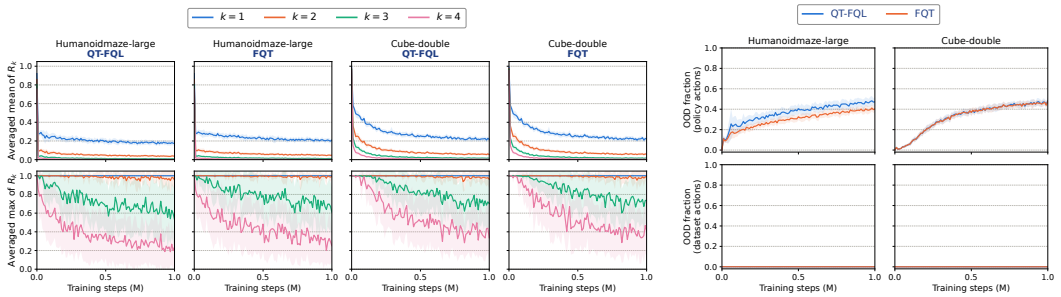


Figure 4: **Task-level learning curves of the QT-FQL variants**, where QT-FQL+POA&DRS is FQT. Solid curves use the tuned discount-factor setting (*QT-FQL Variants* columns of Table 3) and dashed curves the original FQL's discount-factor setting (*QT-FQL Variants* columns of Table 4), which differs only on `antmaze-large`, `scene`, `puzzle-3x3` and `puzzle-4x4`.

D.2 Analysis of the Conditional Density Ratio Estimator

The conditional density ratio estimator Γ_w is used in three parts of FQT: estimating the importance-sampling ratio, identifying OOD actions for POA, and filtering OOD candidate actions for DRS. We analyze whether the learned Γ_w provides meaningful signals for multi-step trace weighting and pointwise OOD action identification.



(a) Ratio-dependent trace weights R_k (b) Fraction of actions identified as OOD

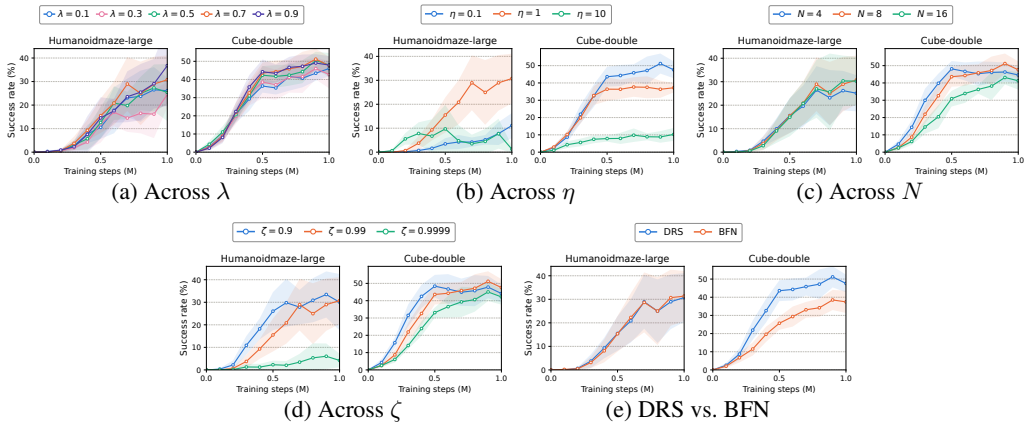
Figure 5: **Analysis of the conditional density ratio estimator** Γ_w for QT-FQL and FQT on `humanoidmaze-large` and `cube-double`. (a) Ratio-dependent cumulative trace weight $R_k = \lambda^{-k}\prod_{i=1}^{k} c_{t+i} = \prod_{i=1}^{k} \min\left(1, \frac{\Gamma_w(s_{t+i},a_{t+i})}{1-\Gamma_w(s_{t+i},a_{t+i})}\right)$ at backup depth k. The mean (top) and maximum (bottom) are computed over sampled sub-trajectories and then averaged over 40 runs, consisting of 5 singletasks over 8 seeds. (b) Fraction of learned-policy actions (top) and dataset actions (bottom) identified as OOD by $\Gamma_w(s,a) \geq \zeta$, averaged over the same 40 runs.

Ratio-dependent cumulative trace weights To examine how the estimated density ratios affect the effective backup depth, we consider the cumulative trace-weight $\prod_{i=1}^{k} c_{t+i}$, where $c_{t+i} = \lambda \min(1, \pi(a_{t+i}|s_{t+i})/\beta(a_{t+i}|s_{t+i}))$. Since this weight contains the deterministic decay factor λ^k, it is upper-bounded by λ^k. We therefore report $R_k = \lambda^{-k}\prod_{i=1}^{k} c_{t+i} = \prod_{i=1}^{k} \min(1, \pi/\beta)$, which removes this deterministic decay factor and isolates the contribution of the estimated density ratios. We report the averaged mean and maximum of R_k for the successive trace depths used in our finite-depth Retrace(λ) target, as specified in Algorithm 1.

Figure 5a shows that the mean R_k decreases with backup depth and also tends to decrease over training, indicating that deeper trace terms are increasingly attenuated by the estimated density ratios. This behavior may be consistent with the learned policy gradually deviating from the behavior policy of dataset. Nevertheless, the averaged maximum R_k remains non-negligible even at the deepest backup depth, showing that some sampled sub-trajectories retain substantial multi-step trace weights. Thus, our multi-step critic target adaptively weights different backup depths according to the estimated density ratios rather than collapsing to purely one-step critic learning.

OOD identification Figure 5b shows the fraction of learned-policy and dataset actions identified as OOD by $\Gamma_w(s,a) \geq \zeta$. The fraction of learned-policy actions identified as OOD steadily increases over training, whereas dataset actions are almost never classified as OOD, indicating a clear separation between learned-policy and dataset actions in terms of OOD identification.

D.3 Ablation Studies



(a) Across λ (b) Across η (c) Across N

(d) Across ζ (e) DRS vs. BFN

Figure 6: **Sensitivity and action-selection analysis** of FQT on `humanoidmaze-large` and `cube-double`: (a) trace-decay parameter λ, (b) critic penalization coefficient η for POA, (c) number of candidate actions N for DRS, (d) OOD identification threshold ζ, and (e) comparison DRS with best-of-N (BFN), which selects the highest-Q candidate without density-ratio-based OOD filtering. For each task category, curves indicate the mean performance over 8 seeds averaged across 5 singletasks, with shaded regions indicating the standard deviation.

Figure 6 shows the sensitivity of FQT to key hyperparameters, as well as the effect of DRS compared with best-of-N (BFN) action selection, which directly selects the candidate action with the highest Q-value without using the conditional density ratio estimator for OOD filtering.

In Figure 6a, FQT tends to be robust to λ on both task categories, maintaining comparable performance even with large λ values. In Figure 6b, selecting an appropriate critic penalization coefficient η is important. Moderate penalization performs best on `humanoidmaze-large`, whereas weaker penalization is preferred on `cube-double`, while overly strong penalization degrades performance on both task categories. In Figure 6c, increasing N tends to improve performance on `humanoidmaze-large`, whereas on `cube-double`, further increasing N does not consistently improve performance. These results suggest that an appropriate choice of N can be important for DRS. In Figure 6d, an overly large ζ degrades performance, particularly on `humanoidmaze-large`. Since a larger ζ makes the OOD criterion more permissive, an appropriate choice of ζ is important for effectively handling potentially OOD actions. Finally, Figure 6e shows that DRS outperforms BFN on `cube-double` while achieving comparable performance on `humanoidmaze-large`. This may support the benefit of filtering potentially OOD candidate actions using the conditional density ratio estimator before Q-value-based action selection.

D.4 COMPUTATIONAL COSTS

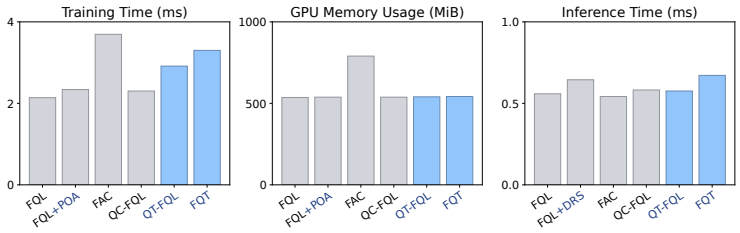


Figure 7: Computational cost comparisons.

Figure 7 compares the computational costs of offline RL methods with flow-based policies on `cube-double-play`, in terms of training time, GPU memory usage, and inference time. All methods are evaluated on the same machine with a single RTX 3090 GPU. Training time is measured as the wall-clock time per gradient step with a batch size of 256, and inference time is measured as the time required to sample a single action.

QT-FQL and FQT require additional training time compared with FQL (Park et al., 2025) and FQL+POA, mainly due to the backward recursion required to construct the Retrace(λ) target. FQL+POA and FQT require additional computation due to the penalizing OOD action (POA) loss term in the critic objective. Nevertheless, GPU memory usage remains comparable across FQL, FQL+POA, QT-FQL, and FQT.

At inference time, FQL+DRS and FQT sample N candidate actions and apply density-ratio-based rejection sampling (DRS). This results in only a modest increase in action-selection time, indicating that DRS introduces limited inference overhead.

E EXPERIMENTAL DETAILS

We implement the proposed algorithm in JAX (Bradbury et al., 2018) on top of an official implementation of FQL (Park et al., 2025).

E.1 BENCHMARKS

We evaluated offline RL algorithms on 50 state-based singletasks of OGBench (Park et al., 2024). Specifically, we considered the following tasks from 10 `navigation` and `manipulation` environments:

- `antmaze-large-navigate-singletask-task{1,2,3,4,5}-v0`
- `antmaze-giant-navigate-singletask-task{1,2,3,4,5}-v0`
- `humanoidmaze-medium-navigate-singletask-task{1,2,3,4,5}-v0`
- `humanoidmaze-large-navigate-singletask-task{1,2,3,4,5}-v0`
- `antsoccer-arena-navigate-singletask-task{1,2,3,4,5}-v0`
- `cube-single-play-singletask-task{1,2,3,4,5}-v0`
- `cube-double-play-singletask-task{1,2,3,4,5}-v0`
- `scene-play-singletask-task{1,2,3,4,5}-v0`
- `puzzle-3x3-play-singletask-task{1,2,3,4,5}-v0`
- `puzzle-4x4-play-singletask-task{1,2,3,4,5}-v0`

OGBench (Park et al., 2024) provides a benchmark suite of environments and datasets for offline goal-conditioned RL. In this work, we use datasets from 10 environments, consisting of 5 `navigation` environments (`antmaze-large`, `antmaze-giant`, `humanoidmaze-medium`, `humanoidmaze-large`, and `antsoccer-arena`) and 5 `manipulation` environments (`cube-single`, `cube-double`, `scene`, `puzzle-3x3`, and `puzzle-4x4`). Each environment provides five singletask variants. For offline RL evaluation, we use the reward-based singletask variants, where the rewards in the dataset transitions are relabeled to match the corresponding singletask specification. In the `navigation` tasks, a sparse reward is used: the agent receives a reward of 0 upon successfully reaching the goal and -1 otherwise. In the `manipulation` tasks, a semi-sparse reward is used: each singletask consists of multiple subtasks, and the reward is defined as the negative number of unsuccessful subtasks.

The `navigation` environments include `antmaze` and `humanoidmaze`, where a quadrupedal agent and a humanoid agent, respectively, must navigate a maze to reach a goal location; `antsoccer`, where a quadrupedal agent is required to dribble a ball to a target position. The `manipulation` environments include `cube`, where a robotic arm must pick and place colored cubes; `scene`, where a robotic arm performs a sequence of manipulation subtasks; `puzzle`, where a robotic arm solves a lights-out puzzle.

E.2 IMPLEMENTATION DETAILS

Network architectures We use a [512, 512, 512, 512]-sized MLP for all neural networks. We apply layer normalization (Ba et al., 2016) to critic networks. The critic network Q_ϕ takes a state-action pair (s, a) as input and outputs a scalar estimate of the expected return under the one-step flow policy π_θ. The Behavior cloning flow policy β_ψ takes state s, Gaussian noise $z \in \mathbb{R}^{d_\mathcal{A}}$, and flow time $u \in [0, 1]$ as input and outputs the velocity field v_u. The one-step flow policy π_θ takes state s and Gaussian noise z and then outputs a single action $a_\theta(s, z)$. The conditional density ratio estimator $\Gamma_w(s, a)$ takes a state-action pair (s, a) as input and outputs a logit $\gamma_w(s, a)$ for the density ratio estimates.

Conditional density ratio estimation We train a conditional density ratio estimator with the binary cross entropy objective (11). In practice, the estimator outputs a logit $\gamma_w(s, a)$, and we compute the density ratio by $\Gamma_w(s, a) = \frac{\pi(a|s)}{\pi(a|s)+\beta(a|s)} = \sigma\left(\gamma_w(s, a)\right) \in [0, 1]$, where $\sigma(x)$ is sigmoid function $\frac{1}{1+e^{-x}}$. Using this logit γ_w, we can easily compute the binary cross entropy loss (11):

$$\begin{aligned} L(\Gamma_w) &= -\mathbb{E}_{s\sim\mathcal{D}}\left[\mathbb{E}_{a\sim\pi}\left[\log \Gamma_w(s, a)\right] + \mathbb{E}_{a\sim\beta}\left[\log(1 - \Gamma_w(s, a))\right]\right] \\ &= -\mathbb{E}_{s\sim\mathcal{D}}\left[\mathbb{E}_{a\sim\pi}\left[\log \sigma\left(\gamma_w(s, a)\right)\right] + \mathbb{E}_{a\sim\mathcal{D}}\left[\log \sigma\left(-\gamma_w(s, a)\right)\right]\right], \end{aligned}$$

and off-policy importance sampling correction can also be computed as $\frac{\pi(a|s)}{\beta(a|s)} = \frac{\Gamma_w(s,a)}{1-\Gamma_w(s,a)} = \frac{\sigma(\gamma_w(s,a))}{1-\sigma(\gamma_w(s,a))} = \frac{1}{1+e^{-\gamma_w(s,a)}} \cdot \frac{1+e^{-\gamma_w(s,a)}}{e^{-\gamma_w(s,a)}} = \exp(\gamma_w(s,a))$.

Flow matching We train a flow matching model with the flow matching objective (4) (Lipman et al., 2022; Park et al., 2025). We use the Euler method with 10 steps to sample actions $a_\psi(s,z)$ from behavior cloning flow policy $\beta_\psi(a|s)$, which serves as an approximated behavior policy of the dataset for actor regularization in eq. (5).

Value learning We use twin critic networks to improve stability (Fujimoto et al., 2018). Practically, we can aggregate them either by the arithmetic mean of the two Q values or by their minimization. We present the aggregation choices in Table 5.

Training and evaluation For offline RL results, we train the RL algorithms for 1M gradient steps on all the state-based tasks of OGBench. We evaluate them every 100K gradient steps using 50 episodes. Following the evaluation scheme (Park et al., 2024; 2025), we report the average success rates across the last three evaluation epochs (800K, 900K, and 1M gradient steps).

E.3 Methods and Hyperparameters

In this paper, we consider three offline RL baselines. FQL (Park et al., 2025) uses a discount factor of $\gamma = 0.995$ for `antmaze-giant`, `humanoidmaze-{medium, large}`, and `antsoccer`, and $\gamma = 0.99$ for the remaining task categories. In contrast, FAC (Chae et al., 2026) uses $\gamma = 0.995$ for all task categories, while QC-FQL (Li et al., 2025) considers only $\gamma = 0.99$. To ensure a fair comparison, we include both $\gamma \in \{0.99, 0.995\}$ in the hyperparameter search space for all methods. For each method, we select the best hyperparameter configuration on the default task of each task category and use the selected configuration for the remaining tasks in the same category. For all reproduced offline RL results, we report the mean and standard deviation over 8 random seeds.

We use the reported performance of Behavior Cloning (BC) from Park et al. (2025). For FQL, FAC, and QC-FQL, we follow the hyperparameter tuning protocols described below.

Flow Q-Learning (FQL) For FQL (Park et al., 2025), we consider discount factors $\gamma \in \{0.99, 0.995\}$. For the behavior-regularization coefficient α in the actor objective, we search over values obtained by multiplying the task-specific value reported in the FQL paper by $\{1/10, 1/3, 1, 3, 10\}$. Among the tested discount factors, $\gamma = 0.995$ achieves the best performance for all task categories except `cube-{single,double}`, for which $\gamma = 0.99$ performs best. The selected hyperparameters are provided in Table 6.

For `antmaze-large`, `scene`, and `puzzle-{3x3,4x4}`, our tuned discount-factor setting selects $\gamma = 0.995$, whereas the original FQL results use $\gamma = 0.99$. To enable a direct comparison under the discount factor used in the original FQL evaluation and to separate the effect of changing γ from that of the proposed method, we additionally provide selected hyperparameters with $\gamma = 0.99$ for these four task categories in Table 7. Under $\gamma = 0.99$, our expanded search selects $\alpha = 3000$ for `puzzle-4x4`, which achieves higher success rates on the default tasks than the original FQL choice of $\alpha = 1000$; the original FQL hyperparameter search considered α up to 1000.

Flow Q-Learning with two OOD Action Handling Techniques (FQL+POA&DRS) We augment the standard one-step RL method, FQL, with the two OOD action handling techniques introduced in our method, Penalizing OOD Actions (POA) and Density-ratio-based Rejection Sampling (DRS), using the proposed conditional density ratio estimator. This variant serves as a one-step counterpart to QT-FQL and FQT, allowing us to assess whether the improvements from POA and DRS alone can account for the performance of our multi-step method. We reuse the discount factor γ and behavior-regularization coefficient α selected for FQL without further tuning, and consider the critic penalization coefficient $\eta \in \{0.1, 1, 10\}$ for POA and the number of candidate actions $N \in \{4, 8, 16\}$ for DRS. The resulting hyperparameter configurations are provided in Tables 6 and 7.

Flow Actor-Critic (FAC) We reproduced FAC using its official open-source implementation (Chae et al., 2026). In the original FAC paper, its offline RL results are reported at the final training epoch

using 25 episodes. To align with the OGBench evaluation protocol (Park et al., 2024) and other methods, we reproduced FAC using the hyperparameter configurations provided in the paper, and report the averaged performance over the last three evaluation epochs (800K, 900K, and 1M gradient steps).

Q-Chunking with Wasserstein Distance Behavior Constraint (QC-FQL) We reproduced QC-FQL using the official open-source implementation of Q-chunking (Li et al., 2025). While the QC paper considers only the discount factor $\gamma = 0.99$, we additionally consider $\gamma = \{0.99, 0.995\}$ to enable a fairer comparison with other methods. For the behavior-regularization coefficient α, we search over values obtained by multiplying the optimal task-specific α of FQL (Park et al., 2025) by $\{1/10, 1/3, 1, 3, 10, 30\}$. Following the original method, we set the action chunk size to 5.

For `antmaze-giant` and `humanoidmaze-large`, QC-FQL achieves almost zero success rate for all tested hyperparameter configurations on the default tasks. In these tasks, we therefore use the hyperparameter configurations selected on the corresponding easier task categories, `antmaze-large` and `humanoidmaze-medium`, respectively.

Q-trace with Wasserstein Distance Behavior Constraint (QT-FQL) QT-FQL is the simplest form of our proposed Flow Q-Trace, obtained by removing the two components that use the conditional density ratio estimator for OOD action handling, Penalizing OOD Actions (POA) and Density-ratio-based Rejection Sampling (DRS), i.e., **FQT w/o POA&DRS**. Accordingly, QT-FQL can be viewed as a variant of FQL (Park et al., 2025) that replaces its standard one-step temporal-difference (TD) target with a Retrace(λ) target using the proposed conditional density ratio estimator.

For hyperparameter tuning, we consider discount factors $\gamma \in \{0.99, 0.995\}$, behavior-regularization coefficients α obtained by multiplying the task-specific FQL value by $\{1/3, 1, 3\}$, and bias-variance trade-off parameters $\lambda \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. Following QC-FQL, we use a sub-trajectory length of $l = 5$ for multi-step critic learning.

Flow Q-Trace (FQT) To avoid excessive hyperparameter tuning and maintain a controlled comparison with QT-FQL, we reuse the hyperparameters $(\gamma, \alpha, \lambda, l)$ selected for QT-FQL. Motivated by Lemma 2 and accounting for inevitable function approximation error in the conditional density ratio estimator, we set the OOD identification threshold to $\zeta = 0.99$. We then consider only the hyperparameters associated with the two OOD action handling techniques: the critic penalization coefficient $\eta \in \{0.1, 1, 10\}$ for Penalizing OOD Actions (POA) and the number of candidate actions $N \in \{4, 8, 16\}$ for Density-ratio-based Rejection Sampling (DRS).

We provide the hyperparameter configurations of all methods in Table 5, 6, and 7.

Table 5: Hyperparameters for the proposed method

Hyperparameters	**Value**
Learning rate	0.0003
Optimizer	Adam (Kingma & Ba, 2014)
Gradient Steps	1000000
Minibatch size	256
MLP dimension	[512, 512, 512, 512]
Nonlinearity	GELU (Hendrycks & Gimpel, 2016)
Target network smoothing coefficient κ	0.005
Step count of the Euler method	10
Flow time sampling distribution	Unif($[0, 1]$)
Q-normalization in actor loss	False
Clipped double Q-learning	False (default), True(`antmaze-{large, giant}-navigate`)
Length of sub-trajectory l	5
Discount factor γ	Table 6&7
BC coefficient α for actor learning	Table 6&7
Bias-variance trade-off parameter λ	Table 6&7
OOD identification threshold ζ	0.99
Critic penalization coefficient η for POA	Table 6&7
Number of candidate actions N for DRS	Table 6&7

Table 6: Task-specific hyperparameters for offline RL methods, used for the results in Table 1 and for the tuned discount-factor setting in Table 2.

Task Category	FQL+POA&DRS $(\gamma, \alpha) + (\eta, N)$	FAC $(\gamma, \alpha_{\text{critic}}, \lambda_{\text{actor}}, \epsilon)$	QC-FQL (γ, α)	QT-FQL+POA&DRS (i.e., FQT) $(\gamma, \alpha, \lambda) + (\eta, N)$
antmaze-large-navigate-singletask-task{1,2,3,4,5}	(0.995, 30) + (0.1, 4)	(0.995, 0.5, 0.1, Dataset-wide)	(0.995, 100)	(0.995, 30, 0.7) + (0.1, 4)
antmaze-giant-navigate-singletask-task{1,2,3,4,5}	(0.995, 10) + (1.0, 4)	(0.995, 1.0, 0.1, Dataset-wide)	(0.995, 100)	(0.995, 10, 0.5) + (1.0, 4)
humanoidmaze-medium-navigate-singletask-task{1,2,3,4,5}	(0.995, 30) + (1.0, 4)	(0.995, 0.5, 0.3, Batch-adaptive)	(0.995, 100)	(0.995, 30, 0.9) + (1.0, 4)
humanoidmaze-large-navigate-singletask-task{1,2,3,4,5}	(0.995, 30) + (1.0, 8)	(0.995, 0.5, 0.1, Batch-adaptive)	(0.995, 100)	(0.995, 30, 0.7) + (1.0, 8)
antsoccer-arena-navigate-singletask-task{1,2,3,4,5}	(0.995, 10) + (0.1, 4)	(0.995, 1.0, 0.1, Dataset-wide)	(0.995, 100)	(0.995, 10, 0.9) + (0.1, 4)
cube-single-play-singletask-task{1,2,3,4,5}	(0.99, 300) + (1.0, 4)	(0.995, 0.5, 10.0, Batch-adaptive)	(0.995, 1000)	(0.99, 300, 0.7) + (1.0, 4)
cube-double-play-singletask-task{1,2,3,4,5}	(0.99, 300) + (0.1, 8)	(0.995, 1.0, 1.0, Batch-adaptive)	(0.995, 1000)	(0.99, 300, 0.7) + (0.1, 8)
scene-play-singletask-task{1,2,3,4,5}	(0.995, 300) + (0.1, 4)	(0.995, 1.0, 1.0, Batch-adaptive)	(0.995, 300)	(0.995, 300, 0.7) + (0.1, 4)
puzzle-3x3-play-singletask-task{1,2,3,4,5}	(0.995, 300) + (1.0, 8)	(0.995, 1.0, 1.0, Batch-adaptive)	(0.995, 3000)	(0.995, 300, 0.5) + (1.0, 8)
puzzle-4x4-play-singletask-task{1,2,3,4,5}	(0.995, 3000) + (10.0, 8)	(0.995, 5.0, 0.3, Batch-adaptive)	(0.995, 3000)	(0.995, 3000, 0.5) + (10.0, 8)

Table 7: Additional task-specific hyperparameters for comparisons under discount-factor provided on FQL (Park et al., 2025), used for the original FQL's discount-factor setting in Table 2. For the other task categories, the hyperparameters in Table 6 are used.

Task Category	FQL+POA&DRS $(\gamma, \alpha) + (\eta, N)$	QT-FQL+POA&DRS (i.e., FQT) $(\gamma, \alpha, \lambda) + (\eta, N)$
antmaze-large-navigate-singletask-task{1,2,3,4,5}	(0.99, 10) + (0.1, 4)	(0.99, 10, 0.7) + (0.1, 4)
scene-play-singletask-task{1,2,3,4,5}	(0.99, 300) + (0.1, 4)	(0.99, 300, 0.7) + (0.1, 4)
puzzle-3x3-play-singletask-task{1,2,3,4,5}	(0.99, 1000) + (10.0, 8)	(0.99, 1000, 0.5) + (10.0, 8)
puzzle-4x4-play-singletask-task{1,2,3,4,5}	(0.99, 3000) + (1.0, 8)	(0.99, 3000, 0.5) + (1.0, 8)