NOISE-CONDITIONED Q-ADJOINT ANCHORING

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose **Noise-conditioned Q-Adjoint Anchoring (NQA)**, an inference-efficient technique that advances the state-of-the-art in offline-to-online reinforcement learning (RL). Expressive RL approaches combining generative policies, distributional critics, and action chunking achieve strong performance, yet incur substantial computational overhead. For efficient inference, multi-step integration in diffusion and flow policies can be replaced with straight flow sampling, but existing approaches encounter two expressivity limitations during training. Specifically, they struggle to (1) directly leverage critic guidance along straight flow trajectories and (2) capture distributional returns while reducing bootstrapping bias. To address these limitations, NQA introduces two algorithmic innovations. First, for policy learning, **Q-Adjoint Anchoring** applies Adjoint Matching to propagate critic gradients along straight flow trajectories while eliminating multi-step forward SDE integration. Second, for value learning, **Noise-conditioned Q-Chunking** estimates optimistic values within the return distribution while mitigating bias accumulation and maintaining a short policy execution horizon. In robotic locomotion and manipulation tasks, NQA achieves both high computational efficiency and strong task performance. To the best of our knowledge, NQA is the first method to achieve an average success rate above 90% on one of the most challenging tasks in OGBench, `humanoidmaze-giant-navigate`. [Code]

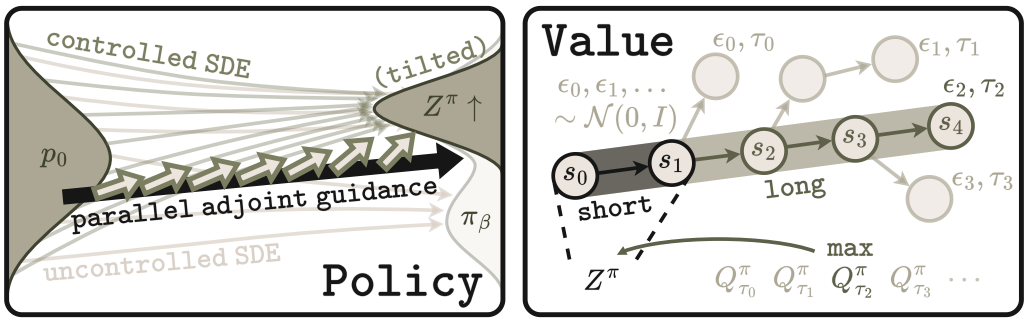


Figure 1: **Overview of NQA.** (*Left*) **Q-Adjoint Anchoring** regularizes straight action trajectories toward flow representing the "tilted" behavior policy distribution. (*Right*) **Noise-conditioned Q-Chunking** models full return distributions and extracts the maximum for the short-horizon policy.

1 INTRODUCTION

Offline-to-online reinforcement learning (RL) seeks optimal sequential decision-making strategies through the active *reuse of data*. Since data collection can be expensive, this line of work first trains policies offline (Fujimoto et al., 2019; Levine et al., 2020) on pre-collected datasets, and subsequently fine-tunes them online (Nakamoto et al., 2023; Ball et al., 2023; Lee & Kwon, 2025) while expanding the dataset through direct environment interactions. Because these datasets are often collected by a diverse mixture of behavior policies, effective data reuse calls for *expressive* policy and value function classes capable of capturing complex probability distributions in the data.

Expressivity has therefore been actively pursued in both policy and value learning. For policy learning, diffusion/flow-based modeling (Ho et al., 2020; Song et al., 2020; Chi et al., 2025) and action chunking (Lai et al., 2022; Zhao et al., 2023) are two widely adopted approaches. Diffusion and flow

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.

policies address the limited ability of unimodal Gaussian policies to represent multimodal action distributions, whereas action chunks enable policies to model temporally extended action sequences. For value learning, two prominent directions are distributional RL (Bellemare et al., 2017) and Q-chunking (Li et al., 2026b). Distributional critics model return distributions rather than expected returns alone, whereas Q-chunking takes action chunks as critic inputs to reduce bias accumulation from single-horizon bootstrapping and off-policy execution. However, such expressivity gains often come at the cost of increased computational complexity. For instance, diffusion and flow policies require iterative sampling to solve the underlying stochastic or ordinary differential equations (SDEs/ODEs), while distributional critics require multiple quantile/expectile estimates or repeated evaluations across noise samples. Action chunking further increases the computational burden.

Consequently, generative policies that omit timestep inputs are highly desirable for inference-efficient offline-to-online RL. Unlike standard generative policies that require multi-step integration, these policies bypass iterative sampling to drastically reduce latency in time-sensitive domains such as robotics. While theoretically retaining the flexibility to model complex action distributions (Mazoure et al., 2020), they also remain computationally efficient when critic bootstrapping requires evaluating future policy outputs, similar to other inference-efficient approaches (Espinosa-Dice et al., 2025; Wang et al., 2026). Despite these advantages, prior work has largely focused on iterative policies with intermediate timestep inputs, motivating the following research question:

"Can straight flow policies without timestep inputs solve challenging offline-to-online RL tasks?"

Existing approaches suffer from two primary limitations.

1. In the absence of multi-step flow iterations, directly leveraging critic gradients *at the flow-trajectory level* is challenging, even though such guidance significantly improves multi-step policies (Li & Levine, 2026). This is because generative policies without timestep inputs bypass intermediate flow states, only indirectly modeling the straight flow trajectories.
2. Current distributional RL methods suffer from *bias accumulation*. While critics with action chunk inputs reduce bootstrapping and off-policy bias (Li et al., 2026b;a; Song et al., 2026), existing distributional critics do not benefit from this architecture (Dabney et al., 2018a; Dong et al., 2026c), defaulting instead to standard temporal-difference (TD) learning.

To address these problems, we propose **Noise-conditioned Q-Adjoint Anchoring (NQA)**. Figure 1 shows the high-level overview with two components targeting policy and value learning.

1. **Q-Adjoint Anchoring** applies Adjoint Matching (Domingo i Enrich et al., 2025; Li & Levine, 2026) to propagate critic gradients along the flow, while compressing iterative forward SDE integration into one-step flow sampling. The key intuition is that a straight flow trajectory can be interpreted as a forward SDE trajectory, where Brownian noise and the control term jointly correct drift to maintain straight paths (Figure 5, left).
2. **Noise-conditioned Q-chunking** adopts a noise-conditioned critic (Lee et al., 2026) to model return distributions, mitigating bias accumulation via Q-chunking while overcoming the challenges of long-horizon policy execution. Our insight is that tracking optimistic returns with a noise-conditioned critic enables both a long critic action input and a short policy execution horizon (Figure 5, right).

Across both offline and offline-to-online RL experiments, NQA not only outperforms prior approaches, but is also the first to solve one of the most challenging tasks in recent robotics benchmarks: on the `humanoidmaze-giant-navigate-singletask` task in OGBench (Park et al., 2025a), NQA achieves over 90% average success rate with the highest inference efficiency, while remaining training-efficient even with distributional critics. These results demonstrate the efficiency and expressivity of our approach in pushing the frontier of offline-to-online RL.

Our contributions are threefold:

1. **Q-Adjoint Anchoring** bypasses the multi-step forward SDE integration of Adjoint Matching, guiding the policy with critic gradients propagated directly along the straight flow.
2. **Noise-conditioned Q-chunking** captures distributional returns while mitigating bootstrapping and off-policy bias accumulation, and also maintains short policy execution horizon.

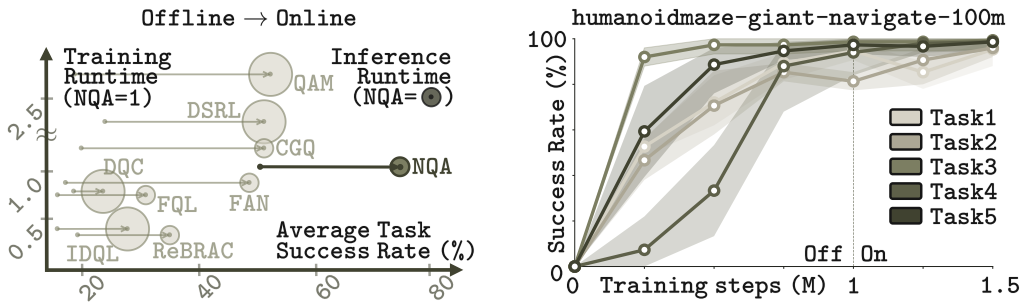


Figure 2: **Performance.** (*Left*) NQA demonstrates high task performance and efficiency. (*Right*) NQA solves long-horizon humanoid navigation tasks while considering a horizon length of 15.

3. As in Figure 2, **NQA** solves challenging continuous control tasks with peak inference efficiency. Moreover, considering for longer horizons improves performance on OGBench `humanoidmaze-giant-navigate`, unlocking the potential of scalable off-policy RL.

2 Related Work

2.1 Expressive Policy Learning

Diffusion and Flow Policies can represent multimodal action distributions beyond conventional unimodal Gaussian policies, and have been widely adopted as expressive policy classes for continuous control (Yang et al., 2023; Zhang et al., 2024; Ren et al., 2025; Chi et al., 2025; McAllister et al., 2026; Lyu et al., 2026). Existing methods can be broadly categorized by whether they use *backpropagation through time (BPTT)*, where BPTT directly conduct backpropagation on critic gradients through iterative denoising or flow trajectories (Wang et al., 2023b; He et al., 2023; Wang et al., 2024b; Ding & Jin, 2024). However, BPTT incurs high computational costs and unstable gradients, motivating increasing interest in non-BPTT approaches. Non-BPTT methods can be categorized by whether value-gradient information is used *(1) only at terminal actions* or *(2) also at intermediate generative timesteps*. The first line of work typically shows high training (Chen et al., 2024a;b) or inference efficiency (Wang et al., 2025; Nguyen & Yoo, 2026), including approaches with rejection sampling (Hansen-Estruch et al., 2023), behavior regularization (Gao et al., 2025), or latent-space optimization (Wagenmaker et al., 2025). Methods in the second category provide more fine-grained policy optimization through reweighted score matching (Ma et al., 2025a), guidance (Fang et al., 2025; Zhou et al., 2026), lean adjoints (Li & Levine, 2026; Dong et al., 2026d), or timestep-conditioned value functions (Doo et al., 2026; Oberai et al., 2026). **NQA** belongs to the second category, but combines ideas from the first category to achieve high efficiency.

Action chunking deals with multi-step action sequences to capture temporal correlations and trajectory-level behaviors (Lai et al., 2022). The concept has been successfully applied to imitation learning (Zhao et al., 2023) and RL (Park et al., 2026a; Kim et al., 2026; Seo & Abbeel, 2026), and has further scaled to large vision-language-action (VLA) models for complex sequential decision-making problems (Black et al., 2024; Physical Intelligence et al., 2025; Dong et al., 2026b;a). Long-horizon action chunks suit well with diffusion and flow models, because of their ability to model complex high-dimensional distributions (Janner et al., 2022; Karunratanakul et al., 2024; Seo et al., 2026). In this work, **NQA** structures its policy to output action chunks, leveraging flow models to capture the complex distributions induced by sequence-level action modeling.

2.2 Expressive Value Learning

Distributional RL models the full distribution of returns rather than their expectations (Morimura et al., 2010). This distributional return information is used to address risk-sensitive (Kim et al., 2023; Ma et al., 2025b), risk-seeking (Cho et al., 2023), or standard RL problems (Ma et al., 2021) with stronger theoretical guarantees (Wang et al., 2023a; 2024a; Cho et al., 2025). While prior work deals with quantiles (Dabney et al., 2018b;a), expectiles (Rowland et al., 2019), or generative modeling (Dong et al., 2026c), it often suffers from high computational overhead. In contrast, **NQA** efficiently captures return distributions by extending the work of Lee et al. (2026) beyond $TD(0)$.

Q-chunking. Multi-step TD learning is motivated by the limitations of $TD(0)$, i.e., single-transition TD learning, whose updates suffer from high bias and bootstrapping errors. Updating across n transitions reduces bias at the cost of increased variance (Sutton et al., 1998; Watkins, 2026). Recently, Q-chunking (Li et al., 2026b) showed that passing action chunks directly to the critic mitigates off-policy and bootstrapping bias accumulated by $TD(0)$. However, it can suffer from suboptimal policies with actions fixed throughout long execution horizon. To address this, decoupled Q-chunking (DQC) (Li et al., 2026a; Song et al., 2026) decouples the critic's action input horizon from the policy's execution horizon, preserving the benefit of Q-chunking while reducing the chunk size of the policy. While similar multi-step abstractions have been extended to model-based RL (Park et al., 2026a; Chung et al., 2026), **NQA** is model-free and extends the idea of DQC to distributional critics.

3 PRELIMINARIES

For additional details, please refer to the extended preliminaries in Appendix A.

Problem setting. We consider a Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, \rho, P, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A} \subseteq \mathbb{R}^d$ is the d-dimensional action space, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $\gamma \in (0, 1)$ is the discount factor. Let $\Delta(\mathcal{X})$ denote the set of probability distributions over a space $\mathcal{X}$. The initial-state distribution is $\rho \in \Delta(\mathcal{S})$, and the transition kernel is $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$. We consider a policy $\pi : \mathcal{S} \to \Delta(\mathcal{A}^n)$ that predicts a sequence of n consecutive actions conditioned on the current state, where $n > 1$ corresponds to *action chunking*. Let $a_{0:n-1} = (a_0, \dots, a_{n-1}) \in \mathcal{A}^n$ denote an n-step action chunk, and let $m = nd$ be the total dimension of the policy output. The goal is to find a policy π that maximizes the discounted return: $Q^\pi(s, a) = \mathbb{E}_{\tau \sim P^\pi(\cdot|s_0=s,\, a_{0:n-1}=a)} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$, where $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \dots)$ denotes a trajectory generated by executing the action chunk $a \in \mathcal{A}^n$ for the first n steps and following π thereafter. The policy is trained using the dataset (or replay buffer) $\mathcal{D} = \{\tau^{(i)}\}_{i=1}^N$ of size N, collected by a set of behavior policies that may differ from π. The dataset is fixed in the offline setting and may be continually updated through environment interaction in the online setting. For each state s, let $p_{\mathcal{D}}(\cdot|s) \in \Delta(\mathcal{A}^n)$ denote the conditional distribution of action chunks in $\mathcal{D}$. We parameterize this distribution using a behavior policy model π_β, and parameterize π with π_θ.

Adjoint Matching (Domingo i Enrich et al., 2025) aims to learn an exponentially tilted distribution $\tilde{p}_{\mathcal{D}} \propto p_{\mathcal{D}} \exp(Q)$, which is a solution to the following stochastic optimal control (SOC) problem:

$$\begin{aligned} \min_u \quad & \mathbb{E}\left[\frac{1}{2}\int_0^1 \|u(X_t^u, t; s)\|_2^2 \, dt - Q(s, X_1^u)\right] \\ \text{s.t.} \quad & dX_t^u = [b_\beta(X_t^u, t; s) + \sigma(t) u(X_t^u, t; s)]\, dt + \sigma(t) dB_t, \qquad X_0^u \sim p_0. \end{aligned} \tag{1}$$

Here, the expectation is taken over SDE trajectories. b_β is the SDE base drift and $\sigma(t)$ is the time-dependent diffusion coefficient of the SDE, and both are fixed for a given SOC problem. X_t^u is the state of the stochastic process, B_t is Brownian motion, and u is the control function. To obtain the optimal control solution u, Domingo i Enrich et al. introduced the lean adjoint $\tilde{a}_t$ that satisfies $\frac{d}{dt}\tilde{a}_t = -\left[\nabla_x b_\beta(X_t^{\bar{u}}, t; s)\right]^\top \tilde{a}_t$, and $\tilde{a}_1 = -\nabla_x Q(s, X_1^{\bar{u}})$. This leads to the Adjoint Matching loss:

$$\mathcal{L}_{\mathrm{AM}}(u) = \mathbb{E}\left[\int_0^1 \frac{1}{2}\left\|u(X_t^{\bar{u}}, t; s) + \sigma(t)^\top \tilde{a}_t\right\|_2^2 dt\right], \qquad \bar{u} = \mathrm{stopgrad}(u), \tag{2}$$

where they first generate the forward SDE controlled process $X_t^{\bar{u}}$, and perform vector–Jacobian products (VJPs) to numerically solve the ODE of $\tilde{a}_t$. **NQA** embraces these forward and backward processes, but replaces forward numerical integration with a single-step flow sampling.

Flow Anchoring (Lee et al., 2026) enables efficient behavior regularization of straight flow policies. The main idea is to bypass iterative sampling during training, using the following objective:

$$\mathcal{L}_{\mathrm{FA}}(\theta) = \mathbb{E}_{s\sim\mathcal{D}, z\sim p_0, t\sim\mathcal{U}(0,1)}\left[\left\| \left(\mu_\theta^\pi(s, z) - z\right) - v_\beta\big((1-t)\, z + t\, \bar{\mu}_\theta^\pi(s, z), t; s\big)\right\|_2^2\right]. \tag{3}$$

Here, μ_θ^π is the straight flow action sampler for the learning policy π_θ with $\bar{\mu}_\theta^\pi = \mathrm{stopgrad}(\mu_\theta^\pi)$, and v_β is the velocity field that forms the behavior policy distribution π_β. Minimizing Eq.3 is known to minimize an upper bound on the Wasserstein-2 (W_2) distance between π_β and π_θ. **NQA** adopts this efficient training mechanism, but anchors μ_θ^π to the flow of the tilted distribution.

Noise-conditioned Q-Learning (NQL). Lee et al. (2026) recently noted that tracking multiple distributional critic samples is inefficient since the optimal value changes during policy improvement. To this end, NQL mainly tracks optimistic value estimates within the return distribution:

$$\begin{aligned}\mathcal{L}_Q(\phi) &= \mathbb{E}_{\substack{(s,a,r,s')\sim\mathcal{D},\\ \epsilon'\sim\mathcal{N}(0,I_d)}}\left[\left(Q_\phi(s,a,\epsilon') - (r+\gamma\, Z_\psi(s', \mu^\pi_\theta(s',\epsilon')))\right)^2\right], \\ \mathcal{L}_Z(\psi) &= \mathbb{E}_{\substack{(s,a)\sim\mathcal{D},\\ \epsilon\sim\mathcal{N}(0,I_d)}}\left[L_2^\kappa\left(Q_{\hat{\phi}}(s,a,\epsilon) - Z_\psi(s,a)\right)\right].\end{aligned} \tag{4}$$

Here, $L_2^\kappa(\hat{x}-x) = |\kappa - \mathbf{1}((\hat{x}-x)<0)|(\hat{x}-x)^2$ lets x capture the κ-expectile of the $\hat{x}$ distribution (Newey & Powell, 1987). ϕ parameterizes the distributional critic with its target network parameter $\hat{\phi}$, and ψ is the parameter for optimistic value estimates targeting for $\lim_{\kappa\to 1^-} Z_\psi(s,a) \approx \operatorname{ess\,sup}_\epsilon Q_\phi(s,a,\epsilon)$. Following Lee et al. (2026), we fix $\kappa = 0.9$ across all experiments, and compute the adjoints with respect to Z_ψ while incorporating the Q-chunking formulation.

4 Why should we use Adjoints and Distributional Returns?

In offline-to-online RL, directly maximizing the value can lead to severe out-of-distribution (OOD) action choices and training instability (Kumar et al., 2019; Nair et al., 2020; Fedus et al., 2020). To prevent OOD actions, we focus on the following behavior-constrained value maximization problem:

$$\max_\pi\ \mathbb{E}_{a\sim\pi(\cdot|s)}\left[Q(s,a)\right] \qquad \text{s.t.} \qquad D_{\mathrm{KL}}\left(\pi(\cdot|s)\ \|\ p_{\mathcal{D}}(\cdot|s)\right) \le \delta, \tag{5}$$

where $\delta > 0$ bounds the reverse Kullback–Leibler (KL) divergence, strictly preventing actions outside the dataset action distribution $p_{\mathcal{D}}$. The closed-form solution to Eq.5 is an exponentially tilted distribution, and Adjoint Matching provides a practical sampling method for this solution. To this end, we begin by analyzing how adjoints and expressive Q-value estimates affect offline-to-online RL performance. Specifically, we evaluate on the default `singletask` robotic manipulation environments `cube-double-play` and `puzzle-4x4-play` from OGBench (Park et al., 2025a).

Observation 1: Adjoints provide better solutions than terminal critic gradients. Multi-step flow methods, *e.g.*, Q-Adjoint Matching (QAM) (Li & Levine, 2026), effectively solve challenging continuous control tasks. Their key distinction from non-iterative flow methods is that they propagate critic gradient information along the entire flow trajectory (*e.g.*, adjoints in QAM). To evaluate the necessity of this trajectory-propagated gradient information, we compare QAM against four variants that omit propagation: *(i) QAM-SDE-endpoint*, *(ii) QAM-ODE-endpoint*, *(iii) QAM-SDE-QGF*, and *(iv) QAM-ODE-QGF*. The *SDE* and *ODE* variants use forward SDE and ODE integrations for action sampling. The *endpoint* variants obtain terminal value gradients with respect to each sampled action, and guide intermediate velocity fields using these gradients. Similarly, the *QGF* variants also guide intermediate velocity fields with terminal value gradients, but compute them with respect to the estimated target action $\widehat{x}_1(t) = x_t + (1-t)v_\beta(s,x_t,t)$, following QGF (Zhou et al., 2026).

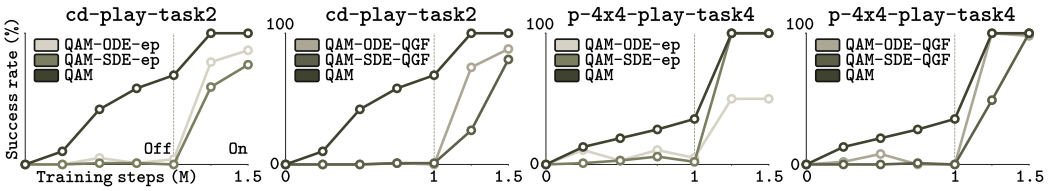


Figure 3: **Obs 1.** Adjoint propagation outperforms methods relying on terminal critic gradients.

Figure 3 compares the performance of QAM against the four variants, where QAM outperforms both *endpoint* and *QGF* variants. This indicates that adjoint-based gradient propagation is more effective than terminal critic gradients, motivating inference-efficient methods that leverage adjoints within non-iterative action sampling frameworks.

Observation 2: NQL improves performance with short-horizon policy execution, but naive combination with Q-chunking can degrade performance. In theory, Q-chunking (QC) (Li et al., 2026b) mitigates bias accumulation in standard TD learning, while NQL orthogonally leverages distributional returns. To evaluate how these theoretical advantages translate to empirical performance, we retain the policy learning mechanism of Flow Q-Learning (FQL) (Park et al., 2025b) and

compare six value-learning variants: *(i) FQL-TD(1)*, *(ii) FQL-TD(5)*, *(iii) FQL-QC(5)*, *(iv) NFQL-TD(1)*, *(v) NFQL-TD(5)*, and *(vi) NFQL-QC(5)*. The *TD(n)* variants apply standard TD learning, whereas the *QC(5)* variants condition critics on five-step action sequence outputs from the policy. The *NFQL* variants incorporate distributional returns following Eq.4.

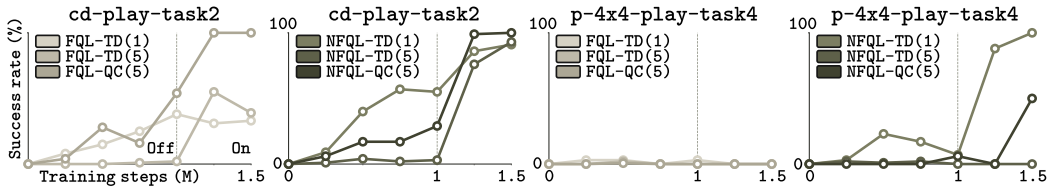


Figure 4: **Obs 2.** Average task performance improves with the use of distributional critics, but the combination of *NFQL* and *QC* does not always lead to better performance.

Figure 4 demonstrates how NQL and Q-chunking affects task performance. Comparing the first two and the latter two plots, we observe that NQL generally improves performance with FQL. However, focusing on the right-most plot, applying Q-chunking to NQL doesn't lead to better performance. In fact, for the `cube-double-play-task2` cases, *NFQL* only improves *TD* variants. Since the *TD* variants have shorter policy execution horizon than *QC*, this motivates how to design the algorithm that connects Q-chunking with both distributional critics and short-horizon policies.

5 Noise-conditioned Q-Adjoint Anchoring (NQA)

Section 4 highlights the possible benefits of adjoints, distributional returns, and short-horizon policy execution. To this end, we aim to integrate straight flow policies with these three key components. However, two challenges remain: first, directly applying Adjoint Matching to straight flows is non-trivial, and second, distributional RL remains unexplored for QAM. We propose **NQA** to address both challenges, using insights illustrated in Figure 5 for (1) policy learning and (2) value learning.

(1) Straight flow policies can directly make use of the adjoints. QAM introduces a variant called QAM-F that learns straight flow policies using adjoints. However, QAM-F *indirectly* applies Adjoint Matching to straight flows, since it applies Adjoint Matching to multi-step flow policies and then distills them. Direct application of Adjoint Matching requires adjoint propagation through straight flow trajectories, which becomes possible with our insight illustrated in Figure 5 *(Left)*. If we interpret the straight flow through the optimal controlled drift b^*_β and Brownian noise B_t, it can be viewed as a surrogate realization of the optimal controlled process. This motivates propagating adjoints along the straight trajectory to obtain information about the tilted distribution without iterative forward SDE integration. In Section 5.1, we describe **Q-Adjoint Anchoring** based on this insight, which combines Q-Adjoint Matching with Flow Anchoring (Eq.3).

(2) Distributional returns can be integrated with adjoints. Park et al. (2026b) highlighted that bias reduction in value learning is crucial for scalable RL performance. Furthermore, Li et al. (2026a) and Song et al. (2026) demonstrated that when mitigating critic bias, one should avoid executing long-horizon action sequences at every inference step. In Section 5.2, we introduce **Noise-conditioned Q-Chunking**, which extends distributional RL to accommodate short-horizon policy execution while mitigating bias accumulation via Q-chunking (Li et al., 2026b).

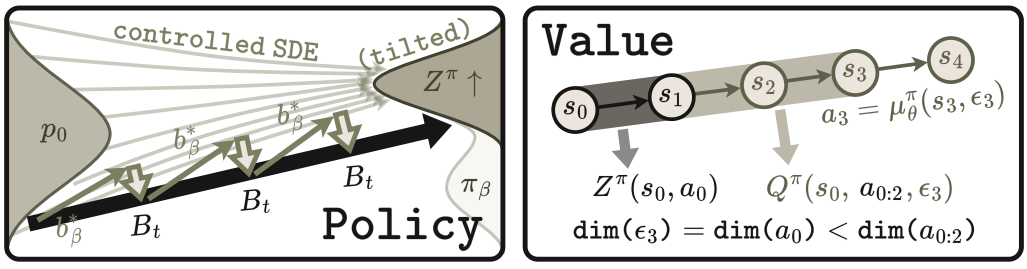


Figure 5: **Insights.** (*Left*) Straight flow trajectories can arise within the controlled SDE, calibrated by the Brownian motion term B_t. (*Right*) Short-horizon policy execution (a_0) can be retrained while incorporating Q-chunking ($a_{0:2}$) for distributional returns.

Algorithm 1: Noise-conditioned Q-Adjoint Anchoring (NQA)

Policy Parameters: β, θ;
Value Parameters: ϕ, ψ;
Hyperparameters: α, λ;

for $k = 1, \ldots, K$ **do**
  Sample an H-step batch $\mathcal{B}_H \sim \mathcal{D}$;
  $\mathcal{B}_H = \{(s_i, a_i, r_i, s_{i+1})\}_{i=0}^{H-1}$;
  `ValueUpdate(`$\mathcal{B}_H$`)`;
  `PolicyUpdate(`$\mathcal{B}_H$`)`;

Function `ValueUpdate(`$\mathcal{B}_H$`)`:
  Update Q_ϕ by noise-conditioned TD learning (Eq.9);
  Update Z_ψ by upper expectile regression (Eq.10);

Function `PolicyUpdate(`$\mathcal{B}_H$`)`:
  Update v_β by behavior flow matching (Eq.11);
  Estimate the tilted velocity v_β^* using α, λ (Eqs. 6–7);
  Update μ_θ^π by anchoring to v_β^* (Eq.8);

5.1 Policy Learning: Q-Adjoint Anchoring

To realize our first insight, we jointly train v_β and μ_θ^π, where v_β estimates the velocity field of the behavior policy $\pi_\beta \approx p_\mathcal{D}$, and μ_θ^π is a straight flow action sampler for the learning policy $\pi_\theta \approx \pi$.

Adjoint propagation along straight flows. Given $s \in \mathcal{S}$, $\epsilon \sim \mathcal{N}(0, I_m)$, and $t \sim \mathrm{Unif}([0,1])$, $a_\theta = \mu_\theta^\pi(s, \epsilon)$ defines the linear interpolation $x_{t,\theta} = (1-t)\epsilon + t a_\theta$. Using v_β trained with Eq.11, we obtain the adjoint $\tilde{a}_t$ by solving the following ODE backward in time:

$$\frac{d}{dt}\tilde{a}_t = -\left[\nabla_x \left(2 v_\beta(x_{t,\bar{\theta}}, t; s) - \frac{x_{t,\bar{\theta}}}{t}\right)\right]^\top \tilde{a}_t, \qquad \tilde{a}_1 = -\frac{1}{\alpha}\nabla_x Z_{\bar{\psi}}(s, a_{\bar{\theta}}), \tag{6}$$

where $\bar{\cdot}$ over θ and ψ indicates the $\mathrm{stopgrad}$ operation, and α is a temperature hyperparameter. The computational advantage over prior work arises because the intermediate states $x_{t,\bar{\theta}}$ can be constructed in *parallel*, without forward numerical integration. Given the resulting adjoints, we construct a sample-wise estimate of the tilted velocity field as:

$$\widehat{v}_\beta^*(x_{t,\bar{\theta}}, t; s) = v_\beta(x_{t,\bar{\theta}}, t; s) - \frac{\sigma(t)^2}{2}\tilde{a}_t, \qquad \sigma(t) = \sqrt{\frac{2(1-t)}{t}}, \tag{7}$$

where Appendix B.1 shows that the corresponding conditional average under the optimal controlled process recovers the exact tilted velocity field v_β^*. Here, we propagate the adjoints along μ_θ^π's straight interpolation as an efficient approximation. For stable training, we introduce a second and final hyperparameter, λ, that clips $\sigma(t)^2\tilde{a}_t/2$ to be within the range of $[-\lambda, \lambda]$. Intuitively, λ limits how far the estimated tilted velocity can deviate from the behavior velocity field v_β.

Adjoint Anchoring. Given $\widehat{v}_\beta^*$, we apply the flow anchoring objective from Eq.3:

$$\mathcal{L}_{\mathrm{AA}}(\theta) = \mathbb{E}_{\substack{s\sim\mathcal{D},\, \epsilon\sim\mathcal{N}(0,I_m),\\ t\sim\mathrm{Unif}([0,1])}}\left[\left\|(\mu_\theta^\pi(s,\epsilon) - \epsilon) - \widehat{v}_\beta^*\left((1-t)\epsilon + t\mu_\theta^\pi(s,\epsilon), t; s\right)\right\|_2^2\right]. \tag{8}$$

Optimizing for Eq.8 minimizes an upper bound of the W_2 distance between π_θ and the tilted distribution approximated with $\widehat{v}_\beta^*$. We provide the corresponding proof in Appendix B.2.

5.2 Value Learning: Noise-conditioned Q-Chunking

With our second insight, we jointly train Q_ϕ and Z_ψ. Please refer to Appendix B.3 for more details.

Noise-conditioned critic update. Let $M = Hd$, where $H \geq n$ is the length of the horizon that is potentially longer than the horizon of the policy output. Given $s \in \mathcal{S}$, $a_{0:H-1} \in \mathbb{R}^M$, and $\epsilon' \sim \mathcal{N}(0, I_m)$, the noise-conditioned critic Q_ϕ is learned through the following objective:

$$\mathcal{L}_Q(\phi) = \mathbb{E}\left[\left(Q_\phi(s_0, a_{0:H-1}, \epsilon') - \left(\sum_{i=0}^{H-1}\gamma^i r_i + \gamma^H Z_\psi\big(s_H, \mu_\theta^\pi(s_H, \epsilon')\big)\right)\right)^2\right]. \tag{9}$$

The expectation is over ϵ' and $\{(s_i, a_i, r_i, s_{i+1})\}_{i=0}^{H-1} \sim \mathcal{D}$, with ϵ' determining the actions at s_H. Q_ϕ uses an H-step backup to reduce bootstrapping frequency and mitigate bias accumulation.

Upper expectile regression. We then optimize the following objective:

$$\mathcal{L}_Z(\psi) = \mathbb{E}_{\substack{\epsilon\sim\mathcal{N}(0,I_m),\\ (s,a_{0:H-1})\sim\mathcal{D}}}\left[L_2^\kappa\left(Q_{\hat{\phi}}(s, a_{0:H-1}, \epsilon) - Z_\psi(s, a_{0:n-1})\right)\right], \tag{10}$$

where the input of Z_ψ only contains the first n-step actions of $a_{0:H-1} \sim \mathcal{D}$. With $\kappa \to 1^-$, this recovers Z_ψ to estimate $\operatorname{ess\,sup}_{\epsilon, a_{n:H-1}} Q_N^\pi(s, a_{0:H-1}, \epsilon)$, where Q_N^π is the fixed point of Eq.40.

6 EXPERIMENTS

We evaluate **NQA** through offline-to-online experiments in OGBench (Park et al., 2025a), which contains challenging long-horizon robotics locomotion and manipulation tasks.

Baseline. We organize state-of-the-art algorithms that applies to both offline and offline-to-online setups. The three most related algorithms are **QAM** (Li & Levine, 2026) applying Adjoint Matching, **DQC** (Li et al., 2026a) using chunked value decoupling, and **FAN** (Lee et al., 2026) with Flow Anchoring and Noise-conditioned Q-Learning. Next, **CGQ** (Song et al., 2026) builds upon DQC, and **ReBRAC** (Tarasov et al., 2023) and **FQL** (Park et al., 2025b) are simple yet effective inference-efficient methods. **IDQL** (Hansen-Estruch et al., 2023) and **DSRL** (Wagenmaker et al., 2025) also show high task success rates by post-processing the diffusion policies. For fair comparison, we extend all methods to deal with action chunks if not implemented in the original paper.

Setup. We tackle `navigate` environments for locomotion, and `play` for manipulation tasks. For offline training, we use 100M-sized datasets for `humanoidmaze-giant`, `cube-quadruple`, and `puzzle-4x6`, and default sizes for all other datasets. For all tasks, we target version `v0` of `singletask` cases, and the horizon length $H = 5$ is fixed using Q-chunking as default unless specified otherwise. We train 1M steps offline and then additionally train 0.5M steps online. Reported scores are averaged over the final three evaluation checkpoints: 0.9M, 0.95M, 1.0M steps for offline, and 1.4M, 1.45M, 1.5M steps for offline-to-online. Please refer to Appendix D for details.

6.1 MAIN RESULTS

As NQA has the highest inference efficiency, we focus on task performance and training efficiency.

Task performance. Table 1 reports the average task success rates of NQA and the baselines. After offline-to-online fine-tuning, NQA achieves an overall average success rate approximately $1.4\times$ that of the strongest baseline. NQA performs particularly well on the `humanoidmaze` environments.

Training compute efficiency. Figure 6 presents relative efficiency of each algorithm compared to NQA, where we measure runtime per training step using JAX and RTX 6000 ADA architecture. NQA training is more than $2.5\times$ faster than QAM training, even with the use of distributional critics.

Table 1: **Offline-to-online results.** We use 8 seeds with hyperparameters in Tables 6–7. Results are bolded if they are within the 95% range of the best value in each task, and are colored in green if both offline and online results are bolded. Please refer to the full results stated in Table 12.

	MULTI-STEP FLOW INFERENCE				SINGLE-STEP FLOW INFERENCE				
TASK ENVS	IDQL	DSRL	QAM	DQC	REBRAC	FQL	CGQ	FAN	NQA
ANTMAZE-LARGE	[2,9] [83,91] 6 →87	[4,10] [85,91] 7 →88	[5,10] [92,94] 7 →93	[6,12] [13,26] 9 →19	[63,71] [38,76] 67→57	[0,0] [42,88] 0 →66	[51,57] [97,100] 54→**99**	[4,4] [64,92] 4 →77	[84,89] [91,97] **86**→94
ANTMAZE-GIANT	[0,2] [17,20] 1 →19	[0,0] [44,74] 0 →56	[0,0] [71,79] 0 →75	[0,0] [0,0] 0 → 0	[24,30] [56,70] 27→62	[0,0] [5,64] 0 →32	[0,0] [74,86] 0 →81	[0,0] [22,61] 0 →43	[38,53] [78,91] **45→86**
ANTSOCCER-ARENA	[4,6] [8,9] 5 → 8	[12,13] [61,65] 13→62	[31,37] [69,75] 33→72	[10,16] [0,1] 13→ 1	[28,31] [9,13] 30→12	[0,0] [6,8] 0 → 7	[61,65] [83,87] **63→85**	[0,0] [63,68] 0 →65	[58,68] [85,91] **63→88**
HUMANOIDMAZE-MEDIUM	[20,34] [9,20] 27→14	[32,47] [53,71] 39→62	[8,14] [30,36] 10→33	[41,53] [12,17] 47→15	[41,58] [15,32] 50→21	[0,2] [25,30] 1 →27	[37,39] [38,40] 38→39	[0,0] [18,20] 0 →19	[88,96] [84,97] **93→92**
HUMANOIDMAZE-LARGE	[0,0] [0,0] 0 → 0	[0,0] [7,20] 0 →12	[0,0] [1,2] 0 → 1	[3,9] [0,2] 6 → 1	[2,2] [0,2] 2 → 1	[0,0] [0,0] 0 → 0	[0,0] [9,18] 0 →13	[0,0] [0,0] 0 → 0	[46,67] [62,83] **57→74**
HUMANOIDMAZE-GIANT	[9,10] [0,0] 9 → 0	[0,0] [0,0] 0 → 0	[0,2] [0,0] 1 → 0	[12,14] [0,8] 13→ 5	[3,8] [22,25] 6 →23	[0,0] [0,0] 0 → 0	[0,0] [7,19] 0 →13	[0,0] [0,0] 0 → 0	[35,53] [56,78] **44→68**
CUBE-DOUBLE	[9,15] [59,70] 12→65	[63,74] [96,99] **68→97**	[52,69] [99,100] 61→**99**	[6,13] [67,72] 9 →70	[4,6] [56,68] 5 →63	[26,40] [95,98] 34→**97**	[34,46] [82,89] 40→86	[32,51] [99,100] 42→**99**	[58,72] [96,99] **65→98**
CUBE-TRIPLE	[0,1] [0,0] 0 → 0	[2,8] [6,10] 5 → 8	[1,6] [29,37] 3 →33	[0,0] [0,0] 0 → 0	[2,4] [17,20] 3 →19	[0,0] [0,0] 0 → 0	[3,6] [41,51] 4 →46	[3,10] [87,92] 6 →**89**	[6,14] [79,89] **10**→84
CUBE-QUADRUPLE	[6,13] [55,58] **9** →56	[2,8] [77,80] 5 →**79**	[2,7] [76,79] 4 →**77**	[0,1] [0,0] 1 → 0	[1,5] [49,52] 2 →50	[0,12] [39,40] 6 →39	[0,1] [23,41] 0 →29	[0,0] [59,72] 0 →66	[5,12] [63,77] 8 →70
SCENE	[12,15] [0,4] 14→ 2	[98,99] [100,100] **99→100**	[55,56] [79,81] 56→80	[42,47] [51,54] 45→52	[32,42] [58,60] 37→59	[50,55] [66,78] 52→73	[31,36] [78,78] 34→78	[47,55] [60,79] 51→72	[93,98] [97,100] **96→98**
PUZZLE-4X4	[13,20] [44,57] 16→51	[20,33] [66,80] 26→73	[37,47] [100,100] **42→100**	[18,26] [90,100] 22→**96**	[2,6] [46,66] 4 →55	[2,7] [13,40] 5 →27	[11,20] [53,80] 15→67	[16,35] [33,60] 26→47	[36,46] [99,100] **41→100**
PUZZLE-4X5	[8,11] [19,20] 9 →**19**	[19,20] [20,20] **19→20**	[17,18] [20,20] 17→**20**	[18,20] [20,20] **19→20**	[14,17] [20,20] 16→**20**	[10,14] [20,20] 12→**20**	[10,17] [20,20] 13→**20**	[12,19] [20,20] 16→**20**	[20,20] [20,20] **20→20**
PUZZLE-4X6-100M	[1,10] [20,20] 5 →**20**	[19,20] [20,20] **19→20**	[9,10] [20,20] 10→**20**	[4,10] [20,20] 7 →**20**	[0,6] [20,20] 2 →**20**	[8,18] [20,20] 13→**20**	[0,0] [20,20] 0 →**20**	[0,14] [20,20] 5 →**20**	[6,10] [18,20] 8 →**19**
OVERALL	[5,13] [12,42] 9 →26	[9,40] [33,70] 23→52	[8,31] [35,73] 19→54	[7,23] [9,41] 15→23	[9,31] [24,47] 19→36	[2,19] [16,48] 9 →31	[9,32] [36,69] 20→52	[3,21] [30,65] 12→47	[33,65] [61,89] **49→76**

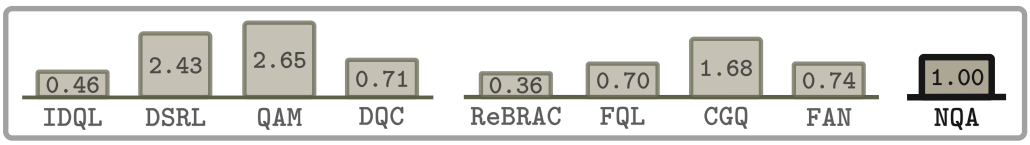


Figure 6: **Normalized training runtime** with respect to NQA, averaged over all tasks in Table 1.

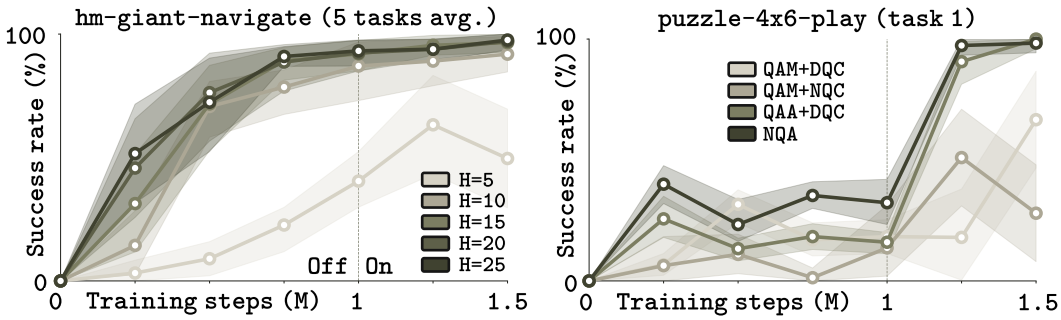


Figure 7: **Ablation studies.** *(Left)* Average success rates and sample efficiency improves as the horizon length H increases. *(Right)* Combination of NQA shows the best stability and success rates.

6.2 Ablation Studies

We conduct three ablation studies. First, we test how the success rates change when scaling up the horizon length H. Second, we check how **Q-Adjoint Anchoring (QAA)** and **Noise-conditioned Q-Chunking (NQC)** each contributes to the performance. Third, we analyze sensitivities on our main hyperparameters α and λ, the expectile coefficient κ, and the policy execution horizon n.

1. Scaling up the horizon length H. With `humanoidmaze-giant-navigate`, we evaluate how NQA performs on long-horizon sparse-reward tasks with different horizon length H. The left part of Figure 7 reveals the training curve averaged over 4 seeds, and we observe that the final offline and offline-to-online task success rates improve as H increases from 5 to 25. Moreover, the sample efficiency gets better. Please refer to the full result at Table 2 in Appendix C.1.

2. Why Q-Adjoint Anchoring (QAA) + Noise-conditioned Q-Chunking (NQC)? **NQA**'s policy and value training mechanisms can be applied *independently* to existing methods. Therefore, we conduct an ablation study on `puzzle-4x6-play-task1` comparing (1) QAM + DQC, (2) QAM + NQC, (3) QAA + DQC, and (4) NQA to isolate their respective effects. With 8 seeds and following Table 10, the right part of Figure 7 demonstrates that applying both QAA and NQC leads to best task performance. Our further analysis in Appendix C.2 provides additional evidence that NQC is itself powerful when attached to various straight flow policy learning approaches.

3. Sensitivity analysis. We provide sensitivity analyses for α and λ in Appendix C.3.1 and evaluate performance across different κ values in Appendix C.3.2. Moreover, analysis on the policy execution horizon n is provided in Appendix C.3.3. In summary, α should be selected carefully, whereas $\lambda = 1$ or 3 generally yields strong performance across all tasks. Furthermore, while $\kappa = 0.9$ serves as a robust default, tuning $\kappa \in [0.7, 0.9]$ can yield additional gains.

7 Conclusion

We propose **Noise-conditioned Q-Adjoint Anchoring (NQA)**, an off-policy reinforcement learning (RL) algorithm that solves challenging offline-to-online tasks with high computational efficiency. Since offline-to-online RL aims to solve tasks with minimal environment interactions, balancing training and inference efficiency with high task performance is crucial. By overcoming the limitations of diffusion/flow policies and distributional critics, NQA solves complex robotic control tasks that prior methods could not, while maintaining high efficiency during both training and inference. Notably, in long-horizon humanoid navigation, NQA's performance scales as the proposed distributional critic incorporates longer action sequence inputs. We believe this unlocks the scaling potential of off-policy RL to efficiently solve long-horizon and sparse-reward tasks.

There exist several promising directions for future work. First, scaling NQA to larger, more capable models (*e.g.*, Vision-Language-Action models) in real-world scenarios represents a vital step toward practical robotic deployment. Second, extending NQA to discrete action spaces could enhance large language models (LLMs) while improving RL training efficiency, given that current LLM post-training relies heavily on on-policy algorithms. We hope NQA serves as a foundational step toward scalable, highly efficient reinforcement learning for next-generation self-improving AI.

AI USE STATEMENT

In this work, large language models (LLMs) were used to assist in formulating mathematical claims and drafting proofs. Additionally, LLMs were used for recommended disclosure subtasks, including searching for and identifying related literature, editing manuscript text for grammatical clarity, and modifying non-core software code (*e.g.*, script refactoring, formatting, and minor debugging). LLMs were not used for any other required disclosure tasks, such as generating synthetic datasets, developing theoretical frameworks, proposing hypotheses, designing research methodology, implementing core algorithmic methods, cleaning datasets, or interpreting empirical results. All LLM-assisted outputs were thoroughly vetted by the authors: mathematical proofs were re-derived line-by-line, code modifications were tested and verified, retrieved literature was cross-referenced against primary sources, and prose edits were reviewed for accurate technical expression. The authors take full responsibility for all content, code, mathematical claims, and references in this submission.

ETHICS STATEMENT

This work presents foundational machine learning research and does not involve human subjects, sensitive personal data, or high-risk dual-use applications. The authors have evaluated the potential ethical implications of this work and anticipate no foreseeable negative societal impacts, safety risks, or ethical concerns beyond those standard to general computational research.

REPRODUCIBILITY STATEMENT

To ensure the full reproducibility of our results, all theoretical claims, underlying assumptions, and complete step-by-step proofs are provided in Appendix B. Full details regarding datasets, data preprocessing pipelines, baseline configurations, hyperparameter settings, and experimental setups are described in Section 6 and Appendix D. Furthermore, source code, model implementations, execution scripts, and environment dependencies required to reproduce all empirical findings are anonymously hosted at `https://anonymous.4open.science/r/NQA-ICLR26`.

REFERENCES

Armaan A Abraham, Lucy Xiaoyang Shi, and Chelsea Finn. Long-horizon q-learning: Accurate value learning via n-step inequalities. *arXiv preprint arXiv:2605.05812*, 2026.

Michael S. Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *Proceedings of the 11th International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=li7qeBbCR1t`.

Philip J Ball, Laura Smith, Ilya Kostrikov, and Sergey Levine. Efficient online reinforcement learning with offline data. In *International Conference on Machine Learning*, pp. 1577–1594. PMLR, 2023.

Marc G. Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 449–458. PMLR, 2017.

Marc G Bellemare, Will Dabney, and Mark Rowland. *Distributional reinforcement learning*. MIT Press, 2023.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

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. `https://github.com/jax-ml/jax`, 2018.

Huayu Chen, Cheng Lu, Zhengyi Wang, Hang Su, and Jun Zhu. Score regularized policy optimization through diffusion behavior. In *Proceedings of the 12th International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=xCRr9DrolJ.

Tianyu Chen, Zhendong Wang, and Mingyuan Zhou. Diffusion policies creating a trust region for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 37, pp. 50098–50125, 2024b.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

Taehyun Cho, Seungyub Han, Heesoo Lee, Kyungjae Lee, and Jungwoo Lee. Pitfall of optimism: Distributional reinforcement learning by randomizing risk criterion. *Advances in Neural Information Processing Systems*, 36:56802–56824, 2023.

Taehyun Cho, Seungyub Han, Seokhun Ju, Dohyeong Kim, Kyungjae Lee, and Jungwoo Lee. Bellman unbiasedness: Toward provably efficient distributional reinforcement learning with general value function approximation. In *ICML*, 2025.

Hojun Chung, Junseo Lee, and Songhwai Oh. Offline reinforcement learning with universal horizon models. In *International Conference on Machine Learning*, 2026.

Will Dabney, Georg Ostrovski, David Silver, and Rémi Munos. Implicit quantile networks for distributional reinforcement learning. In *International Conference on Machine Learning*, volume 80, pp. 1096–1105. PMLR, 2018a.

Will Dabney, Mark Rowland, Marc G Bellemare, and Rémi Munos. Distributional reinforcement learning with quantile regression. In *AAAI Conference on Artificial Intelligence*, volume 32, pp. 2892–2901, 2018b.

Zihan Ding and Chi Jin. Consistency models as a rich and efficient policy class for reinforcement learning. In *International Conference on Learning Representations*, volume 2024, pp. 53047–53066, 2024.

Carles Domingo i Enrich, Michal Drozdzal, Brian Karrer, and Ricky TQ Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. In *International Conference on Learning Representations*, volume 2025, pp. 53791–53846, 2025.

Perry Dong, Kuo-Han Hung, Tian Gao, Dorsa Sadigh, and Chelsea Finn. Expo-ft: Sample-efficient reinforcement learning finetuning for vision-language-action models. *arXiv preprint arXiv:2605.25477*, 2026a.

Perry Dong, Qiyang Li, Dorsa Sadigh, and Chelsea Finn. Expo: Stable reinforcement learning with expressive policies. In *International Conference on Learning Representations*, volume 2026, pp. 131379–131394, 2026b.

Perry Dong, Chongyi Zheng, Chelsea Finn, Dorsa Sadigh, and Benjamin Eysenbach. Value flows. In *International Conference on Learning Representations*, volume 2026, pp. 128614–128646, 2026c.

Yonghoon Dong, Kyungmin Lee, Changyeon Kim, Jaehyuk Kim, and Jinwoo Shin. Trust region q adjoint matching. *arXiv preprint arXiv:2605.27079*, 2026d.

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

Bradley Efron. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011. doi: 10.1198/jasa.2011.tm11181.

Nicolas Espinosa-Dice, Yiyi Zhang, Yiding Chen, Bradley Guo, Owen Oertell, Gokul Swamy, Kiante Brantley, and Wen Sun. Scaling offline rl via efficient and expressive shortcut models. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Linjiajie Fang, Ruoxue Liu, Jing Zhang, Wenjia Wang, and Bingyi Jing. Diffusion actor-critic: Formulating constrained policy iteration as diffusion noise regression for offline reinforcement learning. In *International Conference on Learning Representations*, 2025.

William Fedus, Prajit Ramachandran, Rishabh Agarwal, Yoshua Bengio, Hugo Larochelle, Mark Rowland, and Will Dabney. Revisiting fundamentals of experience replay. In *International conference on machine learning*, pp. 3061–3071. PMLR, 2020.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 34:20132–20145, 2021.

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

Chen-Xiao Gao, Chenyang Wu, Mingjun Cao, Chenjun Xiao, Yang Yu, and Zongzhang Zhang. Behavior-regularized diffusion policy optimization for offline reinforcement learning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 18630–18657. PMLR, 2025. URL `https://proceedings.mlr.press/v267/gao25q.html`.

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.

Longxiang He, Li Shen, Linrui Zhang, Junbo Tan, and Xueqian Wang. Diffcps: Diffusion model based constrained policy search for offline reinforcement learning. *arXiv preprint arXiv:2310.05333*, 2023.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016. URL `https://arxiv.org/abs/1606.08415`.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

Korrawe Karunratanakul, Konpat Preechakul, Emre Aksan, Thabo Beeler, Supasorn Suwajanakorn, and Siyu Tang. Optimizing diffusion noise can serve as universal motion priors. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1334–1345. IEEE, 2024.

Changyeon Kim, Haeone Lee, Younggyo Seo, Kimin Lee, and Yuke Zhu. Deas: Detached value learning with action sequence for scalable offline rl. In *International Conference on Learning Representations*, 2026.

Dohyeong Kim, Kyungjae Lee, and Songhwai Oh. Trust region-based safe distributional reinforcement learning for multiple constraints. In *Advances in Neural Information Processing Systems*, volume 36, pp. 19908–19939, 2023.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, 2015. URL `https://openreview.net/forum?id=8gmWwjFyLj`.

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

Lucy Lai, Ann Zixiang Huang, and Samuel J Gershman. Action chunking as policy compression. *PsyArXiv*, 2022.

Dongsu Lee and Minhae Kwon. Scenario-free autonomous driving with multi-task offline-to-online reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, 26(9):13317–13330, 2025.

Sungyoung Lee, Dohyeong Kim, Eshan Balachandar, Zelal Su Mustafaoglu, and Keshav Pingali. Towards efficient and expressive offline rl via flow-anchored noise-conditioned q-learning. *arXiv preprint arXiv:2605.01663*, 2026.

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 and Sergey Levine. Q-learning with adjoint matching. *arXiv preprint arXiv:2601.14234*, 2026.

Qiyang Li, Seohong Park, and Sergey Levine. Decoupled q-chunking. In *International Conference on Learning Representations*, volume 2026, pp. 145315–145389, 2026a.

Qiyang Li, Zhiyuan Paul Zhou, and Sergey Levine. Reinforcement learning with action chunking. *Advances in Neural Information Processing Systems*, 38:55518–55553, 2026b.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *Proceedings of the 11th International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=PqvMRDCJT9t`.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *Proceedings of the 11th International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=XVjTT1nw5z`.

Lei Lyu, Yunfei Li, Yu Luo, Fuchun Sun, Tao Kong, Jiafeng Xu, and Xiao Ma. Flow-based policy for online reinforcement learning. *Advances in Neural Information Processing Systems*, 38:93967–93990, 2026.

Haitong Ma, Tianyi Chen, Kai Wang, Na Li, and Bo Dai. Efficient online reinforcement learning for diffusion policy. In *International Conference on Machine Learning*, volume 267, pp. 41837–41853. PMLR, 2025a.

Xiaoteng Ma, Junyao Chen, Li Xia, Jun Yang, Qianchuan Zhao, and Zhengyuan Zhou. DSAC: Distributional soft actor-critic for risk-sensitive reinforcement learning. *Journal of Artificial Intelligence Research*, 83, 2025b. doi: 10.1613/jair.1.17526.

Yecheng Jason Ma, Dinesh Jayaraman, and Osbert Bastani. Conservative offline distributional reinforcement learning. *Advances in Neural Information Processing Systems*, 34:19235–19247, 2021.

Bogdan Mazoure, Thang Doan, Audrey Durand, Joelle Pineau, and R Devon Hjelm. Leveraging exploration in off-policy algorithms via normalizing flows. In *Conference on Robot Learning*, pp. 430–444. PMLR, 2020.

David McAllister, Songwei Ge, Brent Yi, Chung Min Kim, Ethan Weber, Hongsuk Choi, Haiwen Feng, and Angjoo Kanazawa. Flow matching policy gradients. In *International Conference on Learning Representations*, volume 2026, pp. 36352–36372, 2026.

Tetsuro Morimura, Masashi Sugiyama, Hisashi Kashima, Hirotaka Hachiya, and Toshiyuki Tanaka. Nonparametric return distribution approximation for reinforcement learning. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 799–806, 2010.

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

Mitsuhiko Nakamoto, Simon Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn, Aviral Kumar, and Sergey Levine. Cal-ql: Calibrated offline rl pre-training for efficient online fine-tuning. *Advances in Neural Information Processing Systems*, 36:62244–62269, 2023.

Whitney K Newey and James L Powell. Asymmetric least squares estimation and testing. *Econometrica: Journal of the Econometric Society*, pp. 819–847, 1987.

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

Aditya Oberai, Seohong Park, and Sergey Levine. Reversal q-learning. *arXiv preprint arXiv:2606.17551*, 2026.

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

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In *International Conference on Learning Representations*, volume 2025, pp. 94937–94982, 2025a.

Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning. *arXiv preprint arXiv:2502.02538*, 2025b.

Seohong Park, Kevin Frans, Deepinder Mann, Benjamin Eysenbach, Aviral Kumar, and Sergey Levine. Horizon reduction makes rl scalable. *Advances in Neural Information Processing Systems*, 38:8350–8389, 2026b.

Physical Intelligence, Kevin Black, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Allen Ren, Justin Lidard, Lars Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. In *International Conference on Learning Representations*, volume 2025, pp. 77288–77329, 2025.

Hannes Risken. *The Fokker–Planck Equation: Methods of Solution and Applications*. Springer, Berlin, Heidelberg, 2 edition, 1989.

Mark Rowland, Robert Dadashi, Saurabh Kumar, Rémi Munos, Marc G Bellemare, and Will Dabney. Statistics and samples in distributional reinforcement learning. In *International Conference on Machine Learning*, volume 97, pp. 5528–5536. PMLR, 2019.

Junwon Seo, Sushant Veer, Ran Tian, Wenhao Ding, Apoorva Sharma, Karen Leung, Edward Schmerling, Marco Pavone, and Andrea Bajcsy. Stressdream: Steering video world models for robust policy evaluation and improvement. *arXiv preprint arXiv:2606.00267*, 2026.

Younggyo Seo and Pieter Abbeel. Coarse-to-fine q-network with action sequence for data-efficient reinforcement learning. *Advances in Neural Information Processing Systems*, 38:109915–109940, 2026.

Gwanwoo Song, Kwanyoung Park, and Youngwoon Lee. Chunk-guided q-learning. *arXiv preprint arXiv:2603.13971*, 2026.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Richard S Sutton, Andrew G Barto, and Andrew Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

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.

Cédric Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2009.

Andrew Wagenmaker, Mitsuhiko Nakamoto, Yunchu Zhang, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space reinforcement learning. *arXiv preprint arXiv:2506.15799*, 2025.

Kaiwen Wang, Kevin Zhou, Runzhe Wu, Nathan Kallus, and Wen Sun. The benefits of being distributional: Small-loss bounds for reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 2275–2312, 2023a. doi: 10.52202/075280-0107.

Kaiwen Wang, Owen Oertell, Alekh Agarwal, Nathan Kallus, and Wen Sun. More benefits of being distributional: Second-order bounds for reinforcement learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 51192–51213. PMLR, 2024a.

Yinuo Wang, Likun Wang, Yuxuan Jiang, Wenjun Zou, Tong Liu, Xujie Song, Wenxuan Wang, Liming Xiao, Jiang Wu, Jingliang Duan, et al. Diffusion actor-critic with entropy regulator. *Advances in Neural Information Processing Systems*, 37:54183–54204, 2024b.

Zeyuan Wang, Da Li, Yulin Chen, Ye Shi, Liang Bai, Tianyuan Yu, and Yanwei Fu. One-step generative policies with q-learning: A reformulation of meanflow. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 26751–26759, 2026. doi: 10.1609/aaai.v40i31.39885.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *International Conference on Learning Representations*, 2023b.

Zhendong Wang, Max Li, Ajay Mandlekar, Zhenjia Xu, Jiaojiao Fan, Yashraj Narang, Linxi Fan, Yuke Zhu, Yogesh Balaji, Mingyuan Zhou, Ming-Yu Liu, and Yu Zeng. One-step diffusion policy: Fast visuomotor policies via diffusion distillation. In *International Conference on Machine Learning*, volume 267, pp. 63399–63416. PMLR, 2025.

Christopher John Cornish Hellaby Watkins. *Learning from delayed rewards*. PhD thesis, 2026.

Long Yang, Zhixiong Huang, Fenghao Lei, Yucun Zhong, Yiming Yang, Cong Fang, Shiting Wen, Binbin Zhou, and Zhouchen Lin. Policy representation via diffusion probability model for reinforcement learning. *arXiv preprint arXiv:2305.13122*, 2023.

Ruoqi Zhang, Ziwei Luo, Jens Sjölund, Thomas B Schön, and Per Mattsson. Entropy-regularized diffusion policy with q-ensembles for offline reinforcement learning. *Advances in neural information processing systems*, 37:98871–98897, 2024.

Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

Zhiyuan Zhou, Andy Peng, Charles Xu, Qiyang Li, Jost Tobias Springenberg, Kevin Frans, and Sergey Levine. Test-time gradient guidance of flow policies in reinforcement learning. *arXiv preprint arXiv:2606.11087*, 2026.

A Extended Preliminaries

Flow Matching. The behavior policy π_β can be learned through flow matching (Lipman et al., 2023; Liu et al., 2023; Albergo & Vanden-Eijnden, 2023). Given a state s, flow matching learns a velocity field $v_\beta(x, t; s) : \mathbb{R}^m \times [0, 1] \to \mathbb{R}^m$ that defines a flow transporting the prior distribution p_0 towards the target distribution $p_{\mathcal{D}}(\cdot|s)$. The corresponding flow map $\psi_\beta(x, t; s) : \mathbb{R}^m \times [0, 1] \to \mathbb{R}^m$ is defined by the ordinary differential equation (ODE): $\frac{d}{dt}\psi_\beta(x, t; s) = v_\beta\left(\psi_\beta(x, t; s), t; s\right)$, with $\psi_\beta(x, 0; s) = x$. The flow policy is then defined as $\pi_\beta(\cdot|s) = [\psi_\beta(\cdot, 1; s)]_{\#} p_0 \approx p_{\mathcal{D}}(\cdot|s)$, where $\#$ denotes the pushforward operator. The Optimal Transport (OT) (Villani et al., 2009) path is typically used, providing $x_t = (1 - t)x_0 + tx_1$ with $x_0 \sim \mathcal{N}(0, I_m)$, $t \sim \mathrm{Unif}([0, 1])$, and $(s, x_1) \sim p_{\mathcal{D}}$.

Rectified flow (Liu et al., 2023) and flow matching (Lipman et al., 2023) learn a velocity field v_β whose induced probability path matches the marginal distributions of the interpolation x_t, written as $p_t(\cdot|s)$. In other words, $\psi_\beta(x_0, t; s)$ follows $p_t(\cdot|s)$ when v_β is optimized for the following loss:

$$\mathcal{L}_{\text{FM}}(\beta) = \mathbb{E}_{\substack{(s,x_1)\sim p_{\mathcal{D}}, x_0\sim\mathcal{N}(0,I_m),\\ x_t=(1-t)x_0+tx_1,\ t\sim\mathrm{Unif}([0,1])}} \left[\left\|v_\beta(x_t, t; s) - (x_1 - x_0)\right\|_2^2\right]. \tag{11}$$

Actions following the dataset distribution $p_{\mathcal{D}}$ can then be sampled by numerically integrating the flow ODE defined by v_β. However, in this work, we do not sample dataset actions with v_β, but instead use v_β to construct the ODE for the "tilted" distribution introduced in the next paragraph.

Adjoint Matching. The goal of Adjoint Matching (Domingo i Enrich et al., 2025) is to learn an exponentially tilted distribution: $\tilde{p}_{\mathcal{D}}(A|s) \propto p_{\mathcal{D}}(A|s) \exp(Q(s, A)/\alpha)$. For a given state s and temperature coefficient α, the action random variable A is weighted by the function Q, where actions with larger Q values receive exponentially larger probability density. Setting $Q = Q^\pi$ increases the relative probability of actions with higher expected returns under policy π.

The tilted distribution can be viewed as a solution to the stochastic optimal control (SOC) problem:

$$\begin{aligned} \min_u \quad & \mathbb{E}\left[\frac{1}{2}\int_0^1 \|u(X_t^u, t; s)\|_2^2 \, dt - Q(s, X_1^u)\right] \\ \text{s.t.} \quad & dX_t^u = \left[b_\beta(X_t^u, t; s) + \sigma(t)u(X_t^u, t; s)\right] dt + \sigma(t) dB_t, \qquad X_0^u \sim p_0, \end{aligned} \tag{12}$$

where the expectation is taken over SDE trajectories. b_β is the SDE base drift and $\sigma(t)$ is the time-dependent diffusion coefficient of the SDE, both of which are fixed for a given SOC problem. X_t^u is the state of the stochastic process, B_t is Brownian motion, and u is the control function to optimize.

To obtain the optimal control solution u of Eq.1, one can directly solve the corresponding Hamilton–Jacobi–Bellman (HJB) equation. However, as this is computationally expensive, Domingo i Enrich et al. introduced the adjoint state $\tilde{a}_t$, which enables the following efficient regression objective that enforces the first-order optimality condition of the HJB:

$$\mathcal{L}_{\text{AM}}(u) = \mathbb{E}\left[\int_0^1 \frac{1}{2}\left\|u(X_t^{\bar{u}}, t; s) + \sigma(t)^\top \tilde{a}_t\right\|_2^2 dt\right], \qquad \bar{u} = \mathrm{stopgrad}(u). \tag{13}$$

Here, $\tilde{a}_t$ is obtained through the ODE: $\frac{d}{dt}\tilde{a}_t = -\left[\nabla_x b_\beta(X_t^{\bar{u}}, t; s)\right]^\top \tilde{a}_t$, with $\tilde{a}_1 = -\nabla_x Q(s, X_1^{\bar{u}})$. Notably, $\sigma(t)$ must be a memoryless noise schedule such that X_1^u becomes independent of X_0^u. Otherwise, the minimizer of Eq.2 yields a biased tilted distribution.

Adjoint Matching adopts the memoryless noise schedule $\sigma(t) = \sqrt{2(1 - t)/t}$ and the uncontrolled SDE trajectory $X_t = (1 - t)X_0 + tX_1$. This simplifies the base drift into $b_\beta(x, t; s) = v_\beta(x, t; s) + \sigma(t)^2 \nabla_x \log p_t(x|s)/2 = 2v_\beta(x, t; s) - x/t$, where the first equality comes from the Fokker–Planck equation (Risken, 1989), and the second comes from Tweedie's identity (Efron, 2011). Standard Adjoint Matching then first generates the controlled trajectory $X_t^{\bar{u}}$ by forward SDE integration. Next, it performs vector–Jacobian products (VJPs) backward in time to numerically solve the ODE of $\tilde{a}_t$. In this work, we embrace the core mechanics of these forward and backward processes, but replace forward numerical integration with a single-step flow sampling.

Flow Anchoring. Lee et al. (2026) recently proposed Flow Anchoring to enable efficient training of inference-efficient generative policies. The approach is motivated by the distillation loss proposed

in FQL (Park et al., 2025b), minimizing the Wasserstein-2 (W_2) distance between π_β and π_θ:

$$\mathcal{L}_{\text{Distill}}(\theta) = \mathbb{E}_{s\sim\mathcal{D}, z\sim p_0}\left[\left\|\mu^\pi_\theta(s,z) - \mu^\pi_\beta(s,z)\right\|^2_2\right]. \tag{14}$$

Here, μ^π_θ and μ^π_β are action samplers for the learning policy π_θ and the flow policy π_β, using the same noise z. However, obtaining μ^π_β in Eq.14 requires numerical iterations of the ODE. To this end, Flow Anchoring bypasses these iterations by optimizing for the following objective:

$$\mathcal{L}_{\text{FA}}(\theta) = \mathbb{E}_{s\sim\mathcal{D}, z\sim p_0, t\sim\mathcal{U}(0,1)}\left[\left\|\left(\mu^\pi_\theta(s,z) - z\right) - v_\beta\big((1-t)\,z + t\,\bar{\mu}^\pi_\theta(s,z), t; s\big)\right\|^2_2\right], \tag{15}$$

where the objective directly uses v_β instead of μ^π_β with $\bar{\mu}^\pi_\theta = \text{stopgrad}(\mu^\pi_\theta)$. Minimizing Eq.3 is known to minimize an upper bound on the Wasserstein-2 (W_2) distance between π_β and π_θ. NQA adopts this efficient training mechanism, but anchors μ^π_θ to the flow of the tilted distribution.

Distributional RL. Distributional RL (Bellemare et al., 2023) aims to capture rich return information beyond expected values. The return random variable is defined as $Q^\pi_D = \sum_{t=0}^\infty \gamma^t r(S_t, A_t)$, taking values in $[r_{\min}/(1-\gamma), r_{\max}/(1-\gamma)]$. S_t and A_t are the state and action random variables at time t, determined by trajectories following policy π and transition kernel P. The conditional return random variable is then defined as $Q^\pi_D(s,a) = r(s,a) + \sum_{h=1}^\infty \gamma^h r(S_h, A_h)$, which shares the same distribution as $r(s,a) + \gamma Q^\pi_D(S', A')$, where $S' \sim P(\cdot|s,a)$ and $A' \sim \pi(\cdot|S')$. Among various distributional RL approaches, our work builds upon Noise-conditioned Q-Learning (NQL).

Noise-conditioned Q-Learning. Lee et al. (2026) recently noted that tracking multiple distributional critic samples is inefficient since the optimal value changes during policy improvement. To this end, they proposed NQL, which mainly tracks the optimistic part of the return distribution:

$$\begin{aligned}
\mathcal{L}_Q(\phi) &= \mathbb{E}_{\substack{(s,a,r,s')\sim\mathcal{D},\\ \epsilon'\sim\mathcal{N}(0,I_d)}}\left[\left(Q_\phi(s,a,\epsilon') - (r + \gamma\, Z_\psi(s', \mu^\pi_\theta(s',\epsilon')))\right)^2\right], \\
\mathcal{L}_Z(\psi) &= \mathbb{E}_{\substack{(s,a)\sim\mathcal{D},\\ \epsilon\sim\mathcal{N}(0,I_d)}}\left[L^\kappa_2\left(Q_{\hat{\phi}}(s,a,\epsilon) - Z_\psi(s,a)\right)\right].
\end{aligned} \tag{16}$$

The expectile loss $L^\kappa_2(\hat{x} - x) = |\kappa - \mathbf{1}((\hat{x} - x) < 0)|, (\hat{x} - x)^2$ lets x capture the κ-expectile of the distribution of $\hat{x}$ (Newey & Powell, 1987; Kostrikov et al., 2021). Here, ϕ parameterizes the noise-conditioned distributional critic, while $\hat{\phi}$ denotes the corresponding target-network parameters. As $\kappa \to 1^-$, $Z_\psi(s,a)$ approaches $\operatorname{ess\,sup}_\epsilon Q^\pi_n(s,a,\epsilon)$, where Q^π_n denotes the fixed point of the following Bellman operator:

$$\mathcal{T}^\pi_n Q(s,a,\epsilon') = r(s,a) + \gamma\, \mathbb{E}_{s'\sim P(\cdot|s,a)}\left[\operatorname*{ess\,sup}_{\epsilon\sim\mathcal{N}(0,I_d)} Q(s', \mu^\pi(s',\epsilon'), \epsilon)\right]. \tag{17}$$

Here, the noise ϵ' parameterizes the stochasticity of π, and μ^π is the map transporting the distribution of ϵ' to the target policy $\pi(\cdot|s')$. Following Lee et al. (2026), we fix $\kappa = 0.9$ across all experiments, and compute the adjoints with respect to Z_ψ while incorporating the Q-chunking formulation.

B Theoretical Guarantees

B.1 Tilted Distribution Velocity Field v^*_β

Fix $s \in \mathcal{S}$ and $\alpha > 0$. Recall that Adjoint Matching considers the memoryless SDE process

$$dX^u_t = [b_\beta(X^u_t, t; s) + \sigma(t)u(X^u_t, t; s)]\, dt + \sigma(t)dB_t, \qquad 0 < t < 1, \tag{18}$$

where B_t is standard m-dimensional Brownian motion, $\sigma(t) > 0$ is scalar, u is the control vector field, and b_β is the drift associated with the exact population behavior velocity field v_β. For the linear interpolation $x_t = (1-t)\epsilon + ta$, with independent $\epsilon \sim \mathcal{N}(0, I_m)$ and $a \sim p_\beta(\cdot \mid s)$, the corresponding noise schedule and behavior drift are

$$\sigma(t)^2 = \frac{2(1-t)}{t}, \qquad b_\beta(x, t; s) = 2v_\beta(x, t; s) - \frac{x}{t}, \qquad 0 < t < 1. \tag{19}$$

Since $\sigma(t)$ is singular at zero, we use the SDE only for $t > 0$ and assume that the distribution of X^u_t converges to $\mathcal{N}(0, I_m)$ as $t \to 0^+$. This is a limit of marginal distributions, not an ordinary Itô initial condition at zero. Integrals starting at zero are understood as limits of integrals starting at a positive time.

For a frozen control $\bar{u} = \mathrm{stopgrad}(u)$, let $\mathbb{E}^{\bar{u}}_s$ denote expectation under the trajectory law of $X^{\bar{u}}$. The corresponding lean adjoint with the value function Z satisfies

$$\frac{d}{dt}\tilde{a}_t = -\left[\nabla_x b_\beta(X^{\bar{u}}_t, t; s)\right]^\top \tilde{a}_t, \qquad \tilde{a}_1 = -\frac{1}{\alpha}\nabla_x Z(s, X^{\bar{u}}_1). \tag{20}$$

Assume that $Z(s, \cdot)$ and $b_\beta(\cdot, t; s)$ are continuously differentiable, that the processes, adjoints, and conditional expectations below are well-defined, and that $\mathbb{E}^{\bar{u}}_s \int_0^1 \sigma(t)^2 \|\tilde{a}_t\|_2^2 dt < \infty$ for each frozen control considered. Adjoint Matching minimizes the unrestricted population regression objective

$$\mathcal{L}_{\mathrm{AM}}(u; \bar{u}) = \mathbb{E}^{\bar{u}}_s \left[\int_0^1 \frac{1}{2} \left\| u(X^{\bar{u}}_t, t; s) + \sigma(t)\tilde{a}_t \right\|_2^2 dt \right]. \tag{21}$$

Lemma B.1 (Conditional minimizer of Adjoint Matching)**.** Fix $s \in \mathcal{S}$ and a frozen control $\bar{u}$. For almost every $t \in (0, 1)$ and almost every x under the marginal law of $X^{\bar{u}}_t$,

$$\begin{aligned} &\arg\min_{q \in \mathbb{R}^m} \mathbb{E}^{\bar{u}}_s \left[\frac{1}{2} \|q + \sigma(t)\tilde{a}_t\|_2^2 \,\middle|\, X^{\bar{u}}_t = x \right] \\ &\qquad = -\sigma(t)\mathbb{E}^{\bar{u}}_s \left[\tilde{a}_t \,\middle|\, X^{\bar{u}}_t = x \right]. \end{aligned} \tag{22}$$

Consequently, if u^* is an unrestricted population fixed point of Adjoint Matching, then, with the same almost-everywhere qualification,

$$u^*(x, t; s) = -\sigma(t)\mathbb{E}^*_s \left[\tilde{a}_t \,|\, X^*_t = x \right], \tag{23}$$

where $\mathbb{E}^*_s := \mathbb{E}^{u^*}_s$.

Proof. With $\bar{u}$ frozen, the trajectory law and adjoint target are fixed. For almost every (t, x), the conditional squared-error objective is strictly convex in q, and its gradient is

$$q + \sigma(t)\mathbb{E}^{\bar{u}}_s[\tilde{a}_t \mid X^{\bar{u}}_t = x].$$

Setting this gradient to zero proves Eq.22. At a population fixed point, $\bar{u} = u^*$, which gives Eq.23. □

Theorem B.2 (Conditional recovery of the tilted velocity field). Fix $s \in \mathcal{S}$, $\alpha > 0$, and Z, and assume $0 < \mathbb{E}_{a\sim p_\beta(\cdot|s)}[e^{Z(s,a)/\alpha}] < \infty$. Let $p^*_\beta(\cdot \mid s)$ be the distribution obtained by reweighting $p_\beta(\cdot \mid s)$ by $e^{Z(s,a)/\alpha}$ and normalizing. Suppose u^* is a population fixed point of Adjoint Matching that is optimal for the memoryless control problem with running cost $\|u\|_2^2/2$ and terminal cost $-Z(s,\cdot)/\alpha$. Let X^*_t and $\tilde{a}_t$ be its controlled process and lean adjoint. Assume the exact memoryless correspondence: X^*_1 has distribution $p^*_\beta(\cdot \mid s)$, and its drift and associated velocity satisfy $b^*_\beta(x,t;s) = 2v^*_\beta(x,t;s) - x/t$. Then, for almost every $t \in (0,1)$ and almost every x under the distribution of X^*_t,

$$\mathbb{E}^*_s\left[v_\beta(X^*_t,t;s) - \frac{\sigma(t)^2}{2}\tilde{a}_t \,\middle|\, X^*_t = x\right] = v^*_\beta(x,t;s). \tag{24}$$

Proof. By Lemma B.1, the optimal control satisfies Eq.23. Therefore, the drift of the optimal controlled process is

$$\begin{aligned} b^*_\beta(x,t;s) &= b_\beta(x,t;s) + \sigma(t)u^*(x,t;s) \\ &= b_\beta(x,t;s) - \sigma(t)^2\mathbb{E}^*_s\left[\tilde{a}_t \,|\, X^*_t = x\right]. \end{aligned} \tag{25}$$

By the assumed exact memoryless correspondence,

$$b^*_\beta(x,t;s) = 2v^*_\beta(x,t;s) - \frac{x}{t}. \tag{26}$$

Subtracting Eq.19 from Eq.26 and using Eq.25 gives

$$v^*_\beta(x,t;s) = v_\beta(x,t;s) - \frac{\sigma(t)^2}{2}\mathbb{E}^*_s\left[\tilde{a}_t \,|\, X^*_t = x\right]. \tag{27}$$

Conditioning on $X^*_t = x$ fixes $v_\beta(X^*_t,t;s)$ at $v_\beta(x,t;s)$. Thus Eq.27 is exactly Eq.24. □

B.2 Q-Adjoint Anchoring

Independently sample $s \sim \mathcal{D}$, $\epsilon \sim \mathcal{N}(0, I_m)$, and $t \sim \mathrm{Unif}([0,1])$. Define

$$x_{t,\theta} = (1-t)\epsilon + t\mu_\theta^\pi(s,\epsilon), \qquad d_\theta = \mu_\theta^\pi(s,\epsilon) - \epsilon.$$

Thus,

$$x_{0,\theta} = \epsilon, \qquad x_{1,\theta} = \mu_\theta^\pi(s,\epsilon), \qquad \frac{d}{dt}x_{t,\theta} = d_\theta.$$

The actor distribution is defined by sampling:

$$\mu_\theta^\pi(s,\epsilon) \sim \pi_\theta(\cdot \mid s), \qquad \epsilon \sim \mathcal{N}(0, I_m).$$

Let v_β^* denote the tilted velocity field characterized in Theorem B.2. Define the ideal Q-Adjoint Anchoring objective using this exact velocity as

$$\mathcal{L}_{\mathrm{AA}}(\theta) := \mathbb{E}_{s,\epsilon,t}\left[\left\|d_\theta - v_\beta^*(x_{t,\theta}, t; s)\right\|_2^2\right]. \tag{28}$$

The stop-gradient copy $x_{t,\bar{\theta}}$ has the same forward value as $x_{t,\theta}$, so it leaves this loss value unchanged, but changes its gradient. The result below uses the ideal loss; it does not prove convergence of stop-gradient updates. Also, Theorem B.2 takes a conditional mean under the optimal SDE law, not an adjoint from an arbitrary actor-generated path.

Assumption B.3 (Regularity of the tilted flow)**.** Assume the actor and v_β^* are jointly measurable. For each $s \in \mathcal{S}$, the ODE

$$\frac{d}{dt}X_t = v_\beta^*(X_t, t; s), \qquad X_0 \sim \mathcal{N}(0, I_m), \tag{29}$$

admits a unique absolutely continuous solution on $[0,1]$ for every initial point. With the stated Gaussian initialization, $X_1 \sim p_\beta^*(\cdot \mid s)$. Moreover, there exists $L \geq 0$, uniformly over (s,t), such that

$$\left\|v_\beta^*(x,t;s) - v_\beta^*(y,t;s)\right\|_2 \leq L\|x-y\|_2.$$

The distributions $\pi_\theta(\cdot \mid s)$ and $p_\beta^*(\cdot \mid s)$ have finite second moments, and $\mathcal{L}_{\mathrm{AA}}(\theta) < \infty$.

Lemma B.4 (Endpoint mismatch is controlled by the flow residual)**.** Under Assumption B.3, let X_t solve Eq.29, and let y_t be any absolutely continuous path initialized at the same point, $y_0 = X_0$. Then

$$\|y_1 - X_1\|_2^2 \leq e^{2L}\int_0^1 \left\|\dot{y}_t - v_\beta^*(y_t, t; s)\right\|_2^2 dt. \tag{30}$$

Proof. The claim is immediate if the integral on the right is infinite. Otherwise, define

$$e_t := y_t - X_t, \qquad r_t := \dot{y}_t - v_\beta^*(y_t, t; s).$$

Since $e_0 = 0$, the ODE and the Lipschitz assumption give

$$\|e_t\|_2 \leq \int_0^t \left(L\|e_\tau\|_2 + \|r_\tau\|_2\right) d\tau.$$

Grönwall's inequality and Cauchy–Schwarz therefore yield

$$\begin{aligned}\|e_1\|_2 &\leq \int_0^1 e^{L(1-t)}\|r_t\|_2\, dt \\ &\leq e^L\left(\int_0^1 \|r_t\|_2^2\, dt\right)^{1/2}.\end{aligned}$$

Squaring proves Eq.30. □

Theorem B.5 (Wasserstein guarantee for Q-Adjoint Anchoring). Under Assumption B.3,

$$\mathbb{E}_{s\sim\mathcal{D}}\left[W_2^2\left(\pi_\theta(\cdot\mid s), p_\beta^*(\cdot\mid s)\right)\right] \le e^{2L}\mathcal{L}_{\mathrm{AA}}(\theta). \tag{31}$$

Let $X_t(s,\epsilon)$ denote the solution of Eq.29 with $X_0 = \epsilon$. In particular, the loss vanishes in the straight-flow case where, for almost every (s,ϵ),

$$X_t(s,\epsilon) = x_{t,\theta} = (1-t)\epsilon + t\mu_\theta^\pi(s,\epsilon), \qquad 0 \le t \le 1,$$

or equivalently, $v_\beta^*(x_{t,\theta}, t; s) = d_\theta$ for almost every t. In this case, $\mathcal{L}_{\mathrm{AA}}(\theta) = 0$ and $\pi_\theta(\cdot \mid s) = p_\beta^*(\cdot \mid s)$ for $\mathcal{D}$-almost every s.

Proof. For fixed s and ϵ, let X_t solve Eq.29 with $X_0 = \epsilon$. When $\epsilon \sim \mathcal{N}(0, I_m)$, the two endpoints $\mu_\theta^\pi(s,\epsilon)$ and X_1 have distributions $\pi_\theta(\cdot \mid s)$ and $p_\beta^*(\cdot \mid s)$. Using the same noise gives a coupling, so

$$W_2^2\left(\pi_\theta(\cdot \mid s), p_\beta^*(\cdot \mid s)\right) \le \mathbb{E}_\epsilon\left[\|\mu_\theta^\pi(s,\epsilon) - X_1\|_2^2\right]. \tag{32}$$

Apply Lemma B.4 to $y_t = x_{t,\theta}$. Since $y_0 = X_0 = \epsilon$ and $\dot{y}_t = d_\theta$,

$$\|\mu_\theta^\pi(s,\epsilon) - X_1\|_2^2 \le e^{2L}\int_0^1 \|d_\theta - v_\beta^*(x_{t,\theta}, t; s)\|_2^2\, dt. \tag{33}$$

Taking expectations over s and ϵ and using the uniform sampling of t proves Eq.31. In the straight-flow case the integrand vanishes, so both the loss and the Wasserstein distance are zero. □

B.3 NOISE-CONDITIONED Q-CHUNKING

Definition B.6 (Conditional horizon reduction)**.** Fix the policy π_θ and horizons $H \geq n \geq 1$, with the actor producing n-step action chunks. Partition an H-step action sequence into the retained prefix and omitted suffix:

$$c := a_{0:n-1}, \qquad b := a_{n:H-1}, \qquad a_{0:H-1} = (c, b).$$

Here, c is the n-step action chunk retained by the reduced value function, while b contains the remaining $H - n$ actions. When $H = n$, the suffix b is empty.

Conditioned on the retained input (s, c), independently sample the omitted suffix B and critic noise E:

$$B \sim \mathcal{D}(\cdot \mid s, c), \qquad E \sim \mathcal{N}(0, I_m). \tag{34}$$

All expectations over B, E below use this conditional sampling rule.

For a bounded critic Q, define

$$X^Q_{s,c} := Q(s, (c, B), E). \tag{35}$$

Its randomness comes from the omitted suffix B and critic noise E.

Definition B.7 (Expectile and essential-supremum functionals)**.** For $\kappa \in [\frac{1}{2}, 1)$, define the asymmetric squared loss

$$L_2^\kappa(w) := |\kappa - \mathbf{1}\{w < 0\}|\, w^2.$$

For a bounded random variable X, its κ-expectile is

$$e_\kappa(X) := \arg\min_{z \in \mathbb{R}} \mathbb{E}\left[L_2^\kappa(X - z)\right], \tag{36}$$

where the expectation is taken with respect to the law of X.

For the limiting case $\kappa = 1$, define

$$e_1(X) := \operatorname{ess\,sup} X. \tag{37}$$

The essential supremum is the smallest number that bounds X with probability one. Thus, $e_\kappa(X^Q_{s,c})$ is the conditional upper expectile over the suffix and noise, while $e_1(X^Q_{s,c})$ is their conditional essential supremum. The latter is defined directly, not as a unique minimizer of the squared loss at $\kappa = 1$.

Lemma B.8 (Non-expansiveness of expectiles)**.** Let X and Y be bounded random variables on the same probability space. For $\kappa \in [\frac{1}{2}, 1]$, if $|X - Y| \leq \delta$ almost surely, then

$$|e_\kappa(X) - e_\kappa(Y)| \leq \delta. \tag{38}$$

For $\kappa < 1$, $e_\kappa(X)$ is unique, non-decreasing in κ, and satisfies

$$e_{1/2}(X) = \mathbb{E}[X], \qquad \operatorname{ess\,inf} X \leq e_\kappa(X) \leq e_1(X).$$

Proof. For $\kappa < 1$, strict convexity and coercivity give a unique minimizer. Its first-order condition is

$$\kappa \mathbb{E}[(X - z)_+] = (1 - \kappa)\mathbb{E}[(z - X)_+], \tag{39}$$

where $(w)_+ := \max\{w, 0\}$. The left side minus the right side is strictly decreasing in z and non-decreasing in X and κ, so its zero is monotone in X and κ. Shifting X and z by the same constant leaves the equation unchanged, proving $e_\kappa(X + c) = e_\kappa(X) + c$. At $\kappa = 1/2$, the equation gives $z = \mathbb{E}[X]$; for all $\kappa < 1$, the root lies between the essential infimum and supremum of X. Finally, $Y - \delta \leq X \leq Y + \delta$ implies

$$e_\kappa(Y) - \delta \leq e_\kappa(X) \leq e_\kappa(Y) + \delta,$$

which proves Eq.38. For $\kappa = 1$, the same argument applies because the essential supremum is also monotone and translation equivariant. □

Definition B.9 (Noise-conditioned Bellman operator)**.** Let

$$R_H := \sum_{i=0}^{H-1} \gamma^i r_i.$$

For a bounded critic Q, define

$$(\mathsf{T}_\kappa^\pi Q)(s, a_{0:H-1}, \varepsilon) := \mathbb{E}\left[R_H + \gamma^H e_\kappa\left(X^Q_{s_H, c_H}\right) \,\middle|\, s_0 = s,\, a_{0:H-1}\right], \tag{40}$$
$$c_H := \mu_\theta^\pi(s_H, \varepsilon).$$

Here, ε is held fixed in the conditional expectation and determines the retained chunk at s_H. In contrast, $X^Q_{s_H,c_H}$ varies over the omitted suffix and independent critic noise according to Definition B.6.

The corresponding population critic-regression objective is

$$\mathcal{L}_Q(\widetilde{Q}; Q) := \mathbb{E}\left[\left(\widetilde{Q}(s_0, a_{0:H-1}, \epsilon') - \left[R_H + \gamma^H e_\kappa\left(X^Q_{s_H, c'_H}\right)\right]\right)^2\right], \tag{41}$$

where

$$\epsilon' \sim \mathcal{N}(0, I_m), \qquad c'_H := \mu_\theta^\pi(s_H, \epsilon'),$$

and the expectation is over the sampled H-step trajectory and ϵ', which is independent of that trajectory.

Let $\mathcal{Q}$ be the space of bounded measurable critics, with

$$\|Q\|_\infty := \sup_{s, a_{0:H-1}, \epsilon} |Q(s, a_{0:H-1}, \epsilon)|.$$

Assume bounded rewards, $0 \le \gamma < 1$, a fixed measurable policy, and fixed measurable conditional trajectory and suffix laws. These laws agree with the regression sampling law on its support and are specified at all inputs needed by the operator, including queried inputs outside the data support.

Theorem B.10 (**Contraction of the noise-conditioned Bellman operator**)**.** For every $\kappa \in [\frac{1}{2}, 1]$, the unrestricted population minimizer of Eq.41 is $\mathsf{T}_\kappa^\pi Q$ almost everywhere under the critic-input distribution. Moreover,

$$\|\mathsf{T}_\kappa^\pi Q_1 - \mathsf{T}_\kappa^\pi Q_2\|_\infty \le \gamma^H \|Q_1 - Q_2\|_\infty. \tag{42}$$

Therefore, T_κ^π has a unique bounded fixed point Q_κ^π, and its exact operator iterates $Q_{j+1} = \mathsf{T}_\kappa^\pi Q_j$ converge uniformly to Q_κ^π from every $Q_0 \in \mathcal{Q}$.

Proof. Squared-error regression is minimized by the conditional mean of its target. Since ϵ' is independent of the sampled trajectory, this mean is Eq.40.

Set $\delta = \|Q_1 - Q_2\|_\infty$. Using the same B, E for both critics gives $|X^{Q_1}_{s,c} - X^{Q_2}_{s,c}| \le \delta$ with probability one. By Lemma B.8,

$$|e_\kappa(X^{Q_1}_{s,c}) - e_\kappa(X^{Q_2}_{s,c})| \le \delta.$$

The reward terms cancel when subtracting the two Bellman targets. Taking their conditional expectations and then the supremum gives

$$\|\mathsf{T}_\kappa^\pi Q_1 - \mathsf{T}_\kappa^\pi Q_2\|_\infty \le \gamma^H \delta.$$

Bounded rewards and fixed measurable conditional laws ensure that T_κ^π maps $\mathcal{Q}$ into itself. The space $\mathcal{Q}$ is complete in the supremum norm, and $\gamma^H < 1$. Banach's fixed-point theorem therefore gives a unique fixed point and uniform convergence of the exact operator iterates. □

Definition B.11 (Upper expectile regression)**.** For $\kappa \in [\frac{1}{2}, 1)$, a fixed bounded critic Q, and a candidate reduced value function $\widetilde{Z}$, define

$$\mathcal{L}_Z(\widetilde{Z}; Q) := \mathbb{E}_{\substack{(s, a_{0:H-1}) \sim \mathcal{D}, \\ \epsilon \sim \mathcal{N}(0, I_m)}}\left[L_2^\kappa\left(Q(s, a_{0:H-1}, \epsilon) - \widetilde{Z}(s, a_{0:n-1})\right)\right]. \tag{43}$$

The input to $\widetilde{Z}$ retains only the first n actions, whereas the regression target varies over both the omitted suffix $a_{n:H-1}$ and the critic noise ϵ.

Theorem B.12 (Convergence to the suffix–noise essential supremum). Fix a bounded critic Q. For $\kappa \in [\frac{1}{2}, 1)$, choose the population minimizer of Eq.43 as

$$\widetilde{Z}^*_\kappa(s, c) = e_\kappa(X^Q_{s,c}). \tag{44}$$

This minimizer is unique up to equality almost everywhere under the data distribution. For the displayed choice, at every (s, c) with a specified conditional law,

$$\begin{aligned}\lim_{\kappa\to 1^-} \widetilde{Z}^*_\kappa(s, c) &= e_1(X^Q_{s,c}) \\ &= \operatorname*{ess\,sup}_{B,E} Q(s, (c, B), E).\end{aligned} \tag{45}$$

Consequently, for $Q = Q^\pi_N := Q^\pi_1$, the unique fixed point of T^π_1 from Theorem B.10,

$$\lim_{\kappa\to 1^-} \widetilde{Z}^*_\kappa(s, a_{0:n-1}) = \operatorname*{ess\,sup}_{\substack{a_{n:H-1}\sim\mathcal{D}(\cdot|s,a_{0:n-1}),\\ \epsilon\sim\mathcal{N}(0,I_m)}} Q^\pi_N(s, a_{0:H-1}, \epsilon). \tag{46}$$

At finite $\kappa < 1$, the population target is the conditional κ-expectile rather than the conditional essential supremum. The critic Q is held fixed throughout this expectile limit. The essential supremum is over the conditional suffix and noise in Eq.34, not over all environment outcomes.

Proof. Conditioning on the retained input (s, c) gives

$$\mathcal{L}_Z(\widetilde{Z}; Q) = \mathbb{E}_{(s,c)\sim\mathcal{D}}\left[\mathbb{E}_{B,E}\left[L^\kappa_2\big(Q(s, (c, B), E) - \widetilde{Z}(s, c)\big)\right]\right],$$

where the inner expectation uses Eq.34. By the definition of the expectile, each conditional loss is uniquely minimized at $e_\kappa(X^Q_{s,c})$, proving Eq.44.

Fix (s, c) and abbreviate

$$X := X^Q_{s,c}, \qquad \ell := \operatorname{ess\,inf} X, \qquad U := \operatorname{ess\,sup} X.$$

Lemma B.8 implies that $e_\kappa(X)$ increases to some $z^* \le U$ as $\kappa \to 1^-$. Its first-order condition gives

$$\begin{aligned}0 \le \mathbb{E}[(X - e_\kappa(X))_+] &= \frac{1-\kappa}{\kappa}\mathbb{E}[(e_\kappa(X) - X)_+] \\ &\le \frac{1-\kappa}{\kappa}(U - \ell) \xrightarrow[\kappa\to 1^-]{} 0.\end{aligned} \tag{47}$$

The positive part is bounded by $U - \ell$, so dominated convergence yields $\mathbb{E}[(X - z^*)_+] = 0$. Thus $X \le z^*$ with probability one, which implies $U \le z^*$. Therefore $z^* = U$, proving Eq.45. Taking $Q = Q^\pi_N = Q^\pi_1$ proves Eq.46. □

C Extended Ablation Studies

C.1 Full Result on Ablation Study 1

To ensure a fair comparison, our main baseline evaluations in Table 12 fix the horizon length at $H = 5$. In ablation study 1, we scale the horizon length to $H \geq 10$ to evaluate how performance varies with longer horizon lengths, setting $\alpha = 1$ and $\lambda = 3$.

Table 2: **Full result on the critic chunk horizon H.** Each cell shows 95% confidence interval of offline $\rightarrow$ online success rate averaged over 4 seeds. Numbers are bolded within 95% of the task's best value. Shading goes dark to light from 90%, 70% to 50% of the task's best online value. The final row averages the success rates on five tasks of `humanoidmaze-giant-navigate`.

Task \ H	10	15	20	25
humanoidmaze-giant-navigate-singletask-task1-v0	[82.89] [77.91] 85 → 84	[85.92] [88.95] **89** → **92**	[88.94] [87.92] **91** → 90	[86.95] [93.98] **91** → **96**
humanoidmaze-giant-navigate-singletask-task2-v0	[92.96] [97.99] 94 → **98**	[97.99] [99.100] **98** → **100**	[97.98] [99.100] **98** → **100**	[98.100] [99.100] **99** → **99**
humanoidmaze-giant-navigate-singletask-task3-v0	[53.60] [78.88] 57 → 83	[70.81] [90.96] 76 → **93**	[75.84] [90.96] 79 → **93**	[81.87] [93.97] **84** → **95**
humanoidmaze-giant-navigate-singletask-task4-v0	[90.96] [97.100] 93 → **98**	[95.98] [98.100] **97** → **99**	[96.99] [98.100] **98** → **99**	[97.99] [98.100] **98** → **99**
humanoidmaze-giant-navigate-singletask-task5-v0	[97.100] [98.100] **99** → **99**	[95.99] [66.100] **98** → 88	[95.98] [98.100] **97** → **99**	[92.99] [98.100] **96** → **99**
average	[83.88] [89.96] 86 → 92	[88.94] [88.98] **92** → **94**	[90.95] [94.98] **93** → **96**	[91.96] [96.99] **94** → **98**

Table 2 presents the performance of NQA across five tasks from the OGBench (Park et al., 2025a) `humanoidmaze-giant-navigate` suite, where we observe that NQA's performance scales with the horizon length H. To the best of our knowledge, the prior state-of-the-art algorithm on OGBench `humanoidmaze-giant-navigate` with a 100M-sample dataset is Long-horizon Q-Learning (LQL) (Abraham et al., 2026), achieving an average success rate of 76% after 1M+1M offline-to-online training steps with $H = 64$. NQA achieves a higher average success rate ($> 90\%$) in fewer training steps (1M+0.5M steps) while operating over a significantly shorter horizon ($H \leq 25$).

C.2 Noise-conditioned Q-Chunking for FAN and QAM-F

We further apply **Noise-conditioned Q-Chunking (NQC)** to FAN (Lee et al., 2026) and QAM-F (Li & Levine, 2026), both of which learn straight flow policies. We compare each method with and without NQC on `humanoidmaze-medium-navigate-singletask-task1`, and include **Q-Adjoint Anchoring (QAA)** versus **NQA** as a reference.

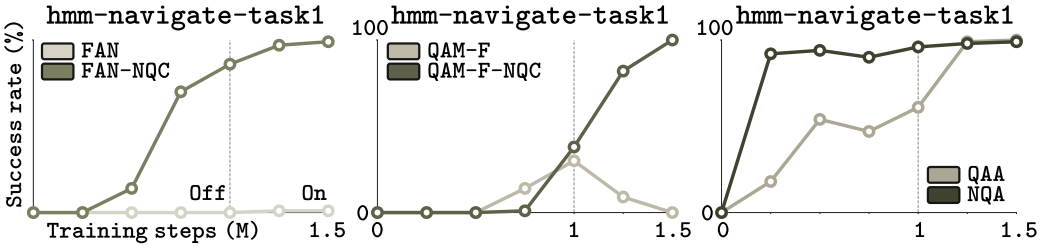


Figure 8: **NQC with straight flow policy learning.** NQC improves the performance of all three.

Figure 8 shows that NQC improves all three policy learning methods. FAN and QAM-F exhibit substantially higher success rates when equipped with NQC, while **NQA** improves the learning speed of QAA. These results suggest that the benefits of NQC extend beyond QAA and complement different objectives for learning straight flow policies.

C.3 Sensitivity Analysis

C.3.1 α, λ

We further analyze the sensitivity of task success rates to our main hyperparameters, α and λ.

Table 3: **Sensitivity Analysis on α and λ.** Each cell shows 95% confidence interval of offline $\rightarrow$ online success rate averaged over 4 seeds. Numbers are bolded within 95% of the task's best value. Shading goes dark to light from 90%, 70% to 50% of the task's best online value.

antmaze-giant-navigate-singletask-task1-v0

$\alpha \backslash \lambda$	0.1	0.3	1.0	3.0	-1
0.3	[0,0] [0,0] 0 → 0	[0,0] [6,9] 0 → 7	[3,8] [47,77] **6** → 64	[2,12] [81,90] **6** → **87**	[0,22] [84,96] **6** → **88**
1.0	[0,0] [0,0] 0 → 0	[0,1] [0,1] 0 → 0	[3,6] [21,37] 4 → 30	[3,9] [41,53] **6** → 46	[2,6] [35,58] 4 → 44
3.0	[0,0] [0,0] 0 → 0	[0,0] [0,0] 0 → 0	[0,0] [0,2] 0 → 1	[0,1] [0,2] 0 → 1	[0,1] [0,3] 0 → 1
10	[0,0] [0,0] 0 → 0	[0,0] [0,0] 0 → 0	[0,0] [0,0] 0 → 0	[0,0] [0,0] 0 → 0	[0,0] [0,0] 0 → 0
30	[0,0] [0,0] 0 → 0	[0,0] [0,0] 0 → 0	[0,0] [0,0] 0 → 0	[0,0] [0,0] 0 → 0	[0,0] [0,0] 0 → 0

humanoidmaze-large-navigate-singletask-task1-v0

$\alpha \backslash \lambda$	0.1	0.3	1.0	3.0	-1
0.3	[0,1] [0,1] 0 → 0	[28,44] [47,59] 33 → 55	[55,86] [91,99] **68** → **95**	[0,81] [98,100] 51 → **99**	[0,55] [0,98] 36 → 64
1.0	[0,0] [0,1] 0 → 0	[10,15] [24,33] 13 → 29	[57,70] [67,85] 64 → 76	[60,78] [80,90] **69** → 85	[0,71] [0,85] 45 → 54
3.0	[0,0] [0,0] 0 → 0	[0,1] [0,1] 0 → 1	[1,5] [5,9] 2 → 7	[8,9] [14,15] 8 → 15	[1,8] [0,15] 6 → 10
10	[0,0] [0,0] 0 → 0	[0,0] [0,1] 0 → 0	[0,1] [0,0] 0 → 0	[0,0] [0,0] 0 → 0	[0,0] [0,0] 0 → 0
30	[0,0] [0,0] 0 → 0	[0,0] [0,1] 0 → 0	[0,0] [0,1] 0 → 0	[0,0] [0,0] 0 → 0	[0,0] [0,0] 0 → 0

puzzle-4x4-play-singletask-task4-v0

$\alpha \backslash \lambda$	0.1	0.3	1.0	3.0	-1
0.3	[1,1] [0,0] 1 → 0	[9,13] [85,97] 11 → 91	[39,49] [92,100] **43** → **97**	[0,4] [34,97] 2 → 60	[0,0] [0,0] 0 → 0
1.0	[0,1] [0,0] 1 → 0	[9,14] [92,99] 11 → **95**	[15,30] [99,100] 23 → **99**	[13,39] [2,97] 23 → 47	[0,0] [0,0] 0 → 0
3.0	[0,2] [0,0] 1 → 0	[11,17] [88,95] 14 → 92	[19,32] [100,100] 26 → **100**	[15,23] [99,100] 18 → **100**	[0,32] [0,64] 12 → 21
10	[0,1] [0,0] 0 → 0	[9,14] [1,80] 11 → 31	[20,34] [100,100] 28 → **100**	[24,29] [99,100] 26 → **100**	[15,25] [30,100] 20 → 65
30	[0,1] [0,0] 0 → 0	[4,8] [1,3] 6 → 2	[27,30] [83,90] 29 → 88	[19,27] [97,100] 24 → **98**	[23,25] [39,63] 24 → 51

scene-play-singletask-task2-v0

$\alpha \backslash \lambda$	0.1	0.3	1.0	3.0	-1
0.3	[7,12] [50,65] 10 → 57	[68,77] [99,100] 71 → **100**	[100,100] [99,100] **100** → **100**	[54,86] [100,100] 67 → **100**	[27,32] [99,100] 28 → **100**
1.0	[7,14] [32,55] 10 → 41	[70,81] [99,100] 75 → **100**	[100,100] [99,100] **100** → **100**	[66,99] [99,100] 88 → **100**	[47,88] [99,100] 71 → **100**
3.0	[6,12] [33,47] 10 → 41	[58,65] [100,100] 62 → **100**	[100,100] [99,100] **100** → **100**	[100,100] [99,100] **100** → **100**	[99,100] [100,100] **100** → **100**
10	[6,16] [22,45] 11 → 31	[56,70] [98,100] 63 → **99**	[99,100] [100,100] **100** → **100**	[99,100] [99,100] **100** → **100**	[100,100] [100,100] **100** → **100**
30	[5,8] [18,28] 6 → 23	[38,46] [85,95] 42 → 91	[80,98] [99,99] 91 → **99**	[90,99] [99,100] **95** → **100**	[88,100] [100,100] **96** → **100**

Table 3 reports these results. A tight clipping bound ($\lambda = 0.1$) degrades performance across all tasks, indicating that the policy requires sufficient flexibility to deviate from the offline behavior policy. Navigation tasks (`antmaze-giant`, `humanoidmaze-large`) require a small α (≤ 1.0) and fail with larger values, whereas manipulation tasks (`puzzle-4x4`, `scene`) remain robust across nearly the entire α range. Notably, `puzzle-4x4` is the only case where removing the clipping bound entirely ($\lambda = -1$) harms performance, particularly when paired with a small α. Overall, NQA degrades smoothly as α and λ deviate from their optimal values, except when clipping is excessively restrictive.

C.3.2 κ

We also evaluate the impact of different κ values on the final performance of NQA.

Table 4: **Sensitivity analysis on κ.** Each cell shows 95% confidence interval of offline $\rightarrow$ online success rate. Numbers are bolded within 95% of the task's best value. Shading goes dark to light from 90%, 70% to 50% of the task's best online value.

Task \ κ	0.5	0.7	0.9	0.95	0.99
antmaze-giant-navigate-singletask-task1-v0	[0,8] [70,90] 4 → 83	[0,8] [88,98] 3 → **93**	[0,11] [84,96] **6** → 88	[0,2] [32,88] 1 → 58	[0,0] [0,0] 0 → 0
humanoidmaze-large-navigate-singletask-task1-v0	[80,94] [86,94] **88** → **90**	[72,88] [90,98] 80 → **93**	[60,78] [80,90] 69 → 85	[23,74] [0,72] 50 → 19	[14,20] [0,6] 17 → 2
puzzle-4x4-play-singletask-task4-v0	[20,44] [0,100] **33** → 75	[28,36] [100,100] 31 → **100**	[20,34] [100,100] 28 → **100**	[24,42] [96,100] 31 → **99**	[16,50] [84,92] 30 → 86
scene-play-singletask-task2-v0	[96,100] [100,100] **98** → **100**	[100,100] [100,100] **100** → **100**	[88,100] [100,100] **96** → **100**	[96,100] [88,100] **98** → **97**	[80,92] [82,100] 87 → 90

As reported in Table 4, NQA achieves optimal performance across all four tasks when κ lies between 0.7 and 0.9. Setting κ too high (e.g., 0.95 or 0.99) degrades performance, validating our choice of $\kappa = 0.9$ as a reliable default.

C.3.3 n

We also evaluate the impact of different n values on the final performance of NQA.

Table 5: **Sensitivity analysis on the action horizon length n.** Each cell shows 95% confidence interval of offline $\rightarrow$ online success rate. Numbers are bolded within 95% of the task's best value. Shading goes dark to light from 90%, 70% to 50% of the task's best online value.

TASK $\setminus$ n	1	2	3	4	5
ANTMAZE-GIANT-NAVIGATE-SINGLETASK-TASK1-V0	[0,22] [84,96] **6** $\rightarrow$ **88**	[0,1] [43,85] 0 $\rightarrow$ 64	[0,0] [0,55] 0 $\rightarrow$ 28	[0,0] [18,55] 0 $\rightarrow$ 37	[0,0] [1,25] 0 $\rightarrow$ 13
HUMANOIDMAZE-LARGE-NAVIGATE-SINGLETASK-TASK1-V0	[60,78] [80,90] **69** $\rightarrow$ **85**	[51,59] [87,89] 55 $\rightarrow$ **88**	[29,39] [56,61] 34 $\rightarrow$ 58	[3,5] [3,8] 4 $\rightarrow$ 6	[0,0] [0,0] 0 $\rightarrow$ 0
PUZZLE-4X4-PLAY-SINGLETASK-TASK4-V0	[10,14] [41,57] 12 $\rightarrow$ 50	[29,33] [99,100] 31 $\rightarrow$ **99**	[20,34] [100,100] 28 $\rightarrow$ **100**	[29,39] [99,100] **33** $\rightarrow$ **100**	[21,27] [100,100] 24 $\rightarrow$ **100**
SCENE-PLAY-SINGLETASK-TASK2-V0	[63,88] [43,68] 76 $\rightarrow$ 55	[90,96] [98,100] 93 $\rightarrow$ **99**	[88,100] [100,100] **96** $\rightarrow$ **100**	[98,100] [99,100] **99** $\rightarrow$ **100**	[98,100] [100,100] **99** $\rightarrow$ **100**

As reported in Table 5, the optimal action horizon length n is task-dependent: the navigation tasks (`antmaze-giant`, `humanoidmaze-large`) achieve their best performance at a short horizon ($n = 1$) and degrade as n grows, collapsing to near-zero success at $n = 5$, whereas the manipulation tasks (`puzzle-4x4`, `scene`) perform poorly at $n = 1$ and instead require $n \geq 2$. Setting n to the wrong regime for a given task category substantially degrades performance, validating our choice of a short action horizon for navigation and a longer one for manipulation.

D Experimental Details

NQA is built on JAX (Bradbury et al., 2018), following the code implementations of QAM (Li & Levine, 2026) and FAN (Lee et al., 2026).

D.1 Benchmark

OGBench. OGBench (Park et al., 2025a) is a benchmark originally proposed for offline goal-conditioned RL algorithms. In addition to goal-conditioned tasks, it provides `singletask` variants for general offline RL, as well as online evaluation environments for general off-policy algorithms. These `singletask` environments use semi-sparse reward functions, where the transition reward equals the negative number of remaining subtasks at a given state. Locomotion tasks involve a single subtask, yielding rewards of -1 or 0. Manipulation tasks comprise multiple subtasks, bounding rewards between $-n_{\text{subtask}}$ and 0.

We evaluate on 65 state-based `singletask-v0` datasets from OGBench, focusing particularly on the benchmark's most challenging environments, such as `humanoidmaze-giant-navigate`, `cube-quadruple-play`, and `puzzle-4x6-play`. The 65 datasets consist of:

1. **1M-sized State-based Datasets** (5 dataset variants per task):
   - `antmaze-large-navigate`
   - `antmaze-giant-navigate`
   - `antsoccer-arena-navigate`
   - `humanoidmaze-medium-navigate`
   - `humanoidmaze-large-navigate`
   - `cube-double-play`
   - `cube-triple-play`
   - `puzzle-4x4-play`
   - `puzzle-4x5-play`
   - `scene-play`
2. **100M-sized State-based Datasets** (5 dataset variants per task):
   - `humanoidmaze-giant-navigate`
   - `cube-quadruple-play`
   - `puzzle-4x6-play`

For locomotion tasks, we target `navigate` datasets: `antmaze-{large,giant}` for quadrupedal maze navigation, `antsoccer-arena` for quadrupedal ball dribbling, and `humanoidmaze-{medium,large,giant}` for high-dimensional humanoid maze navigation.

For manipulation tasks, we use `play` datasets: `cube-{double,triple,quadruple}` for pick-and-place manipulation, `puzzle-{4x4,4x5,4x6}` for combinatorial generalization on "Lights Out" puzzles, and `scene` for long-horizon object interaction.

D.2 Baseline Methods

We compare NQA against eight representative baselines, sweeping hyperparameters for each method over the ranges specified below. For each task domain, we tune hyperparameters on a single representative task and apply the best-performing configuration across all related tasks. Specifically, tuning on `task1` is applied to `cube-triple-play`, `cube-quadruple-play`, `puzzle-4x5-play`, and `puzzle-4x6-play`. All other task environments are tuned on their respective default tasks.

IDQL. Implicit Diffusion Q-Learning (IDQL) (Hansen-Estruch et al., 2023) combines IQL (Kostrikov et al., 2021) value learning with policy extraction from an expressive behavior diffusion model. Following the prior work of Park et al., we evaluate its flow counterpart, *i.e.*, IFQL, which samples action candidates from a behavior-cloning flow and selects the highest-value action. We use 10 flow steps and sweep the expectile $\kappa \in \{0.5, 0.6, 0.7, 0.8, 0.9, 0.95\}$ and fix the number of candidates $N = 32$.

DSRL. Diffusion Steering via Reinforcement Learning (DSRL) (Wagenmaker et al., 2025) performs RL in the latent noise space of a pretrained diffusion or flow policy, steering its outputs without backpropagating through the generative trajectory. We use the flow variant with 10 flow steps and sweep the latent noise scale $\sigma_z \in \{0.1, 0.2, 0.4, 0.5, 0.6, 0.8, 1, 1.2, 1.4\}$.

QAM. Q-learning with Adjoint Matching (QAM) (Li & Levine, 2026) converts terminal critic gradients into step-wise policy objectives through adjoint matching, avoiding BPTT while retaining an expressive multi-step flow policy. We use 10 integration steps and sweep the temperature coefficient $\tau \in \{0.1, 0.3, 1, 3, 10, 30\}$.

DQC. Decoupled Q-Chunking (DQC) (Li et al., 2026a) decouples the critic and policy chunk horizons, allowing long-horizon value propagation with shorter policy execution horizons. We fix the critic chunk horizon to $h = 5$ and use the action chunk horizon $h_a = 1$ for locomotion and $h_a = 3$ for manipulation tasks. We sweep the distillation expectile $\kappa_d \in \{0.5, 0.7, 0.8, 0.9, 0.95\}$ and the backup quantile $\kappa_b \in \{0.5, 0.7, 0.9, 0.95\}$.

ReBRAC. ReBRAC (Tarasov et al., 2023) extends TD3+BC (Fujimoto & Gu, 2021) with separate behavior-cloning regularization for the actor and critic. We sweep the actor coefficient $\alpha_1 \in \{0.003, 0.01, 0.03, 0.1, 0.3, 1\}$ and set critic coefficient $\alpha_2 = 0$. We sweep the actor noise scale $\sigma_a \in \{0.0, 0.1, 0.2\}$.

FQL. Flow Q-Learning (FQL) (Park et al., 2025b) trains a straight flow policy to maximize the critic while remaining close to actions generated by a behavior-cloning flow. We use 10 flow steps for the behavior flow during training and sweep the behavior-regularization coefficient $\alpha \in \{0.3, 1, 3, 10, 30, 100, 300, 1000\}$.

CGQ. Chunk-Guided Q-Learning (CGQ) (Song et al., 2026) regularizes a fine-grained critic toward an action-chunked critic trained with multi-step TD backups. We fix the critic chunk horizon to $h = 5$ and use the action chunk horizon $h_a = 1$ for locomotion and $h_a = 3$ for manipulation tasks. Following the original paper, we sweep $(\alpha, \alpha_{\text{step}}) \in \{100, 300\}^2$, $\beta \in \{0.01, 0.1, 1\}$, and $\tau \in \{0.5, 0.7, 0.8, 0.95\}$ for manipulation tasks, and for locomotion tasks, we sweep $\alpha \in \{30, 100, 300\}$, $\alpha_{\text{step}} \in \{10, 30\}$, $\beta \in \{0.01, 0.1, 0.5, 1\}$, and $\tau \in \{0.5, 0.6, 0.7, 0.8, 0.95\}$.

FAN. Flow-Anchored Noise-conditioned Q-Learning (FAN) (Lee et al., 2026) combines a one-step flow-anchored policy with a noise-conditioned distributional critic that requires only a single noise sample per critic update. We fix the expectile $\kappa = 0.9$ and sweep $\alpha_1 \in \{10, 30, 100, 300\}$ and $\alpha_2 \in \{0, 0.1, 0.3, 1, 3\}$.

NQA. We sweep $\alpha \in \{0.3, 1.0, 3.0, 10, 30\}$ and $\lambda \in \{-1, 1, 3\}$, where $\lambda = -1$ indicates that there is no clipping bound for the adjoint term. We fix the expectile $\kappa = 0.9$ for all tasks.

QAM+DQC, QAM+NQC, QAA+DQC. In Ablation Study 2, we cross-evaluate the actor and critic components across QAM, DQC, and NQA. We sweep the temperature parameter $\tau (= \alpha) \in \{1.0, 3.0, 10, 30\}$ for QAM/QAA, the QAA clipping bound $\lambda \in \{1, 3\}$, and the DQC backup quantile $\kappa_b \in \{0.5, 0.7\}$. Meanwhile, we fix the DQC distillation expectile to $\kappa_d = 0.5$ and the NQC critic expectile to $\kappa = 0.9$. Specifically, we first determine the optimal hyperparameter combination for QAM, which yields $\tau = \alpha = 30$. Fixing this temperature, we subsequently sweep the remaining hyperparameter configurations for the other approaches.

D.3 Hyperparameters and Results

Table 6: **Hyperparameter configurations for NQA** shared across experiments.

Hyperparameter	Value
Learning rate	0.0003
Optimizer	Adam (Kingma & Ba, 2015)
Offline-to-Online Gradient steps (Offline)	1000000 (1M)
Offline-to-Online Gradient steps (Online)	500000 (0.5M)
Minibatch size	256
MLP dimensions	$[512, 512, 512, 512]$
Nonlinearity	GELU (Hendrycks & Gimpel, 2016)
Target network smoothing coefficient	0.005
Critic chunk horizon H	5
Action chunk horizon n	1 (locomotion), 3 (manipulation)
Expectile κ	0.9
Discount factor γ	0.995 (default), 0.999 (`humanoid`, 100M datasets)
Flow time sampling distribution	Unif($[0, 1]$)
Number of flow steps (adjoint backward, etc)	10
Number of Q_ϕ ensembles	2
Number of Z_ψ ensembles	2
Target value aggregation	mean
Temperature coefficient α	Please refer to Table 7
Adjoint clipping coefficient λ	Please refer to Table 7

Table 7: **Detailed hyperparameter configurations.** We adopt the search space used in prior work, and do not allow hyperparameter changes in offline-to-online conversions. The critic chunk horizon is 5 for all methods, and the action chunk horizons for DQC and CGQ follow Table 6. We use rejection sampling for multi-step flow methods, and fix the number of candidates to 32.

	Multi-step Flow Inference					Single-step Flow Inference										
	IDQL	DSRL	QAM	DQC		ReBRAC		FQL	CGQ				FAN		NQA	
Task	κ	σ_z	τ	κ_d	κ_b	α	σ_a	α	α	α_{step}	β	τ	α_1	α_2	α	λ
antmaze-large-navigate	0.9	0.8	1	0.5	0.5	0.01	0	3	100	10	0.01	0.7	10	0.1	1	3
antmaze-giant-navigate	0.8	1.2	3	0.5	0.5	0.01	0.1	3	300	10	0.01	0.8	10	0.1	0.3	-1
antsoccer-arena-navigate	0.9	0.8	10	0.5	0.5	0.01	0	3	100	10	0.01	0.7	10	0.1	0.3	3
humanoidmaze-medium-navigate	0.7	0.6	3	0.5	0.5	0.01	0	30	100	10	0.5	0.8	10	0.1	1	3
humanoidmaze-large-navigate	0.8	0.8	3	0.5	0.5	0.01	0.1	30	100	10	0.5	0.8	10	0.1	1	3
humanoidmaze-giant-navigate-100M	0.8	0.8	3	0.5	0.5	0.01	0.1	30	100	10	0.5	0.8	10	0.1	1	3
cube-double-play	0.9	1	1	0.8	0.5	0.01	0	300	100	100	1	0.8	100	0	1	1
cube-triple-play	0.95	1.4	3	0.8	0.5	0.01	0.2	30	300	300	0.01	0.8	100	0	1	3
cube-quadruple-play-100M	0.95	1.4	1	0.8	0.5	0.01	0.2	100	300	300	0.01	0.95	100	0	1	1
puzzle-4x4-play	0.9	0.5	10	0.5	0.9	0.01	0.2	300	300	300	0.1	0.5	100	3	10	1
puzzle-4x5-play	0.9	1	30	0.5	0.5	0.01	0.2	100	300	300	0.1	0.7	100	3	3	3
puzzle-4x6-play-100M	0.9	1	30	0.5	0.5	0.01	0.2	300	300	300	0.1	0.5	100	3	10	1
scene-play	0.9	0.4	1	0.5	0.5	0.03	0	300	300	100	0.1	0.8	100	0	30	-1

Table 8: **Hyperparameter configurations for Observation 1.** All variants use Q-chunking, where the critic backup horizon and policy execution horizon are both 5 for all cases. We fix $N = 32$ for all variants, the number of action candidates used for rejection sampling. The remaining hyperparameters follow Table 6 and are kept fixed throughout the offline-to-online transition. SDE variants retain the diffusion weighting of QAM, and ODE variants omit the diffusion weighting. We use 4 random seeds to plot Figure 3.

	(i) QAM-SDE-Endpoint	*(ii) QAM-SDE-QGF*	*(iii) QAM-ODE-Endpoint*	*(iv) QAM-ODE-QGF*
TASK	α	α	α	α
ANTMAZE-LARGE-NAVIGATE (TASK 1)	1	1	1	1
ANTSOCCER-ARENA-NAVIGATE (TASK 4)	10	10	10	10
CUBE-DOUBLE-PLAY (TASK 2)	1	1	1	1
PUZZLE-4X4-PLAY (TASK 4)	10	10	10	10

Table 9: **Hyperparameter configurations for Observation 2.** All variants retain FQL's policy learning mechanism, and n denotes the policy execution horizon. Hyperparameters are kept fixed throughout the offline-to-online transition. We use 4 random seeds to plot Figure 4.

	(i) FQL-TD(1)		*(ii) FQL-TD(5)*		*(iii) FQL-QC(5)*		*(iv) NFQL-TD(1)*			*(v) NFQL-TD(5)*			*(vi) NFQL-QC(5)*		
TASK	α	n	α	n	α	n	α	κ	n	α	κ	n	α	κ	n
CUBE-DOUBLE-PLAY (TASK 2)	300	1	300	1	300	5	300	0.9	1	300	0.9	1	300	0.9	5
PUZZLE-4X4-PLAY (TASK 4)	300	1	300	1	300	5	300	0.9	1	300	0.9	1	300	0.9	5

Table 10: **Hyperparameter configurations for ablation study 2, Figure 7 *(Right)*.** The remaining hyperparameters follow Table 6 and are kept fixed throughout the offline-to-online transition. For a fair comparison, we first search for the best hyperparameter combination for QAM + DQC, and then apply similar parameter values to the other methods. We use 4 random seeds.

	QAM + DQC				QAM + NQC			QAA + DQC				NQA (QAA + NQC)		
TASK	α	κ_b	κ_d	N	α	κ	N	α	λ	κ_b	κ_d	α	λ	κ
PUZZLE-4X6-PLAY (TASK1)	30	0.7	0.5	32	30	0.9	32	30	3	0.7	0.5	30	3	0.9

Table 11: **Hyperparameter configurations for ablation study 2, Figure 8.** All methods use a policy execution horizon of 1. The critic action input and TD backup horizons are 1 for FAN, QAM-F, and QAA, and 5 for FAN-NQC, QAM-F-NQC, and NQA. The remaining hyperparameters follow Table 6 and are kept fixed throughout the offline-to-online transition. We use 4 random seeds.

	FAN		FAN-NQC		QAM-F		QAM-F-NQC		QAA		NQA	
TASK	α_1	α_2	α_1	α_2	α	α_{FQL}	α	α_{FQL}	α	λ	α	λ
HUMANOIDMAZE-MEDIUM-NAVIGATE (TASK 1)	10	0.1	10	0.1	1	30	1	30	1	3	1	3

Table 12: **Full offline-to-online results** for Table 1. We evaluate across 8 random seeds using the hyperparameters specified in Tables 6–7. Bracketed bounds denote 95% confidence intervals. Result numbers are bolded if they fall within 95% of the best performance on each task, and are colored in green if both offline and online results are bolded except for zero success rates. The offline dataset size is 100M transitions for `humanoidmaze-giant-navigate`, `cube-quadruple-play`, and `puzzle-4x6-play`, while all other tasks use default dataset sizes.

		Multi-step Flow Inference				Single-step Flow Inference				
Task Env	Task	IDQL	DSRL	QAM	DQC	ReBRAC	FQL	CGQ	FAN	NQA
antmaze-large	task1	1 → **95**	7 → 88	7 → **95**	5 → 27	**87** → 31	0 → **97**	**91** → **99**	21 → 92	**87** → **96**
	task2	1 → 81	0 → 89	0 → **95**	0 → 1	71 → 0	0 → 27	0 → **97**	0 → 37	**80** → 92
	task3	15 → 91	4 → **95**	15 → **97**	3 → 1	**90** → **99**	0 → 79	81 → **100**	0 → **96**	**90** → **95**
	task4	7 → 83	0 → 82	0 → 89	9 → 15	**84** → 94	0 → 84	5 → **99**	0 → 76	**85** → 94
	task5	5 → 83	25 → 85	15 → 89	26 → 51	1 → 59	0 → 43	**94** → **98**	0 → 83	**90** → **95**
antmaze-giant	task1	0 → 0	0 → 33	0 → 87	0 → 0	5 → 51	0 → 1	0 → **94**	0 → 3	**6** → 88
	task2	3 → 23	0 → 61	0 → 83	0 → 0	38 → **87**	0 → 39	0 → **91**	0 → 82	**82** → **89**
	task3	0 → 0	0 → 37	0 → 40	0 → 0	14 → 53	0 → 23	0 → 45	0 → 40	**45** → 77
	task4	0 → 1	0 → 72	0 → **80**	0 → 0	1 → 27	0 → 31	0 → 77	0 → 35	**40** → **82**
	task5	1 → 69	0 → 77	0 → 87	0 → 0	77 → **94**	0 → 67	0 → **98**	0 → 57	54 → 93
antsoccer-arena	task1	17 → 31	31 → 70	56 → 87	42 → 1	49 → 20	0 → 4	75 → **95**	0 → 69	**90** → **96**
	task2	0 → 5	22 → 81	47 → 87	3 → 0	15 → 17	0 → 13	**87** → **97**	0 → 83	**89** → **98**
	task3	7 → 5	8 → 57	33 → 77	11 → 1	43 → 14	0 → 12	49 → **83**	0 → 23	**56** → **84**
	task4	0 → 1	1 → 51	15 → 45	7 → 1	21 → 2	0 → 0	**54** → 69	0 → 56	**52** → **76**
	task5	1 → 0	2 → 52	17 → 65	3 → 0	21 → 5	0 → 5	**51** → 82	0 → 52	27 → **87**
humanoidmaze-medium	task1	17 → 3	47 → 51	3 → 7	53 → 53	43 → 0	0 → 0	0 → 0	0 → 0	**96** → **99**
	task2	17 → 5	64 → 63	12 → 10	55 → 6	88 → 23	0 → 46	**94** → **96**	0 → 1	**98** → **100**
	task3	48 → 24	29 → 71	37 → 88	78 → 9	7 → 0	0 → 0	0 → 0	0 → 0	**96** → **99**
	task4	9 → 1	0 → 48	0 → **58**	0 → 0	15 → 0	0 → 0	0 → 0	0 → 0	**75** → **61**
	task5	45 → 39	55 → 75	0 → 0	49 → 5	**95** → 83	3 → 89	**96** → **99**	0 → 93	**98** → **100**
humanoidmaze-large	task1	0 → 0	0 → 12	0 → 0	10 → 1	1 → 0	0 → 0	0 → 0	0 → 0	**69** → **85**
	task2	0 → 0	0 → 0	0 → 0	0 → 0	0 → 0	0 → 0	0 → 0	0 → 0	**30** → **52**
	task3	1 → 0	0 → 45	0 → 7	10 → 0	9 → 5	0 → 1	0 → 63	0 → 7	**87** → **99**
	task4	0 → 0	0 → 2	0 → 0	1 → 0	0 → 0	0 → 0	0 → 0	0 → 0	**51** → **79**
	task5	0 → 0	0 → 1	0 → 0	7 → 3	0 → 0	0 → 0	0 → 0	0 → 0	**47** → **57**
humanoidmaze-giant-100m	task1	0 → 0	0 → 0	0 → 0	0 → 0	1 → 6	0 → 0	0 → 2	0 → 0	**25** → **72**
	task2	1 → 0	0 → 1	0 → 0	3 → 1	23 → 41	0 → 0	0 → 64	0 → 0	**35** → **92**
	task3	0 → 0	0 → 0	0 → 0	0 → 0	1 → 0	0 → 0	0 → 0	0 → 0	**32** → **46**
	task4	1 → 0	0 → 0	0 → 0	0 → 0	0 → 0	0 → 0	0 → 0	0 → 0	**49** → **42**
	task5	46 → 0	0 → 1	5 → 0	61 → 23	3 → 66	0 → 0	0 → 0	0 → 0	**79** → **87**
cube-double	task1	18 → **99**	**89** → **100**	**93** → **100**	18 → **99**	19 → **97**	60 → **99**	63 → **99**	75 → **99**	86 → **100**
	task2	10 → 67	**78** → **97**	58 → **100**	13 → 80	3 → 89	45 → **100**	45 → **97**	40 → **100**	68 → **99**
	task3	15 → 75	**87** → **99**	57 → **100**	11 → 87	0 → 75	28 → **100**	49 → 93	33 → **100**	65 → **99**
	task4	3 → 5	22 → 92	21 → **97**	3 → 1	0 → 28	7 → 86	2 → 47	2 → **99**	**31** → 94
	task5	13 → 77	66 → **98**	77 → **100**	2 → 84	1 → 24	28 → **99**	41 → 94	60 → **99**	76 → **97**
cube-triple	task1	2 → 1	21 → 39	12 → **98**	1 → 0	13 → 92	1 → 0	20 → **100**	22 → **99**	33 → **100**
	task2	0 → 0	0 → 0	1 → 3	0 → 0	0 → 1	0 → 0	1 → 71	2 → **97**	**4** → 85
	task3	0 → 0	3 → 1	2 → 60	0 → 0	0 → 0	0 → 0	1 → 55	4 → **96**	**8** → 82
	task4	0 → 0	0 → 1	0 → 3	0 → 0	0 → 0	0 → 1	0 → 4	1 → **95**	**3** → 82
	task5	0 → 0	0 → 0	0 → 0	0 → 0	0 → 0	0 → 0	0 → 1	0 → 59	1 → 72
cube-quadruple-100m	task1	**35** → **99**	21 → **100**	21 → **100**	3 → 0	11 → **100**	29 → **100**	2 → **100**	0 → **99**	30 → **95**
	task2	3 → 81	2 → **100**	0 → **99**	0 → 0	1 → **99**	0 → **95**	0 → 25	0 → 89	**5** → 88
	task3	6 → 69	1 → **91**	0 → **92**	0 → 0	0 → 25	0 → 1	0 → 21	0 → 85	**7** → 77
	task4	**0** → 33	**0** → **98**	**0** → **94**	**0** → 0	**0** → 27	**0** → 1	**0** → 0	**0** → 54	**0** → 75
	task5	**0** → 0	**0** → 5	**0** → 0	**0** → 0	**0** → 0	**0** → 0	**0** → 0	**0** → 0	**0** → **14**
scene	task1	63 → 3	**100** → **100**	**100** → **100**	**99** → **100**	91 → **100**	**100** → **100**	81 → **100**	**100** → **100**	**100** → **100**
	task2	1 → 1	**100** → **100**	**100** → **100**	0 → 0	83 → **100**	**98** → **100**	90 → **100**	**96** → **100**	**96** → **100**
	task3	6 → 5	**99** → **100**	**78** → **99**	**89** → **95**	3 → **95**	61 → **100**	3 → **99**	61 → **100**	**95** → **99**
	task4	0 → 1	**100** → **100**	0 → **99**	35 → 63	2 → 0	1 → 64	0 → 92	0 → 62	**97** → **99**
	task5	0 → 0	**94** → **99**	0 → 1	1 → 1	4 → 0	0 → 0	0 → 0	0 → 0	**90** → 94
puzzle-4x4	task1	29 → **100**	61 → **100**	70 → **100**	39 → **100**	14 → **99**	20 → 66	43 → **100**	55 → **100**	66 → **100**
	task2	13 → **100**	3 → 67	20 → **100**	15 → **100**	1 → 13	0 → 0	0 → 67	0 → 0	**31** → **99**
	task3	20 → 53	50 → **100**	**74** → **100**	29 → **100**	4 → **99**	2 → 67	26 → **100**	54 → 67	50 → **100**
	task4	11 → 0	11 → **99**	**31** → **100**	17 → 80	1 → 66	1 → 0	8 → 67	20 → 67	28 → **100**
	task5	9 → 0	7 → 0	16 → **100**	10 → **100**	0 → 0	0 → 0	0 → 0	0 → 0	**31** → **100**
puzzle-4x5	task1	47 → **97**	**96** → **100**	87 → **100**	**95** → **100**	78 → **100**	61 → **100**	66 → **100**	79 → **100**	**100** → **100**
	task2	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**
	task3	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**
	task4	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**
	task5	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**
puzzle-4x6-100m	task1	23 → **100**	**97** → **100**	49 → **100**	35 → **100**	11 → **100**	67 → **100**	23 → **100**	23 → **100**	42 → **97**
	task2	0 → **0**	0 → **0**	0 → **0**	**1** → **0**	0 → **0**	0 → **0**	0 → **0**	0 → **0**	**1** → **0**
	task3	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**
	task4	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**
	task5	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**	**0** → **0**

Table 13: **Comparison with offline results reported in prior work.** Baseline success rates for IDQL, DSRL, QAM, ReBRAC, and FQL are from Li & Levine (2026). DQC and CGQ results are from Song et al. (2026), and FAN results are from Lee et al. (2026). Scores at least 95% of the best value for each task are bolded. A dash (–) indicates that the source paper does not report a result with the same task variant, offline dataset size, and training budget as ours.

		Multi-step Flow Inference				Single-step Flow Inference				
Task Env	Task	IDQL	DSRL	QAM	DQC	ReBRAC	FQL	CGQ	FAN	NQA
antmaze-large	task1	41	62	77	72	**98**	93	88	–	87
	task2	14	75	80	61	**88**	**86**	74	–	80
	task3	54	81	89	74	**98**	59	92	–	90
	task4	25	18	69	73	**94**	54	26	–	85
	task5	45	69	89	74	**95**	86	55	–	90
antmaze-giant	task1	0	0	7	0	**36**	0	0	–	6
	task2	0	0	0	41	73	0	0	–	**82**
	task3	1	0	0	1	13	0	0	–	**45**
	task4	2	12	34	5	**74**	0	0	–	40
	task5	2	2	49	2	**89**	0	17	–	54
antsoccer-arena	task1	–	–	–	–	–	–	–	**89**	**90**
	task2	–	–	–	–	–	–	–	**91**	**89**
	task3	–	–	–	–	–	–	–	49	**56**
	task4	–	–	–	–	–	–	–	49	**52**
	task5	–	–	–	–	–	–	–	21	**27**
humanoidmaze-medium	task1	86	49	40	87	38	34	0	–	**96**
	task2	92	91	**97**	**94**	91	**95**	80	–	**98**
	task3	**93**	36	**96**	**94**	83	**96**	26	–	**96**
	task4	60	0	3	**90**	37	14	0	–	75
	task5	**98**	90	**99**	**98**	**96**	**99**	62	–	**98**
humanoidmaze-large	task1	36	14	6	–	36	8	–	–	**69**
	task2	0	0	0	–	1	0	–	–	**30**
	task3	55	1	16	–	32	19	–	–	**87**
	task4	2	0	16	–	10	9	–	–	**51**
	task5	29	2	19	–	7	9	–	–	**47**
cube-double	task1	16	**90**	85	51	29	81	81	84	**86**
	task2	12	**87**	79	28	6	46	77	59	68
	task3	10	**85**	54	22	2	42	78	40	65
	task4	4	**32**	22	4	1	10	9	5	**31**
	task5	11	76	**82**	48	4	50	74	43	76
cube-triple	task1	2	7	14	–	4	15	–	–	**33**
	task2	0	0	1	–	0	0	–	–	**4**
	task3	0	0	2	–	0	1	–	–	**8**
	task4	0	0	0	–	0	0	–	–	**3**
	task5	0	0	0	–	0	0	–	–	**1**
cube-quadruple-100m	task1	8	9	13	–	**35**	11	–	–	30
	task2	0	1	0	–	**6**	0	–	–	5
	task3	2	1	1	–	3	0	–	–	**7**
	task4	0	0	0	–	0	0	–	–	0
	task5	0	0	0	–	0	0	–	–	0
scene	task1	–	–	–	–	–	–	–	**100**	**100**
	task2	–	–	–	–	–	–	–	**96**	**96**
	task3	–	–	–	–	–	–	–	**93**	**95**
	task4	–	–	–	–	–	–	–	0	**97**
	task5	–	–	–	–	–	–	–	0	**90**
puzzle-4x4	task1	–	–	–	13	–	–	65	**83**	66
	task2	–	–	–	11	–	–	5	21	**31**
	task3	–	–	–	6	–	–	51	**81**	50
	task4	–	–	–	4	–	–	18	12	**28**
	task5	–	–	–	3	–	–	2	13	**31**

Our baseline experiments use settings that may differ from those in the original papers, including the discount factor and critic ensemble size. To complement these experiments, Table 13 compares NQA with published offline success rates using the same task variants, offline dataset sizes, and training budget of 1M gradient steps. The published results retain the discount factors and other hyperparameters used in their source papers. NQA achieves a higher average success rate than each baseline over the tasks for which that baseline reports comparable results.