GUIDED FLOW POLICY: LEARNING FROM HIGH-VALUE ACTIONS IN OFFLINE REINFORCEMENT LEARNING

Franki Nguimatsia Tiofack[1,*] **Théotime Le Hellard**[1,*] **Fabian Schramm**[1,*]
Nicolas Perrin-Gilbert[2] **Justin Carpentier**[1]

[1]Inria and DI-ENS, PSL Research University [2]Sorbonne Université, CNRS, ISIR, France
[*]Equal contribution. Correspondence to: franki.nguimatsia-tiofack@inria.fr

ABSTRACT

Offline reinforcement learning often relies on behavior regularization that enforces policies to remain close to the dataset distribution. However, such approaches fail to distinguish between high-value and low-value actions in their regularization components. We introduce Guided Flow Policy (GFP), which couples a multi-step flow-matching policy with a distilled one-step actor. The actor directs the flow policy through weighted behavior cloning to focus on cloning high-value actions from the dataset rather than indiscriminately imitating all state-action pairs. In turn, the flow policy constrains the actor to remain aligned with the dataset's best transitions while maximizing the critic. This mutual guidance enables GFP to achieve state-of-the-art performance across 144 state and pixel-based tasks from the OGBench, Minari, and D4RL benchmarks, with substantial gains on suboptimal datasets and challenging tasks.
Webpage: simple-robotics.github.io/publications/guided-flow-policy

1 INTRODUCTION

Offline Reinforcement Learning (RL) aims to learn effective policies from static datasets without further interaction with the environment (S. Lange, 2012; Ernst et al., 2005). This paradigm is essential in domains such as robotics and logistics, where online exploration can be unsafe or costly. However, standard off-policy algorithms such as DDPG (Lillicrap et al., 2015) and SAC (Haarnoja et al., 2018), which are successful in online RL, tend to underperform in offline settings since the RL agent cannot interact with the environment. The main challenge is extrapolation error, corresponding to the inability to properly evaluate out-of-distribution actions (Wu et al., 2019; Fujimoto et al., 2019; Kumar et al., 2019; 2020).

Two main lines of work have been proposed to address this challenge. The first one focuses on learning a critic without querying the values of actions outside the dataset (Kostrikov et al., 2021; Nair et al., 2020). The second one, known as the Behavior-Regularized Actor–Critic (BRAC) family[1], mitigates these errors by forcing the learned policy to stay "close" to the unknown behavior policy that generated the dataset (Fujimoto & Gu, 2021; Tarasov et al., 2023; Jaques et al., 2019; Laroche et al., 2019; Wu et al., 2019). The key idea is that out-of-distribution state–action pairs are especially vulnerable to Q-value overestimation, while staying near the empirical distribution reduces extrapolation errors. Minimalist variants achieve this by simply adding a behavior cloning (BC) loss to the policy and/or value updates with respect to dataset actions (Fujimoto & Gu, 2021; Tarasov et al., 2023). Although this approach improves stability, it also raises a trade-off: regularizing too strictly to a potentially suboptimal dataset action may restrict the policy from exploiting higher-reward actions contained in the dataset.

Until recently, most offline RL algorithms were based on Gaussian policies, with limited modeling capacities. Recent development of flow and diffusion-based expressive models (Lipman et al., 2022; Ho et al., 2020; Song et al., 2020), led to new RL algorithms to capture complex and multimodal

[1]The abbreviation BRAC refers to the family of methods, not solely the BRAC paper (Wu et al., 2019).

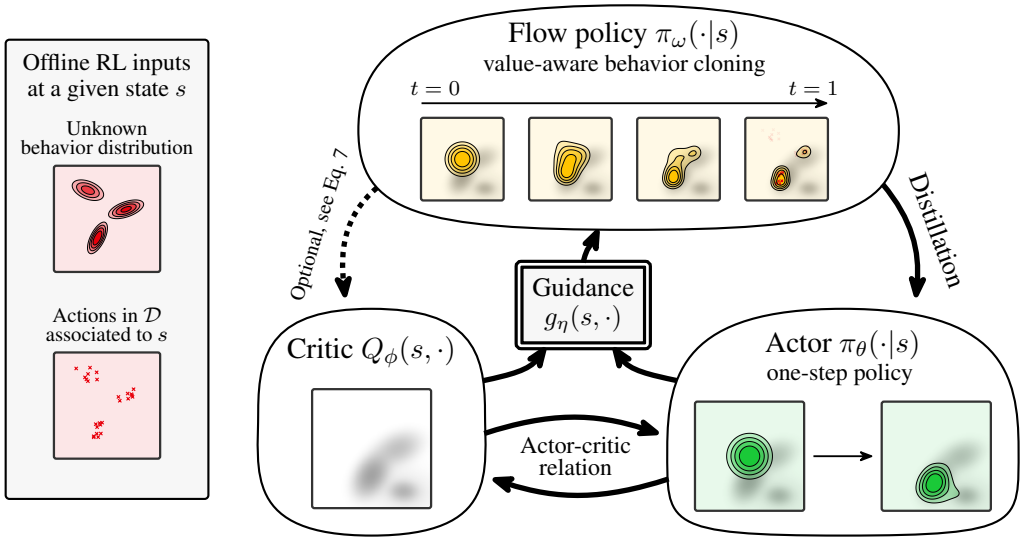


Figure 1: **Overview of the Guided Flow Policy framework.** GFP consists of three main components: (i) in yellow, VaBC, a multi-step flow policy π_ω trained via weighted BC using the guidance term g_η, (ii) in green, a one-step actor π_θ distilled from the flow policy, and (iii) in gray, a critic Q_ϕ guiding action evaluation. π_ω regularizes the actor toward high-value actions from the dataset $\mathcal{D}$; in turn, the actor shapes the flow and optimizes the critic following the actor–critic approach. The different components of the figure are introduced throughout the paper. Each drawing represents the probability distribution of actions $a \in \mathcal{A}$ of a policy, in a current state s, except for the gray ones, where it is the value of actions $a \in \mathcal{A}$ in state s, according to the critic.

action distributions (Chi et al., 2023; Janner et al., 2022; Wang et al., 2022; Zhang et al., 2025). However, they come at the risk of high computational overhead: iterative sampling slows inference, and directly optimizing the values of output actions would result in unstable backpropagation through time (BPTT). Among recent approaches to address these challenges, Park et al. (2025) proposed a BRAC method with a flow-matching BC model distilled into a one-step policy that also optimizes the critic, enabling expressive policy learning while avoiding BPTT and iterative sampling at inference. Despite these advances, a central limitation remains: the flow-based BC component, similar to standard BC, does not incorporate reward information.

We propose **Guided Flow Policy (GFP)**, a dual-policy BRAC framework with a bidirectional guidance mechanism between a multi-step flow-matching policy, termed Value-aware Behavior Cloning (VaBC), and a distilled one-step actor. VaBC acts as a distributional regularizer for the actor, encouraging it to remain within the support of the behavior policy. VaBC is trained via a weighted-BC mechanism, close to Peng et al. (2019); Zhang et al. (2025), but leveraging the actor and the critic to prioritize cloning high-value actions from the dataset. Unlike previous BRAC approaches, in which the BC regularization indiscriminately treats all state-action pairs (Fujimoto & Gu, 2021; Park et al., 2025), VaBC is a value-aware regularizer integrated into a BRAC approach. In turn, the actor optimizes the critic while being distilled toward VaBC, allowing it to align with the dataset's high-value actions in a given state while maximizing expected returns. Fig. 1 illustrates the GFP framework and Tab. 1 shows how it differs in the regularization mechanisms compared to other BRAC methods.

Our contributions are threefold: (i) we introduce Guided Flow Policy, a simple yet effective BRAC method that integrates value-awareness in the regularization term via a jointly trained weighted flow BC policy, thereby regularizing the actor with the most promising transitions of the dataset; (ii) we extensively evaluate GFP on 144 tasks from standard offline RL benchmarks, showing strong performances with substantial gains on suboptimal datasets and challenging tasks compared to prior works; (iii) we re-assess two previous state-of-the-art offline RL algorithms on these benchmarks, highlighting the critical role of hyperparameter choices and subtle implementation details, aligned in the spirit with the retrospective analysis provided in Tarasov et al. (2023).

2 Background

Actor-critic framework in RL. RL problems are typically formalized as a Markov Decision Process (MDP) (Sutton et al., 1998; Konda & Tsitsiklis, 1999), defined by the tuple $(\mathcal{S}, \mathcal{A}, p, r, \rho, \gamma)$. Here, $\mathcal{S}$ denotes the state space, $\mathcal{A}$ the action space, p the transition dynamics, r the reward function, ρ the initial state distribution, and $\gamma \in [0, 1)$ the discount factor. The behavior of the agent is

Table 1: **Overview of regularization mechanisms within the BRAC framework.**

	Regularization target	Value-aware regularization	Expressive variant
TD3+BC (Fujimoto & Gu, 2021) ReBRAC (Tarasov et al., 2023)	↪ dataset actions	✗	Diffusion-QL (Wang et al., 2022)
FQL (Park et al., 2025)	↪ learned behavior cloning policy	✗	✓
GFP (ours)	↪ learned value-aware behavior cloning policy	✓	✓

governed by a policy π, mapping states to probability distributions over actions. The objective is to maximize the expected discounted return $\mathbb{E}_{a_t\sim\pi(\cdot|s_t)}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)\right]$, i.e., the expected cumulative reward when following π in the MDP. In actor-critic approaches, the policy π, also referred to as the actor, is trained jointly with a critic Q, which approximates the state–action value function. The Q-function is defined as $Q^\pi(s,a)=\mathbb{E}_{a\sim\pi(\cdot|s)}\left[\sum_{t=0}^{\infty}\gamma^t r\left(s_t,a_t\right)\mid s_0=s,a_0=a\right]$, estimating the expected return after taking action a in state s and subsequently following π.

Both actor and critic are parametrized as neural networks, with parameters θ and ϕ respectively, and optimized by alternating gradient descent steps on the two objectives:

$$\mathcal{L}^{\mathcal{A}}(\theta)=\mathbb{E}_{s\sim\mathcal{D},\,a_\theta\sim\pi_\theta(\cdot|s)}\big[-Q_\phi(s,a_\theta)\big], \tag{1}$$

$$\mathcal{L}^{\mathcal{C}}(\phi)=\mathbb{E}_{(s,a,r,s')\sim\mathcal{D},\,a'\sim\pi_\theta(\cdot|s')}\left[\left(Q_\phi(s,a)-r-\gamma Q_{\bar{\phi}}(s',a')\right)^2\right], \tag{2}$$

where $\mathcal{L}^{\mathcal{A}}$ and $\mathcal{L}^{\mathcal{C}}$ refer to the actor and critic losses, respectively, and $\mathcal{D}$ is the set of transitions (s,a,r,s') collected during training. $Q_{\bar{\phi}}$ denotes a second target Q-function parameterized by a slowly updated set of weights $\bar{\phi}$, maintained via Polyak averaging, a common stabilization technique in actor–critic methods.

Minimalist approaches in offline RL. In offline RL, the agent learns exclusively from a static dataset $\mathcal{D}$, consisting of transitions (s,a,r,s') generated by an unknown behavior policy. In a given state s, the distribution of actions of such a behavior policy is illustrated on the left of Fig. 1. This introduces a key challenge compared to the online setting: the distributional shift (S. Lange, 2012; Kumar et al., 2019; Konda & Tsitsiklis, 1999). Indeed, since the learned policy π_θ may select actions outside the dataset's support, value estimates for such out-of-distribution actions can be inaccurate (Kumar et al., 2020; Fujimoto & Gu, 2021). The BRAC approach addresses this issue by constraining the policy π_θ to remain close to the behavior policy (Jaques et al., 2019; Kumar et al., 2019). As emphasized in Fujimoto & Gu (2021), a simple and effective choice is to add a BC term directly into the actor objective. Incorporating this into the actor–critic framework, the actor loss in Eq. 1 becomes:

$$\mathcal{L}^{\mathcal{A}}(\theta)=\mathbb{E}_{(s,a)\sim\mathcal{D},a_\theta\sim\pi_\theta(\cdot|s)}\Big[-Q_\phi(s,a_\theta)+\alpha\overbrace{\|a_\theta-a\|^2}^{\text{BC term}}\Big], \tag{3}$$

where α is a hyperparameter that balances between exploiting high Q-values and staying close to the behavior policy. This objective encourages actions that both achieve high-expected returns and remain within the support of the dataset. The critic loss in Eq. 2 remains unchanged.

Behavior cloning with flow matching. Flow Matching (FM) (Lipman et al., 2022) is a generative modeling framework that learns a continuous-time transformation, or flow, which maps a simple base distribution (in this work, a standard Gaussian) to a target data distribution. This transformation is defined through a family of intermediate, time-dependent distributions governed by an ordinary differential equation (ODE).

In the context of BC, FM is extended to a conditional setting, where the goal is to approximate a behavior policy π_ω underlying the dataset $\mathcal{D}$. This is achieved by learning a state and time dependent velocity field $v_\omega:[0,1]\times\mathcal{S}\times\mathbb{R}^d\to\mathbb{R}^d$ that governs the dynamics of a flow, where d is the action dimension. This flow $\psi_\omega(t,s,z)$ is the solution of the family of ODEs characterized by:

$$\forall s\in\mathcal{S},\quad \frac{d}{dt}\psi_\omega(t,s,z)=v_\omega(t,s,\psi_\omega(t,s,z)),\quad \psi_\omega(0,s,z)=z. \tag{4}$$

This flow, conditioned on the state s, maps noise samples $z\sim\mathcal{N}(0,I_d)$ into actions distributed according to $\pi_\omega(\cdot\mid s)$. While sophisticated conditioning strategies can help enhance expressiveness

Algorithm 1: Guided Flow Policy (GFP)

function ***Integrate*** $\mu_\omega(s, z)$
  // Explicit discrete Euler integration with M steps
  for $t = 0, 1, \ldots, M-1$ **do**
    $z \leftarrow z + \frac{1}{M} v_\omega(t/M, s, z)$
  return z
while *not converged* **do**
  Sample $\{(s, a, r, s')\} \sim \mathcal{D}$
  // Step 1 -- Train critic Q_ϕ
  $z' \sim \mathcal{N}(0, I_d)$, $a' = \mu_{\bar{\theta}}(s', z')$
  Update ϕ to minimize $\mathbb{E}[(Q_\phi(s, a) - r - \gamma Q_{\bar{\phi}}(s', a'))^2]$
  // Step 2 -- Train the distilled one-step actor π_θ
  $z \sim \mathcal{N}(0, I_d)$, $a^{\pi_\theta} = \mu_\theta(s, z)$,
  $a^{\pi_\omega} = \mu_\omega(s, z)$ // Using the Integrate-μ_ω function, Line. 1
  Compute $\lambda = \frac{1}{\frac{1}{N}\sum |Q_\phi(s, a^{\pi_\theta})|}$ // Stop gradient a^{π_θ}
  Update θ to minimize $\mathbb{E}[-\lambda Q_\phi(s, a^{\pi_\theta}) + \alpha \|a^{\pi_\theta} - a^{\pi_\omega}\|_2^2]$
  // Step 3 -- Train the value-aware BC policy π_ω
  Compute $g_\eta(s, a) = \frac{\exp\left(\frac{\lambda}{\eta} Q_\phi(s, a)\right)}{\exp\left(\frac{\lambda}{\eta} Q_\phi(s, a^{\pi_\theta})\right) + \exp\left(\frac{\lambda}{\eta} Q_\phi(s, a)\right)}$ // Stop gradient a^{π_θ}
  $a_t = (1-t)\epsilon + ta$, with $\epsilon \sim \mathcal{N}(0, I_d)$ and $t \sim \mathcal{U}([0, 1))$
  Update ω to minimize $\mathbb{E}\left[g_\eta(s, a) \|v_\omega(t, s, a_t) - (a - \epsilon)\|_2^2\right]$
Output: π_θ, π_ω, Q_ϕ

(e.g., classifier-free guidance (Ho & Salimans, 2022)), we adopt in this work the simplest variant of conditional flow matching (Holderrieth & Erives, 2025). We further employ the optimal transport variant of FM, which uses linear interpolation with uniformly sampled time points (Lipman et al., 2022). For $(s, a) \sim \mathcal{D}$, $\epsilon \sim \mathcal{N}(0, I_d)$, and $t \sim \mathcal{U}([0, 1])$, we define the interpolated point $a_t = (1-t)\epsilon + ta$, whose target velocity is $a - \epsilon$. The velocity field v_θ is then trained by least-squares regression toward this reference, yielding the conditional flow-matching BC loss (Holderrieth & Erives, 2025):

$$\mathcal{L}^{\text{FM-BC}}(\omega) = \mathbb{E}_{(s,a)\sim\mathcal{D}, \epsilon\sim\mathcal{N}(0,I_d), t\sim\mathcal{U}([0,1])}\left[\|v_\omega(t, s, a_t) - (a - \epsilon)\|_2^2\right]. \tag{5}$$

Once the velocity field is learned, the corresponding flow $\psi_\omega : [0, 1] \times \mathcal{S} \times \mathbb{R}^d \to \mathcal{A}$ defines an approximation of the behavior policy. At inference, an action is obtained by sampling a random noise $z \sim \mathcal{N}(0, I_d)$ and integrating the flow from 0 to 1 using an ODE solver (e.g., an explicit Euler method). We denote by $\mu_\omega(s, z) := \psi_\omega(1, s, z)$ the value of the integrated flow at time 1. In this way, behavior cloning can be naturally expressed as conditional flow matching in the action space.

Flow policy for offline RL. Following the idea of Diffusion Q-Learning by Wang et al. (2022), a straightforward way to train a flow policy for offline RL is to replace the BC term in the actor loss (Eq. 3) with the flow-matching BC loss (Eq. 5). However, the iterative sampling procedure makes training expensive, due to recursive backpropagation through time (BPTT) in the actor loss, and also results in a slow inference at test time. To mitigate these limitations, Park et al. (2025) suggests distilling the iterative flow-matching BC policy into a one-step policy that also maximizes the critic.

3 Guided Flow Policy

We now detail the GFP algorithm that builds on top of Fujimoto & Gu (2021); Park et al. (2025). GFP integrates a Value-aware Behavior Cloning (VaBC) flow policy with a distilled one-step actor through bidirectional guidance. VaBC leverages the actor and the critic to selectively clone high-value dataset actions, providing more targeted regularization than in standard BRAC approaches. The distilled actor, in turn, maximizes the critic while avoiding BPTT and iterative sampling. GFP is composed of three main components: the critic Q_ϕ, the actor π_θ, and the VaBC policy π_ω. The complete algorithm is presented in Algo. 1, and the approach is illustrated in Fig. 1.

Step 1 – Learning the critic Q_ϕ. The critic is trained using the Bellman mean-squared loss:

$$\mathcal{L}^{\mathcal{C}}(\phi) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D},\, a'\sim\pi_\theta(\cdot|s')}\Big[\big(Q_\phi(s, a) - \underbrace{(r + \gamma Q_{\bar{\phi}}(s', a'))}_{\text{Bellman target } y}\big)^2\Big], \tag{6}$$

where $Q_{\bar{\phi}}$ denotes the target network. $y(s, r, s') := r + \gamma Q_{\bar{\phi}}(s', a')$ corresponds to the standard Bellman target in actor-critic methods, which we use by default in this work. Yet, since VaBC is designed to prioritize cloning the most promising dataset actions for a given state, we have also

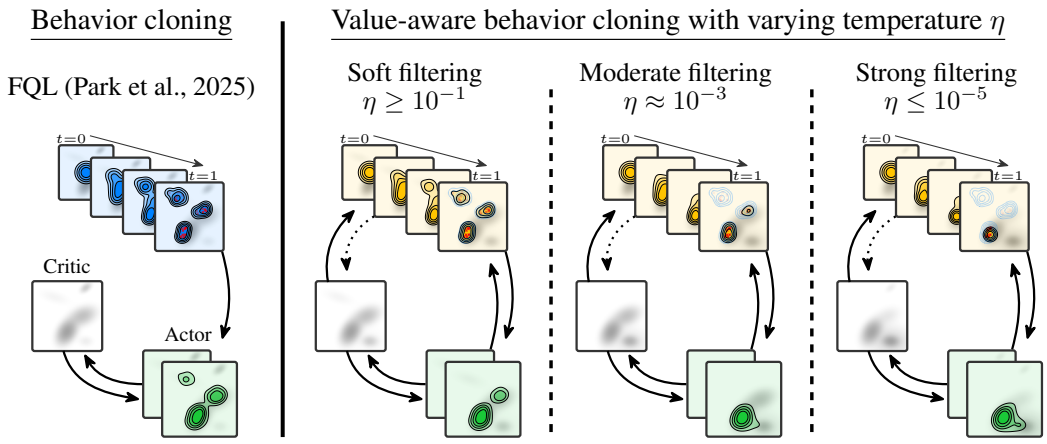


Figure 2: **Comparison of behavior cloning under different levels of guidance. Left:** Prior work (e.g., FQL by Park et al. (2025)) uses no filtering, indiscriminately imitating all state-action pairs. **Right:** In contrast, our method introduces a temperature-controlled guidance mechanism, as shown in Eq. 10, resulting in VaBC. At high temperatures, the guidance is weak, so the actor is influenced by many candidate actions. At moderate temperatures, the filtering becomes sharper, giving more weight to higher-value actions while still keeping enough regularization and exploration. At low temperatures, the filtering is very selective, concentrating almost exclusively on the highest-value actions according to the critic. However, excessive concentration at very low temperatures may allow the actor to escape the dataset's action distribution, as shown on the right in green, leading to critic overestimation and out-of-distribution issues. Importantly, VaBC cannot escape the dataset's action distribution even at very low temperatures, since it trains exclusively on in-distribution state-action pairs. The dashed blue contours in the final yellow drawings (first row) illustrate this constraint.

considered a more conservative variant of the Bellman target:

$$y^{\text{VaBC}}(s,r,s') = r + \frac{\gamma}{2}\Big(Q_{\bar{\phi}}(s', \mu_\theta(s',z)) + Q_{\bar{\phi}}(s', \mu_\omega(s',z))\Big), \quad z \sim \mathcal{N}(0, I_d), \tag{7}$$

where $\mu_\theta(s',z)$ denotes the action from the actor and $\mu_\omega(s',z)$ the action from the VaBC policy. Here, as mentioned in Sec. 2 and outlined in Line 1 of Alg. 1, $\mu_\theta(s,z)$ and $\mu_\omega(s,z)$ are actions sampled from $\pi_\theta(\cdot|s)$ and $\pi_\omega(\cdot|s)$, respectively, with initial input noise z. The Bellman target $y^{\text{VaBC}}(s,r,s')$ corresponds to an averaging between two estimates of the Q-value: $Q_{\bar{\phi}}(s', \mu_\theta(s',z))$ which can overestimate the real Q-value; and $Q_{\bar{\phi}}(s', \mu_\omega(s',z))$ which can underestimate the real Q-value. This choice can lead to substantial performance improvements in certain situations, as studied in the appendix, Sec. B.1.

Step 2 – Learning the actor π_θ. The actor π_θ is trained by behavior regularized policy gradients, to maximize the Q-function while distilling the distribution of valuable actions learned by π_ω. This is achieved by minimizing the following objective:

$$\mathcal{L}^{\mathcal{A}}(\theta) = \mathbb{E}_{s\sim\mathcal{D}, z\sim\mathcal{N}(0,I_d)}\Big[-\lambda Q_\phi\left(s, \mu_\theta(s,z)\right) + \alpha \|\mu_\theta(s,z) - \mu_\omega(s,z)\|_2^2\Big]. \tag{8}$$

The normalization term $\lambda = \frac{1}{\frac{1}{N}\sum |Q_\phi(s,a)|}$ is based on the average absolute Q-value, estimated over mini-batches rather than over the entire dataset (Fujimoto & Gu, 2021).

The distillation term encourages the actor to stay close to VaBC. In this way, the actor learns to select actions that maximize return while avoiding out-of-distribution actions, as it is constrained to remain near the support of high-value dataset behaviors.

Step 3 – Learning the flow policy π_ω. The VaBC policy π_ω is optimized via a weighted flow-matching behavior cloning:

$$\mathcal{L}^{\text{VaBC}}(\omega) = \mathbb{E}_{(s,a)\sim\mathcal{D}, \epsilon\sim\mathcal{N}(0,I_d), t\sim\mathcal{U}([0,1])}\Big[g_\eta(s,a)\, \|v_\omega(t,s,a_t) - (a-\epsilon)\|_2^2\Big], \tag{9}$$

where

$$g_\eta(s,a) := \frac{\exp\left(\frac{\lambda}{\eta} Q_\phi(s,a)\right)}{\exp\left(\frac{\lambda}{\eta} Q_\phi(s,a)\right) + \exp\left(\frac{\lambda}{\eta} Q_\phi(s, \mu_\theta(s,z))\right)}, \quad z \sim \mathcal{N}(0, I_d). \tag{10}$$

Intuitively, for a given state-action pair (s, a) sampled from the dataset $\mathcal{D}$, $g_\eta(s, a)$ compares the quality between the dataset action a and a proposal of the actor $\mu_\theta(s, z)$ in a soft-max approach. If the dataset action has a higher Q-value, this implies that $g_\eta(s, a) > 0.5$, placing greater emphasis on cloning it. Conversely, if the dataset action is worse, $g_\eta(s, a) < 0.5$, then it reduces its influence. This ensures that VaBC selectively clones high-value dataset behaviors. This makes sense because the actor itself is constrained to remain close to the dataset's action distribution. Importantly, VaBC is learned jointly with the actor and the critic, not beforehand.

Here, λ is the same Q-normalization factor used in the actor loss, ensuring consistent scaling across components. The parameter $\eta > 0$ is a temperature hyperparameter that controls the sharpness of the weighting: small η makes $g_\eta(s, a)$ more selective, while large η smooths the weighting. Importantly, since $g_\eta(s, a) \in (0, 1)$, VaBC avoids degeneracy during early training when the critic is unreliable, ensuring stable learning.

The key contribution of GFP is to add value-awareness in the behavior regularization component of a BRAC framework, effectively combining the two predominant policy extraction methods studied by Park et al. (2024b): weighted-BC and behavior regularized policy gradients, further presented in Sec. 5. One could employ an advantage weighted term, similar to AWR (Peng et al., 2019), using the actor to compute a baseline:

$$g_\eta^{\text{AWR}}(s, a) := \exp\left(\frac{\lambda}{\eta}\big(Q_\phi(s, a) - Q_\phi(s, \mu_\theta(s, z))\big)\right) \tag{11}$$

As recommended by Peng et al. (2019), in that case, a clipping term should be added for stability. Instead, we propose the guiding function g_η, which corresponds to a soft-max between $Q_\phi(s, a)$ and $Q_\phi(s, \mu_\theta(s, z))$. In appendix, Sec. B.3, Tab. 7 reports results when using g_η^{AWR} instead of g_η, over 65 tasks.

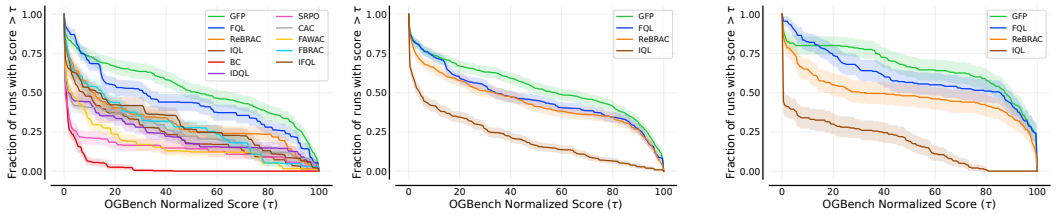
Analysis of the temperature in the guidance function g_η. In Fig. 2, we illustrate how the temperature parameter controls value-guided filtering, balancing dataset fidelity with value exploitation. Lower temperatures sharpen the filter, shifting the policy from broadly imitating the dataset to emphasizing higher-value actions. Moderate values achieve the best trade-off, prioritizing promising actions while preserving diversity. In contrast, excessively low temperatures over-concentrate the VaBC policy, destabilizing training and degrading the critic by pushing the actor out of distribution.

4 Experiments

4.1 Main results: extensive offline RL benchmarks

Benchmarks. We conducted extensive experiments over a suite of robot locomotion and manipulation tasks, spanning three major benchmarks: D4RL (Fu et al., 2020), its successor Minari (Younis et al., 2024), and the recently proposed OGBench (Park et al., 2024a). For comparability with existing works, we first evaluate on D4RL's AntMaze (6 tasks) and Adroit (12 tasks). We also present results on Minari, evaluating both GFP and FQL, to facilitate the community's migration from D4RL. Minari includes all available Gym-Mujoco datasets (Hopper, HalfCheetah, and Walker, each with 3 tasks), as well as Adroit (12 tasks). Our most extensive evaluation focuses on OGBench, which offers substantially more complex and challenging tasks than D4RL. Following Park et al. (2025), we use the reward-based single-task variants of OGBench. We evaluate across 9 locomotion and 11 manipulation environments, each with 5 tasks, for a total of 100 state-based tasks, and we also consider 5 pixel-based tasks. Combined with D4RL and Minari, we benchmark GFP on 144 tasks overall, including evaluation of prior methods, we did about 15 000 runs in total. Our JAX-based implementation of GFP will be released after the rebuttal phase. It can complete one training run in under 30 minutes on modern GPUs.

Comparison against previous works. We first compare GFP against 10 prior methods, including standard offline RL, flow, and diffusion-based methods on the 50 OGBench tasks reported by Park et al. (2025) (see Sec. 5 and Fig 5 for a presentation of prior works, Tab. 10 for the full results). Performance profiles synthesizing these comparisons are presented Fig. 3a, following Agarwal et al. (2021). GFP clearly stands out compared to all these prior works. We also compare against 6 methods on D4RL (Tab. 11) and evaluate both GFP and FQL on Minari (see Tab. 12) to facilitate the community's migration from D4RL.



(a) (b) (c)

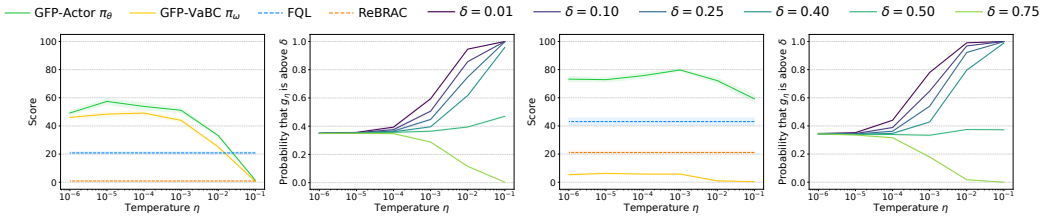
Figure 3: **OGBench analysis.** (a) Performance profiles for 50 tasks comparing GFP against a wide range of prior works, showing the fraction of tasks where each algorithm achieves a score above threshold τ, using the evaluation reported by Park et al. (2025). (b) Performance profiles on 105 tasks, including more challenging ones, and carefully reevaluated prior methods. (c) Performance profiles restricted to 30 noisy and explore tasks.

Extensive study on 144 tasks. GFP is further evaluated against 3 baselines: (i) FQL (Park et al., 2025), as it comes second only to GFP on the first 50 tasks; (ii) ReBRAC (Tarasov et al., 2023), as we were able to improve its performance compared to previously reported results substantially, see Sec. 4.3, and (iii) IQL (Kostrikov et al., 2021) to represent in-sampling approaches. Tab. 2 summarizes our results grouped by environment, with detailed per-task results in the appendix (Tabs. 8, 9, 11, and 12). Together with performance profile plots Figs. 3b and 3c, it confirms that GFP achieves state-of-the-art performance, with particularly substantial gains on noisy and challenging environments. For instance, on the cube-double-noisy and cube-triple-noisy datasets, GFP achieves an average score of 63 and 24, respectively, compared to 38 and 4 for FQL, and 20 and 5 for ReBRAC. Similarly, GFP stands out in some very challenging locomotion tasks, such as humanoidmaze-large-navigate (18 vs. 7 for FQL and 13 for ReBRAC), and manipulation tasks, such as cube-triple-play (16 vs. 4 for FQL and 3 for ReBRAC).

Table 2: **Offline RL results.** GFP achieves best or near-best performance on all 144 benchmark tasks. Results are averaged over 8 seeds for state-based tasks and 4 seeds for pixel-based ones, with values reported from prior works (Park et al., 2025; Tarasov et al., 2023; Fu et al., 2020) in *italic*, and values within 95% of the best performance are shown in bold. GFP actor π_θ represents our primary policy, while GFP VaBC π_ω is reported as a byproduct of the training procedure. Full per-task results are provided in the appendix Tabs. 8, 9, 11, and 12, and the comparison with 10 methods on the 50 previously evaluated OGBench tasks Tab. 10.

Task Category	**Offline RL algorithms**				
	IQL	ReBRAC	FQL	GFP actor π_θ	GFP VaBC π_ω
OGBench antmaze-large-navigate-singletask (5 tasks)	*53* ± 3	**95.9** ± 0.4	88.1 ± 3.4	**93.8** ± 1.5	90.0 ± 1.3
OGBench antmaze-large-stitch-singletask (5 tasks)	30.4 ± 3.2	**89.2** ± 6.6	58.1 ± 8.7	68.9 ± 0.8	57.6 ± 3.2
OGBench antmaze-large-explore-singletask (5 tasks)	12.9 ± 1.7	82.7 ± 7.6	**87.9** ± 6.6	**91.9** ± 0.9	**89.3** ± 1.1
OGBench antmaze-giant-navigate-singletask (5 tasks)	*4* ± 1	**33.2** ± 5.7	16.3 ± 8.2	27.9 ± 8.5	0.8 ± 0.2
OGBench humanoidmaze-medium-navigate-singletask (5 tasks)	*33* ± 2	59.2 ± 12.1	*58* ± 5	**72.0** ± 2.8	35.9 ± 2.7
OGBench humanoidmaze-medium-stitch-singletask (5 tasks)	27.3 ± 2.9	61.1 ± 8.2	**63.2** ± 6.7	**66.2** ± 5.7	39.5 ± 2.1
OGBench humanoidmaze-large-navigate-singletask (5 tasks)	*2* ± 1	12.9 ± 4.2	6.5 ± 2.7	**17.8** ± 9.6	2.4 ± 1.1
OGBench antsoccer-arena-navigate-singletask (5 tasks)	*8* ± 2	55.9 ± 1.5	**60** ± 4	**57.9** ± 1.9	10.3 ± 0.7
OGBench antsoccer-arena-stitch-singletask (5 tasks)	2.8 ± 1.0	22.0 ± 1.5	28.6 ± 2.3	**30.5** ± 2.2	1.4 ± 0.3
OGBench cube-single-play-singletask (5 tasks)	*83* ± 3	*91* ± 2	***96*** ± 1	**98.8** ± 0.4	39.7 ± 4.1
OGBench cube-single-noisy-singletask (5 tasks)	53.2 ± 4.1	**98.4** ± 0.6	**100.0** ± 0.0	**100.0** ± 0.0	**99.9** ± 0.1
OGBench cube-double-play-singletask (5 tasks)	*7* ± 1	12.6 ± 1.8	*29* ± 2	**47.2** ± 1.6	6.4 ± 1.0
OGBench cube-double-noisy-singletask (5 tasks)	4.5 ± 0.8	19.6 ± 2.1	38.2 ± 5.3	**63.1** ± 3.3	9.4 ± 0.8
OGBench cube-triple-play-singletask (5 tasks)	0.1 ± 0.1	2.9 ± 1.2	3.9 ± 1.5	**15.9** ± 2.0	7.6 ± 1.6
OGBench cube-triple-noisy-singletask (5 tasks)	4.8 ± 1.2	5.2 ± 2.9	3.5 ± 1.6	**24.5** ± 2.8	8.6 ± 1.2
OGBench puzzle-3×3-play-singletask (5 tasks)	*9* ± 1	21 ± 1	***30*** ± 1	23.1 ± 2.2	19.2 ± 2.9
OGBench puzzle-4×4-play-singletask (5 tasks)	*7* ± 1	17.1 ± 1.3	*17* ± 2	**26.1** ± 2.1	9.5 ± 1.1
OGBench puzzle-4×4-noisy-singletask (5 tasks)	0.1 ± 0.0	1.1 ± 0.3	15.6 ± 1.1	**18.8** ± 1.7	**19.3** ± 1.0
OGBench scene-play-singletask (5 tasks)	*28* ± 1	41.6 ± 3.6	***56*** ± 2	53.5 ± 2.9	**57.6** ± 1.7
OGBench scene-noisy-singletask (5 tasks)	16.0 ± 1.2	39.9 ± 2.6	**59.3** ± 1.4	**57.5** ± 0.9	**58.5** ± 1.0
OGBench visual manipulation (5 tasks)	*42* ± 4	*60* ± 2	***65*** ± 2	**62.8** ± 1.5	–
D4RL antmaze (6 tasks)	*17*	*76.8*	***84*** ± 3	**83.1** ± 2.7	70.2 ± 3.0
D4RL Adroit (12 tasks)	*48*	*59*	*52* ± 1	52.8 ± 1.4	49.6 ± 1.3
Minari Adroit (12 tasks)	–	–	40.6 ± 0.4	**48.3** ± 2.3	**46.1** ± 1.7
Minari hopper (3 tasks)	–	–	79.6 ± 10.3	**91.7** ± 4.5	**91.5** ± 12
Minari halfcheetah (3 tasks)	–	–	97.8 ± 2.0	**109.1** ± 2.0	**103.1** ± 1.8
Minari walker2d (3 tasks)	–	–	**121.7** ± 1.3	**124.5** ± 0.8	**122.2** ± 1.1
Average OGBench (105 tasks)	20.4	43.9	46.7	**53.2**	33.1[2]
Average D4RL (18 tasks)	54.0	**64.8**	**62.1**	**63.0**	56.5
Average Minari (21 tasks)	–	–	65.9	**74.1**	**71.6**

[2] average without pixel-based tasks.



(a) Puzzle score (b) Puzzle $\mathbb{P}(g_\eta > \delta)$ (c) Cube score (d) Cube $\mathbb{P}(g_\eta > \delta)$

Figure 4: **Temperature analysis on challenging OGBench Puzzle (left) and Cube (right) tasks with suboptimal data. Plots (a) and (c):** performance scores across temperature values η for our GFP method (Actor π_θ and VaBC π_ω) compared to baselines (FQL, ReBRAC) on puzzle-4x4-noisy-task3 and cube-double-noisy-task2. **Plots (b) and (d):** probability that the guidance term g_η is above different thresholds δ as a function of temperature, illustrating how temperature controls the sharpness of value-guided filtering.

Value-aware behavior cloning. VaBC serves as the regularization component of GFP, encouraging the actor maximizing the value to stay close to some actions from the dataset. While the actor π_θ is GFP primary policy, through our bidirectional training procedure, we obtain the VaBC policy π_ω as a byproduct that can also be exploited and evaluated. As shown in Tab. 2, VaBC achieves good performance across benchmarks while being fundamentally in-sample, justifying its role as a value-aware regularizer for the main actor.

4.2 Analysis of the temperature parameter η

We investigate the impact of the temperature-controlled guidance mechanism. In Fig. 4, GFP is evaluated with varying temperature η on two challenging *noisy* tasks from OGBench, characterized as *highly suboptimal data* according to Park et al. (2024a). The presence of low-quality demonstrations makes selective action emphasis decisive for effective learning. Figs. 4a and 4c demonstrate the advantages of moderate temperatures. Very low temperatures cause training instability due to over-concentration on narrow action sets, while very high temperatures fail to provide sufficient filtering of suboptimal actions. As the temperature decreases, VaBC π_ω performance improves, confirming that the value-guided filtering mechanism successfully emphasizes higher-value actions, until the temperature is too low.

Figs. 4b and 4d illustrate the filtering behavior by showing the probability that the guidance term g_η exceeds various thresholds δ (ranging from 0.01 to 0.75). At extremely low temperatures ($\eta \leq 10^{-5}$), the guidance term exhibits near-binary behavior: any slight differences between $Q(s, a)$, with a from the dataset, and $Q(s, a^{\pi_\theta})$, with $a^{\pi_\theta} \sim \pi_\theta(\cdot|s)$, in state s, result in the guidance term approaching either 1 or 0 according to Eq. 10, in this case $\mathbb{P}(g_\eta > 0.75) \approx \mathbb{P}(g_\eta > 0.01)$. As the temperature increases, the filtering becomes softer, creating more gradual transitions in the guidance values. This leads to a broader distribution of filtering probabilities across different thresholds, demonstrating how higher temperatures preserve more of the original dataset diversity. In contrast, lower temperatures create sharper distinctions between high-value and low-value actions. In the appendix, in Sec. B.4, we extend this study to two additional tasks.

Sensitivity analysis to α and η. In Sec. B.5 we evaluate how sensitive GFP is to variations of α, the BC coefficient, and η, on 12 tasks, with comparisons to FQL. It reveals how they relate and demonstrates that GFP is only mostly sensitive to the proper choice of α, as for any BRAC method.

4.3 Re-evaluation of prior works on OGBench

To obtain a fair comparison of GFP against prior methods, we reevaluate existing baselines on OGBench. During the development of our method, we observed that several task-specific hyperparameters (e.g., discount factor γ, minibatch size B, and critic aggregation scheme for doubled Q-learning (Fujimoto et al., 2018)) have a significant impact on performance. Tab. 3 reports results under these revised settings, showing that careful tuning can substantially improve the reported scores of both ReBRAC and FQL. Since the optimal values for these hyperparameters were generally consistent across methods, we treated them as task-specific and, by default, applied the same settings to GFP, FQL, and ReBRAC (see Tabs. 4 and 5 in the appendix for detailed values).

Table 3: **Impact of task-specific hyperparameters on OGBench performance.**

Task Environment	Bigger discount factor γ			
	Previously reported in Park et al. (2025)		Our evaluations	
	ReBRAC	FQL	ReBRAC	FQL
antmaze-large-navigate (5 tasks)	*81* $\pm$ 5	*79* $\pm$ 3	95.9 $\pm$ 0.4	88.1 $\pm$ 3.4
humanoidmaze-large-navigate (5 tasks)	*2* $\pm$ 1	*4* $\pm$ 2	12.9 $\pm$ 4.2	6.5 $\pm$ 2.7
	Bigger minibatch size			
	Previously reported in Park et al. (2025)		Our evaluations	
	ReBRAC	FQL	ReBRAC	FQL
antmaze-giant-navigate (5 tasks)	*26* $\pm$ 8	*9* $\pm$ 6	33.2 $\pm$ 5.7	16.3 $\pm$ 8.2
	Same critic aggregation for ReBRAC as used in FQL			
	Previously reported in Park et al. (2025)		Our evaluations	
	ReBRAC		ReBRAC	
humanoidmaze-medium-navigate (5 tasks)	*22* $\pm$ 8		59.2 $\pm$ 12.1	
antsoccer-arena-navigate (5 tasks)	*0* $\pm$ 0		55.9 $\pm$ 1.5	
cube-double-play (5 tasks)	*12* $\pm$ 1		12.6 $\pm$ 1.8	
scene-play (5 tasks)	*41* $\pm$ 3		41.6 $\pm$ 3.6	
puzzle-4x4-play (5 tasks)	*14* $\pm$ 1		17.1 $\pm$ 1.3	

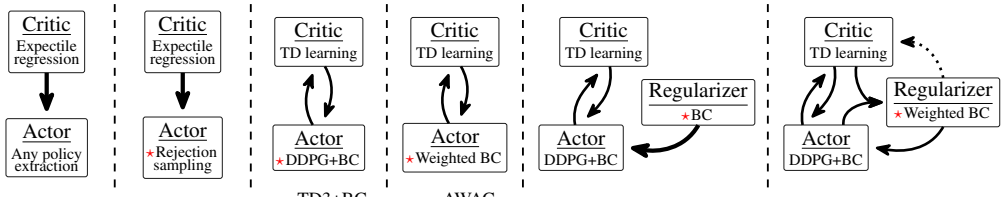
5 Related Work

Offline RL. Early methods addressed distributional shift by working on both how the critic is learned and how the policy is extracted. On the critic side, Conservative Q-Learning (CQL) (Kumar et al., 2020) penalizes out-of-distribution value estimates and Implicit Q-Learning (IQL) (Kostrikov et al., 2021) avoids querying such values via expectile regression, see Fig 5a. For policy extraction, weighted behavior cloning treats offline RL as supervised learning, typically using the advantage value as the weight, as in AWR or AWAC (Peng et al., 2019; Nair et al., 2020), see Fig 5d. The policy extraction method found to lead to the best performance and scalability in standard offline-RL is behavior-regularized policy gradient (Fujimoto & Gu, 2021; Park et al., 2024b), abbreviated DDPG+BC and also called reparametized policy gradients, which consists in directly maximizing the value function, while regularizing with a behavior cloning term. The BRAC family studied in this paper uses behavior-regularized policy gradient and trains the critic directly using the actor, see Fig 5c. ReBRAC (Tarasov et al., 2023) demonstrated state-of-the-art performance from this simple idea, through careful hyperparameter tuning and architectural choices.

Control with diffusion and flow models. Expressive generative models enable multi-modal action and trajectory distributions modeling and have been integrated into control along several axes. They have been used for trajectory and motion planning, trajectory optimization, hierarchical control (Janner et al., 2022; Ajay et al., 2022; Zheng et al., 2023; Liang et al., 2023; Li et al., 2023; Suh et al., 2023; Venkatraman et al., 2023; Lee et al., 2023; Ma et al., 2024; Chen et al., 2024a; Pan et al., 2024; Le Hellard et al., 2025), as well as for world modeling (Ding et al., 2024b; Alonso et al., 2024; Lee et al., 2024). In offline RL, numerous ideas emerged to integrate diffusion and flow models in the various policy extraction methods.

Weighted behavior cloning. A straightforward approach is to add weight in front of the cloning loss of diffusion and flow models, examples include FAWAC (Park et al., 2025), EDP (Kang et al., 2023), and QVPO (Ding et al., 2024a). Related energy-guided methods explicitly target to sample from $\pi_\theta(a \mid s) \propto \mu(a \mid s) \exp(\eta Q(s, a))$, where μ is the behaviour policy. This includes CEP (Lu et al., 2023) and QIPO (Zhang et al., 2025; Alles et al., 2025), see Fig 5d.

Behavior-regularized policy gradient. Combining the most effective policy extraction method with expressive models is a motivating research direction: Diffusion-QL (Wang et al., 2022) (Fig 5c), DiffCPS (He et al., 2023), trust-region formulations for diffusion policies (DTQL) (Chen et al., 2024b), entropy Q-ensemble regularization (Zhang et al., 2024), Consistency-AC with consistency models (Ding & Jin, 2023), and diffusion policy gradients (Li et al., 2024; Yang et al., 2023). In that case, the main challenge is the drawbacks of backpropagation through time (BPTT) when trying to directly optimize the value of the outputs of an iterative process. To avoid BPTT, FQL (Park et al., 2025) (Fig 5e) trains a flow matching with a standard BC loss, and then leverages it in the BC regularization term of a one-step BRAC actor. Beyond these families, other policy extractors include rejection sampling from a fixed BC generative policy: IDQL (Hansen-Estruch et al., 2023) (Fig 5b), DICE (Mao et al., 2024), and SfBC (Chen et al., 2022).



(a) IQL (b) ⋆IDQL (c) TD3+BC ReBRAC ⋆DQL (d) AWAC ⋆FAWAC ⋆QIPO (e) ⋆FQL (f) ⋆Our GFP

Figure 5: **Overview of some offline-RL frameworks.** The symbol ⋆ indicates the use of a diffusion or a flow model, and specifically in which component. Each box, provides the name of the component (e.g. critic) and the working principle that is used to train it (e.g. TD learning). The arrows indicate how the components depend on each other, while the dashed arrow is optional (see Eq. 7)

Positioning GFP within prior work. Our method, GFP (illustrated Fig 1 and Fig 5f), extends the BRAC family of offline RL algorithms and is closely related to FQL (Park et al., 2025), sharing the same goal of avoiding backpropagation through time. However, the distinctive contribution of GFP lies in integrating the weighted BC approach for the regularization component of the BRAC actor. Rather than using weighted BC to train the final actor directly (as in Fig 5d), GFP employs it to inject value-awareness into the regularizer VaBC. Compared to other methods using a weighted-BC policy, VaBC is not GFP's actor, its weights are computed using the one-step actor trained by DDPG+BC, with the latter using VaBC as a regularizer. This bidirectional guidance between the two policies fundamentally distinguishes GFP from other methods that use weighted BC as their main actor, computing the weights based on the weighted BC policy itself, see Fig 5d.

Removing the value-aware guidance from the GFP flow policy yields FQL (Fig 5e), and removing the one-step actor implies using the flow policy to train the critic, yielding FAWAC (Fig 5d). Alternatively, one could use rejection sampling instead of weighted-BC to regularize the actor with the best actions found by a BC policy, yet at the cost of sampling multiple actions from the flow policy for each policy update step.

6 Discussion and Conclusion

In this work, we revisited behavior regularization for offline RL. BRAC approaches have been shown to be highly effective for policy extraction (Tarasov et al., 2023; Park et al., 2024b). However, conventional BRAC approaches constrain the learned policy to remain close to the raw dataset distribution. While this reduces instability, the regularization term itself typically does not distinguish between the high and low-value actions. This limitation is problematic in suboptimal datasets, where regularizing with all transitions indiscriminately can hinder performance.

To address this, we introduced Guided Flow Policy (GFP). GFP couples a multi-step flow-matching policy trained with value-aware behavior cloning and a distilled one-step actor through a bidirectional guidance mechanism. GFP leverages the expressiveness of flow policies while adding value awareness directly in the flow part, without the drawbacks of backpropagation through time.

Our analysis provides several insights. First, although standard behavior-regularized actor–critic methods, such as ReBRAC, are competitive with good hyperparameter tuning, their dependence on value-agnostic behavior regularization limits performance on suboptimal datasets. In particular, in prior BRAC approaches, adding low-quality transitions in the dataset will degrade performance, while the value-awareness of VaBC filters them out. Second, while generative models such as flow or diffusion policies can represent dataset distributions more flexibly, they inherit these same limitations if trained to match the raw dataset without explicit guidance. GFP leverages the benefits of two distinct policy extraction methods: the effectiveness and scalability of behavior-regularized policy gradient, and the stability of weighted BC. This combination exploits well the expressivity of flow models, without backpropagating through time. This synergy enables GFP to consistently achieve state-of-the-art results, as demonstrated by our extensive and rigorous study on 144 offline RL tasks.

Nonetheless, GFP depends on the availability of a sufficiently accurate critic. In datasets lacking high-value actions or when the critic cannot reliably evaluate them, improvements are limited. Future research directions could explore ways to reduce reliance on the critic or extend GFP to settings with weaker or sparse reward signals.

ACKNOWLEDGMENTS

This work has received support from the French government, managed by the National Research Agency, under the France 2030 program with the references Organic Robotics Program (PEPR O2R) and "PR[AI]RIE-PSAI" (ANR-23-IACL-0008). This research was funded, in part, by l'Agence Nationale de la Recherche (ANR), projects NIMBLE project (ANR-22-CE33-0008), RODEO (ANR-24-CE23-5886), PEPR O2R - AS2 (ANR-22-EXOD-0006), and PEPR O2R – PI3 ASSISTMOV (ANR-22-EXOD-0004). The European Union also supported this work through the ARTIFACT project (GA no. 101165695) and the AGIMUS project (GA no. 101070165). The Paris Île-de-France Région also supported this work in the frame of the DIM AI4IDF. The authors gratefully acknowledge the support and resources provided by the CLEPS infrastructure at Inria Paris. This work was performed using HPC resources from the GENCI-IDRIS Jean-Zay cluster (Grant 2024-AD010616763). Views and opinions expressed are those of the author(s) only and do not necessarily reflect those of the funding agencies.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in Neural Information Processing Systems*, 2021.

Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.

Marvin Alles, Nutan Chen, Patrick van der Smagt, and Botond Cseke. Flowq: Energy-guided flow policies for offline reinforcement learning. *arXiv preprint arXiv:2505.14139*, 2025.

Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos J Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in atari. *Advances in Neural Information Processing Systems*, 37:58757–58791, 2024.

Chang Chen, Fei Deng, Kenji Kawaguchi, Caglar Gulcehre, and Sungjin Ahn. Simple hierarchical planning with diffusion. *arXiv preprint arXiv:2401.02644*, 2024a.

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

Tianyu Chen, Zhendong Wang, and Mingyuan Zhou. Diffusion policies creating a trust region for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 37:50098–50125, 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*, pp. 02783649241273668, 2023.

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

Zihan Ding and Chi Jin. Consistency models as a rich and efficient policy class for reinforcement learning. *arXiv preprint arXiv:2309.16984*, 2023.

Zihan Ding, Amy Zhang, Yuandong Tian, and Qinqing Zheng. Diffusion world model: Future modeling beyond step-by-step rollout for offline reinforcement learning. *arXiv preprint arXiv:2402.03570*, 2024b.

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6, 2005.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

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

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

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

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

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.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

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

Peter Holderrieth and Ezra Erives. An introduction to flow matching and diffusion models. *arXiv preprint arXiv:2506.02070*, 2025.

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

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

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

Vijay Konda and John Tsitsiklis. Actor-critic algorithms. *Advances in neural information processing systems*, 12, 1999.

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.

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

Romain Laroche, Paul Trichelair, and Remi Tachet Des Combes. Safe policy improvement with baseline bootstrapping. In *International conference on machine learning*, pp. 3652–3661. PMLR, 2019.

Théotime Le Hellard, Franki Nguimatsia Tiofack, Quentin Le Lidec, and Justin Carpentier. Sobolev diffusion policy. 2025.

Jaewoo Lee, Sujin Yun, Taeyoung Yun, and Jinkyoo Park. Gta: Generative trajectory augmentation with guidance for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 37:56766–56801, 2024.

Kyowoon Lee, Seongun Kim, and Jaesik Choi. Refining diffusion planner for reliable behavior synthesis by automatic detection of infeasible plans. *Advances in Neural Information Processing Systems*, 36:24223–24246, 2023.

Steven Li, Rickmer Krohn, Tao Chen, Anurag Ajay, Pulkit Agrawal, and Georgia Chalvatzaki. Learning multimodal behaviors from scratch with diffusion policy gradient. *Advances in Neural Information Processing Systems*, 37:38456–38479, 2024.

Wenhao Li, Xiangfeng Wang, Bo Jin, and Hongyuan Zha. Hierarchical diffusion for offline decision making. In *International Conference on Machine Learning*, pp. 20035–20064. PMLR, 2023.

Zhixuan Liang, Yao Mu, Mingyu Ding, Fei Ni, Masayoshi Tomizuka, and Ping Luo. Adaptdiffuser: Diffusion models as adaptive self-evolving planners. *arXiv preprint arXiv:2302.01877*, 2023.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

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

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

Xiao Ma, Sumit Patidar, Iain Haughton, and Stephen James. Hierarchical diffusion policy for kinematics-aware multi-task robotic manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18081–18090, 2024.

Liyuan Mao, Haoran Xu, Xianyuan Zhan, Weinan Zhang, and Amy Zhang. Diffusion-dice: In-sample diffusion guidance for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 37:98806–98834, 2024.

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

Chaoyi Pan, Zeji Yi, Guanya Shi, and Guannan Qu. Model-based diffusion for trajectory optimization. *Advances in Neural Information Processing Systems*, 37:57914–57943, 2024.

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

Seohong Park, Kevin Frans, Sergey Levine, and Aviral Kumar. Is value learning really the main bottleneck in offline rl? *Advances in Neural Information Processing Systems*, 37:79029–79056, 2024b.

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

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

M. Riedmiller S. Lange, T. Gabel. Batch reinforcement learning. in: M. wiering, m. van otterlo (eds) reinforcement learning. *Adaptation, Learning, and Optimization*, 12(3):729, 2012.

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.

HJ Terry Suh, Glen Chou, Hongkai Dai, Lujie Yang, Abhishek Gupta, and Russ Tedrake. Fighting uncertainty with gradients: Offline reinforcement learning via diffusion score matching. In *Conference on Robot Learning*, pp. 2878–2904. PMLR, 2023.

Richard S Sutton, Andrew G Barto, et al. *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.

Siddarth Venkatraman, Shivesh Khaitan, Ravi Tej Akella, John Dolan, Jeff Schneider, and Glen Berseth. Reasoning with latent diffusion in offline reinforcement learning. *arXiv preprint arXiv:2309.06599*, 2023.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.

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

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.

Omar G. Younis, Rodrigo Perez-Vicente, John U. Balis, Will Dudley, Alex Davey, and Jordan K Terry. Minari, September 2024. URL https://doi.org/10.5281/zenodo.13767625.

Ruoqi Zhang, Ziwei Luo, Jens Sjölund, Thomas 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.

Shiyuan Zhang, Weitong Zhang, and Quanquan Gu. Energy-weighted flow matching for offline reinforcement learning. *arXiv preprint arXiv:2503.04975*, 2025.

Qinqing Zheng, Matt Le, Neta Shaul, Yaron Lipman, Aditya Grover, and Ricky TQ Chen. Guided flows for generative modeling and decision making. *arXiv preprint arXiv:2311.13443*, 2023.

APPENDIX CONTENTS

A Implementation details **16**

B Additional experiments **17**

B.1 Modified Bellman target 17

B.2 Advantage weight similar to AWR 18

B.3 Modified guiding function to integrate a^{π_ω} 18

B.4 Temperature analysis 19

B.5 Sensitivity analysis to η and α 19

C Complete results over 144 tasks **19**

Note on LLM usage: In this work, we only used LLMs for grammar and spelling corrections.

A IMPLEMENTATION DETAILS

Network architectures. All critic, actor, and flow neural networks use $[512, 512, 512, 512]$ multi-layer perceptrons with GeLU activations. Layer normalization is applied to the critic network.

Flow matching. The number of Euler steps, M, used in Algo. 1, Line 1, is fixed to 10 for both GFP and FQL, except on the humanoidmaze-large-navigate environment, where we set M to 30. For GFP, we employ a sinusoidal position embedding of the flow step t, with an embedding size of 64.

Doubled Q-learning. Following standard practice, two separate critic networks are trained and then aggregated to compute action values, either by taking the mean or minimum (Fujimoto et al., 2018). As detailed in Sec. 4.3, we find that the aggregation function has a significant impact on performance for specific tasks. Specifically, by reevaluating ReBRAC on OGBench using the same aggregation function as GFP and FQL, we achieved substantial performance improvements.

Minibatch size. We use a minibatch size of $B = 256$ across most experiments, except on the most challenging tasks, where we evaluate each method with both $B = 256$ and $B = 1024$. The humanoidmaze-large-navigate environment is the only task where methods benefit from different batch sizes: GFP performs best with $B = 1024$, while other methods work better with $B = 256$. Note that on this task, using $\gamma = 0.999$ substantially improves the performance of ReBRAC and FQL compared to previously reported results. For Minari Gym-Mujoco, we use $B = 1024$ following the recommendation in Tarasov et al. (2023) for D4RL Gym-Mujoco.

Training and evaluation. To ensure a fair comparison with FQL, we use identical training durations: 1 M gradient steps on OGBench state-based environments, and 500 K steps on D4RL and OGBench pixel-based tasks. For Minari, we adopt 1 M steps following standard practices. Evaluation differs according to the benchmark: D4RL and Minari scores are computed at the end of training, while OGBench scores are averaged over the final three checkpoints (800K, 900K, and 1M steps for state-based, 300K, 400K and 500K steps for pixel-based) following their official evaluation protocol. All results are reported across 8 random seeds that were not used during the hyperparameter tuning process, and each evaluation is done over 100 episodes. Except for OGBench pixel-based tasks, where scores are computed using 50 episodes and 4 seeds, due to high computational cost, following Park et al. (2025).

Minari MuJoCo reference scores. Since reference scores for the MuJoCo locomotion tasks are not yet available in Minari (Younis et al., 2024), we use the reference scores reported in its predecessor D4RL (Fu et al., 2020).

Parameter search methodology. Tab. 4 summarizes the set of hyperparameter values shared across environments and methods. For task and method specific hyperparameters, our search follows a systematic approach. First, we conduct a logarithmic sweep over the BC coefficient α, which is the main parameter for all methods. For ReBRAC, we sweep over the actor coefficient α_1 while keeping

the critic coefficient α_2 fixed at 0.01. Then, we sweep over α_2 after selecting the optimal α_1. For GFP, we fix $\eta = 10^{-3}$ first and then sweep over η once α is chosen. Sec. B.5 studies how sensitive GFP is to η and α, and justifies first searching α before adjusting η. Hyperparameter search uses four seeds, separate from the eight seeds used for final evaluation. On OGBench, hyperparameters are shared across the five tasks within each environment.

Table 4: **Summary of shared hyperparameters used across all methods and benchmark evaluations.** Environment-specific variations are indicated where applicable.

Hyperparameter	**Value**
Learning rate	0.0003
Gradient steps	1,000,000 (OGBench, Minari), 500,000 (D4RL)
Minibatch size	256 (default), 1024 (Minari Gym-Mujoco), OGBench Tab. 5
Discount factor	0.99 (D4RL, Minari), OGBench Tab. 5
Euler integration steps	10 (default), 30 (humanoidmaze-large)
Critic aggregation function	mean (default), min (D4RL-antmaze, OGBench-antmaze)
Critic target network smoothing coefficient	0.005
Bellman target	$y = r + \gamma Q(s', a')$ (default), y^{VaBC} (cube, humanoidmaze-medium)

Table 5: **Discount factor and minibatch size for OGBench environments.** The asterisk * in some discount factors indicates cases where we modified the discount factor used in prior work, which led to significant performance improvements for the corresponding methods.

Task Category	**Discount factor γ**	**Minibatch size**
antmaze-large-navigate-singletask (5 tasks)	0.995*	256
antmaze-large-stitch-singletask (5 tasks)	0.995 (except for FQL, 0.99)	256
antmaze-large-explore-singletask (5 tasks)	0.995	1024
antmaze-giant-navigate-singletask (5 tasks)	0.995	1024
humanoidmaze-medium-navigate-singletask (5 tasks)	0.995	256
humanoidmaze-medium-stitch-singletask (5 tasks)	0.999	256
humanoidmaze-large-navigate-singletask (5 tasks)	0.999*	256 (except for GFP, 1024)
antsoccer-arena-navigate-singletask (5 tasks)	0.99	256
antsoccer-arena-stitch-singletask (5 tasks)	0.99	256
cube-single-play-singletask (5 tasks)	0.99	256
cube-single-noisy-singletask (5 tasks)	0.99	256
cube-double-play-singletask (5 tasks)	0.99	256
cube-double-noisy-singletask (5 tasks)	0.99	256
cube-triple-play-singletask (5 tasks)	0.99	1024
cube-triple-noisy-singletask (5 tasks)	0.99	1024
puzzle-3×3-play-singletask (5 tasks)	0.99	256
puzzle-4×4-play-singletask (5 tasks)	0.99	256
puzzle-4×4-noisy-singletask (5 tasks)	0.99	256
scene-play-singletask	0.99	256
scene-noisy-singletask	0.99	256
visual tasks	0.99	256

B Additional Experiments

B.1 Modified Bellman target

As described in Sec. 3, we propose a variant of the Bellman target, y^{VaBC} (Eq. 7), that leverages the VaBC policy. Tab. 6 presents experimental results showing average scores over 8 seeds for the selected hyperparameters, with "$\sim$" indicating configurations that were tested but not ultimately chosen. These experiments show that this modified target provides improvements in the cube and humanoidmaze-medium environments, and constant or slightly degraded results on other environments, hence y^{VaBC} is used only for the former ones, as reported in Tab. 4.

Implementation Note on Eq. 7: After finishing the experiments, we realized that in our implementation of the modified y^{VaBC} Bellman target, we were computing $Q_{\bar{\phi}}(s', \mu_\omega(s, z))$ instead of $Q_{\bar{\phi}}(s', \mu_\omega(s', z))$. In our public release of the code, we kept this behavior by default to match the code used in our experiments, but we added an option to use $Q_{\bar{\phi}}(s', \mu_\omega(s', z))$, for future experiments.

Table 6: **Comparison of the modified Bellman target for GFP.**

Task Category	**Standard target y**	**Modified y^{VaBC}** Eq. 7
cube-double-play (5 tasks)	~ 28	47.2 ± 1.6
cube-double-noisy (5 tasks)	~ 46	63.1 ± 3.3
humanoidmaze-medium-navigate (5 tasks)	~ 64	72.0 ± 2.8
humanoidmaze-medium-stitch (5 tasks)	~ 59	66.2 ± 5.7
puzzle-4x4-play (5 tasks)	26.1 ± 2.1	~ 22
scene-play (5 tasks)	53.5 ± 2.9	~ 50

B.2 Advantage weight similar to AWR

As explained at the end of Sec. 3, our g_η is a soft-max weight comparing a to a^{π_θ}, an action sampled from the actor, but one could alternatively use an advantage weighted term similar to AWR (Peng et al., 2019), but using the actor to compute a baseline:

$$g_\eta^{\text{AWR}}(s,a) := \exp\left(\frac{\lambda}{\eta}\big(Q_\phi(s,a) - Q_\phi(s,\mu_\theta(s,z))\big)\right) \tag{12}$$

Tab. 7 reports results when using g_η^{AWR} instead of g_η, over 65 tasks, with α and η tuned accordingly, Tab. 13. The modified Bellman target y^{VaBC} has also been tested for this variant, as in Sec. B.1, and was found to be beneficial for the cube, humanoidmaze, and puzzle environments.

g_η^{AWR} achieves slightly better performance on standard tasks, but our main g_η scales better on more challenging ones, e.g., humanoid environments and cube-triple ones. Important note: GFP-AWR only changes the weighting function, keeping the general structure presented in Fig 1 and Fig 5f. In particular, it does not mean first using AWR to learn a weighted-BC policy (like FAWAC, see Fig 5d) and then using it to regularize the actor. GFP-AWR preserves the bidirectional guidance, a core contribution of GFP (see the arrows in Fig 5f). As an alternative, one could also learn a value-function network $V(s) = \mathbb{E}_{s;a\sim\pi_\theta(\cdot|s)}(Q(s,a))$ and use it instead of the stochastic baseline $Q(s,\mu_\theta(s,z))$ in the weighting functions.

Table 7: **Comparison with an AWR guiding function**

Task Category	**GFP**-g_η (default)	**GFP**-AWR
antmaze-large-stitch (5 tasks)	$\mathbf{68.9} \pm 0.8$	$\mathbf{69.8} \pm 1.9$
antmaze-giant-navigate (5 tasks)	$\mathbf{27.9} \pm 8.5$	$\mathbf{29.1} \pm 8.4$
humanoidmaze-medium-navigate (5 tasks)	$\mathbf{72.0} \pm 2.8$	60.9 ± 3.4
humanoidmaze-medium-stitch (5 tasks)	$\mathbf{66.2} \pm 5.7$	$\mathbf{64.3} \pm 8.5$
humanoidmaze-large-navigate (5 tasks)	$\mathbf{17.0} \pm 9.6$	14.3 ± 3.5
antsoccer-arena-stitch (5 tasks)	$\mathbf{30.5} \pm 2.2$	$\mathbf{29.5} \pm 2.6$
cube-double-play (5 tasks)	$\mathbf{47.2} \pm 1.6$	$\mathbf{50.4} \pm 3.5$
cube-double-noisy (5 tasks)	$\mathbf{63.1} \pm 3.3$	55.7 ± 3.4
cube-triple-play(5 tasks)	$\mathbf{15.9} \pm 2.0$	9.9 ± 2.9
cube-triple-noisy (5 tasks)	$\mathbf{24.5} \pm 2.8$	11.2 ± 1.4
puzzle-3x3-play (5 tasks)	23.1 ± 2.2	$\mathbf{28.3} \pm 1.0$
puzzle-4x4-play (5 tasks)	$\mathbf{18.8} \pm 1.7$	16.5 ± 1.7
puzzle-4x4-noisy (5 tasks)	$\mathbf{26.1} \pm 2.1$	$\mathbf{27.1} \pm 1.7$
Average (65 tasks)	**38.6**	35.9

B.3 Modified guiding function to integrate a^{π_ω}

Our guiding function g_η, defined in Eq. 10, compares a dataset action a to a^{π_θ}, an action sampled from the actor. Since we also sample actions using the VaBC flow policy (Algo. 1, Line 10), we can alternatively use a^{π_ω} for a more conservative guiding function, leading to a modified guided function:

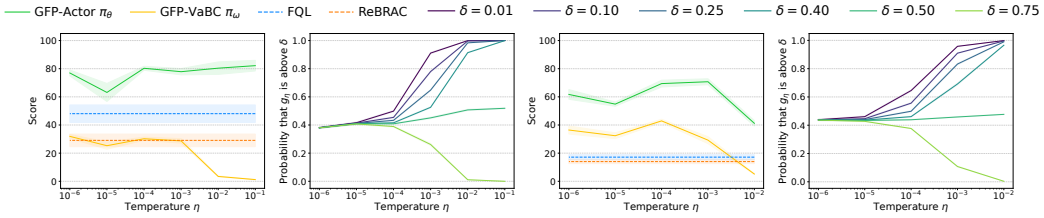
$$g_\eta^{min}(s,a) = \frac{\exp\left(\frac{\lambda}{\eta}Q_\phi(s,a)\right)}{\exp\left(\frac{\lambda}{\eta}Q_\phi(s,a)\right) + \exp\left(\frac{\lambda}{\eta}\ \min\left(Q_\phi(s,a^{\pi_\theta}), Q_\phi(s,a^{\pi_\omega})\right)\right)} \tag{13}$$

This modified guiding function is more conservative because it only filters out action a when both policies produce more valuable alternatives. Initially, we employed $g_\eta^{min}(s,a)$ for evaluation on

OGBench antmaze, humanoidmaze, and antsoccer environments. However, we found no significant performance difference, so all remaining tasks use the standard g_η defined in Eq. 10.

B.4 Temperature analysis

To complete the analysis presented in Sec. 4.2, we conducted experiments on two additional tasks beyond those shown in Fig. 4. We tested humanoidmaze-medium-stitch as a locomotion task and cube-triple-play as a non-noisy manipulation task. The results of this extended temperature analysis are reported in Fig. 6, illustrating how temperature controls the sharpness of value-guided filtering.



(a) Humanoidmaze score (b) Humanoidmaze $\mathbb{P}(g_\eta > \delta)$ (c) Cube-triple score (d) Cube-triple $\mathbb{P}(g_\eta > \delta)$

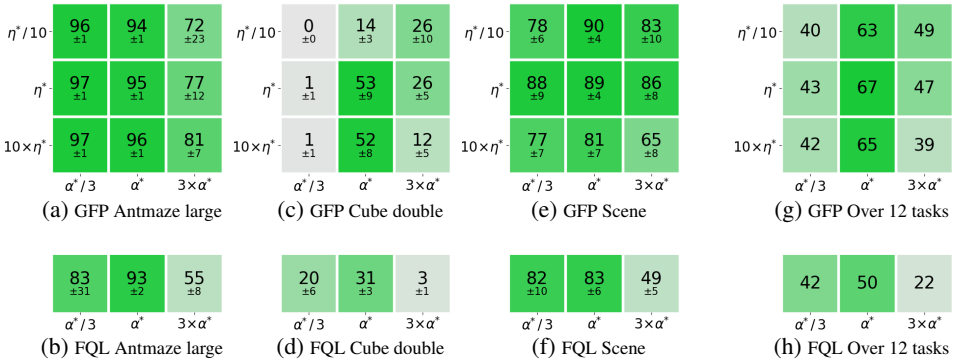
Figure 6: **Temperature analysis on two additional very challenging tasks.** Plots (a) and (c), performance scores across temperature values η for our GFP method (Actor π_θ and VaBC π_ω) compared to baselines (FQL, ReBRAC) on humanoidmaze-medium-stitch-task1 and cube-triple-play-task1. Plots (b) and (d), probability that the guidance term g_η is above different thresholds δ as a function of temperature, illustrating how temperature controls the sharpness of value-guided filtering.

B.5 Sensitivity analysis to η and α

To study how sensitive GFP is to the temperature η and the BC coefficient α, we evaluated 8 variations around task-specific hyperparameters (α^*, η^*) on 12 tasks, with comparison to FQL sensitivity to α. These experiments reveal that α is the most important hyperparameter of GFP, as it is for most offline RL methods (Tarasov et al., 2023; Park et al., 2024b), and that GFP is less sensitive to precise η tuning. It justifies our methodology for hyperparameter search, by first fixing $\eta = 10^{-3}$, and then sweeping over η only once α is chosen. From the results averaged on 12 tasks, it can be observed that α and η are partially correlated: when increasing α (i.e. stronger regularization), it is better to decrease η (stronger filtering).

C Complete results over 144 tasks

This section presents the comprehensive experimental results across all 144 tasks from OGBench (Tabs. 8 and 9), D4RL (Tab. 11), and Minari (Tab. 12) benchmarks. Tab. 10 covers comparison against 10 prior works over 50 tasks. All results are averaged over 8 seeds, except for pixel-based tasks. The evaluation encompasses approximately 15,000 individual training runs, providing detailed performance comparisons across diverse offline RL scenarios, including navigation, manipulation, and locomotion tasks. The hyperparameters used are stated in Tabs. 14 and 13.



(a) GFP Antmaze large (c) GFP Cube double (e) GFP Scene (g) GFP Over 12 tasks

(b) FQL Antmaze large (d) FQL Cube double (f) FQL Scene (h) FQL Over 12 tasks

Figure 7: **Sensitivity analysis to the BC coefficient α, and to the temperature η, in log-scale.** Evaluates the sensitivity by testing variations around task-specific hyperparameters (α^*, η^*) reported Tab. 13, using 8 seeds. Plots (a), (c) and (e), report GFP's sensitivity to α^* and η^*, on antmaze-large-navigate-task1, cube-double-play-task2, and scene-play-task2, respectively. These three tasks were chosen following FQL (Park et al., 2025) ablation study on α (FQL page 15). Plots (b), (d) and (f), report the sensitivity of FQL with respect to α (reevaluated as FQL's authors did not share exact numbers). Plots (g) and (h) aggregate the analysis over 12 tasks (cube single and double, play and noisy; scene play and noisy; antmaze navigate large and giant; humanoidmaze medium navigate and stitch, where for each environment we use the default task). These experiments reveal that α is the most important hyperparameter of GFP, as it is for most offline RL methods (Tarasov et al., 2023; Park et al., 2024b), and that GFP is less sensitive to precise η tuning.

Table 8: **Offline RL full results on OGBench, part 1/2.** Models were trained with 8 random seeds and evaluated over 100 episodes, following the setup of prior work (Park et al., 2024a; 2025). Scores are averaged across seeds; values within 95% of the best performance are shown in bold, while *italics* indicate scores reported from prior work (Park et al., 2025). GFP actor π_θ is our primary policy, while GFP VaBC π_ω is reported as a byproduct of training. The $\pm$ symbol denotes the standard deviation over seeds. See Tab 9 for the other tasks.

Task	Offline RL algorithms				
	IQL	ReBRAC	FQL	GFP actor π_θ	GFP VaBC π_ω
antmaze-large-navigate-singletask-task1-v0	*48* ± 9	**97.7** ± 0.5	92.6 ± 1.8	**95.4** ± 0.8	92.0 ± 2.3
antmaze-large-navigate-singletask-task2-v0	*42* ± 6	**92.2** ± 1.6	80.0 ± 10.4	**92.2** ± 3.0	**88.5** ± 1.6
antmaze-large-navigate-singletask-task3-v0	*72* ± 7	**98.5** ± 1.8	93.5 ± 1.6	**95.6** ± 2.7	**94.4** ± 3.0
antmaze-large-navigate-singletask-task4-v0	*51* ± 9	**94.4** ± 0.9	82.3 ± 15.5	**90.6** ± 2.6	85.6 ± 3.7
antmaze-large-navigate-singletask-task5-v0	*54* ± 22	**96.5** ± 0.9	**92.3** ± 1.5	**95.0** ± 1.3	89.8 ± 1.7
antmaze-large-stitch-singletask-task1-v0	28.2 ± 7.8	**88.4** ± 16.2	71.8 ± 28.4	**90.3** ± 2.1	85.0 ± 9.7
antmaze-large-stitch-singletask-task2-v0	5.5 ± 4.1	**85.2** ± 5.9	25.4 ± 25.3	**82.9** ± 2.3	69.5 ± 7.2
antmaze-large-stitch-singletask-task3-v0	83.4 ± 2.5	**98.0** ± 0.8	88.4 ± 3.5	**93.2** ± 3.4	90.2 ± 1.5
antmaze-large-stitch-singletask-task4-v0	8.8 ± 2.9	**79.4** ± 29.9	23.7 ± 24.6	0.2 ± 0.6	1.2 ± 3.1
antmaze-large-stitch-singletask-task5-v0	26.2 ± 14.9	**95.1** ± 1.7	81.5 ± 7.2	77.9 ± 7.1	42.2 ± 8.7
antmaze-large-explore-singletask-task1-v0	0.4 ± 1.0	**91.8** ± 6.1	**91.0** ± 10.9	**92.8** ± 4.1	**92.3** ± 1.0
antmaze-large-explore-singletask-task2-v0	0.0 ± 0.0	86.4 ± 5.2	**90.8** ± 3.4	**88.5** ± 3.6	84.1 ± 3.1
antmaze-large-explore-singletask-task3-v0	54.8 ± 7.7	**99.1** ± 0.6	**97.5** ± 0.9	**98.0** ± 0.5	92.8 ± 1.1
antmaze-large-explore-singletask-task4-v0	9.2 ± 4.1	54.4 ± 30.5	**89.0** ± 2.9	**87.1** ± 2.3	87.0 ± 2.8
antmaze-large-explore-singletask-task5-v0	0.0 ± 0.1	81.8 ± 23.7	71.1 ± 32.0	**93.1** ± 2.1	**90.2** ± 9.7
antmaze-giant-navigate-singletask-task1-v0	*0* ± 0	**17.5** ± 3.8	0.8 ± 2.1	12.6 ± 15.1	0.1 ± 0.1
antmaze-giant-navigate-singletask-task2-v0	*1* ± 1	44.9 ± 6.7	23.2 ± 15.3	**52.2** ± 26.5	1.2 ± 0.7
antmaze-giant-navigate-singletask-task3-v0	*0* ± 0	2.5 ± 1.3	0.9 ± 1.1	**13.7** ± 10.2	0.2 ± 0.2
antmaze-giant-navigate-singletask-task4-v0	*0* ± 0	**20.0** ± 20.0	9.9 ± 10.8	17.8 ± 19.9	0.4 ± 0.3
antmaze-giant-navigate-singletask-task5-v0	19 ± 7	**81.4** ± 6.1	46.9 ± 25.5	43.2 ± 35.7	2.0 ± 0.7
humanoidmaze-medium-navigate-singletask-task1-v0	*32* ± 7	34.1 ± 16.4	*19* ± 12	**83.5** ± 3.7	28.8 ± 6.5
humanoidmaze-medium-navigate-singletask-task2-v0	*41* ± 9	75.8 ± 29.	***94*** ± 3	**91.2** ± 6.3	60.2 ± 9.4
humanoidmaze-medium-navigate-singletask-task3-v0	*25* ± 5	69.5 ± 25.7	*74* ± 18	**86.3** ± 10.7	28.9 ± 4.1
humanoidmaze-medium-navigate-singletask-task4-v0	*0* ± 1	**19.5** ± 17.5	*3* ± 4	3.0 ± 6.6	1.3 ± 2.0
humanoidmaze-medium-navigate-singletask-task5-v0	*66* ± 4	**97.0** ± 0.9	***97*** ± 2	**95.8** ± 1.3	60.1 ± 6.1
humanoidmaze-medium-stitch-singletask-task1-v0	26.4 ± 3.0	29.1 ± 18.3	48.0 ± 25.4	**77.9** ± 9.2	28.7 ± 6.5
humanoidmaze-medium-stitch-singletask-task2-v0	27.9 ± 9.9	**94.4** ± 1.9	87.5 ± 3.7	**95.2** ± 1.8	49.2 ± 6.3
humanoidmaze-medium-stitch-singletask-task3-v0	30.0 ± 4.4	56.6 ± 24.5	**85.4** ± 17.8	55.2 ± 33.9	44.3 ± 3.2
humanoidmaze-medium-stitch-singletask-task4-v0	3.7 ± 1.6	**33.1** ± 28.4	0.8 ± 0.8	3.6 ± 9.5	14.9 ± 9.0
humanoidmaze-medium-stitch-singletask-task5-v0	48.5 ± 4.6	92.5 ± 3.1	**94.1** ± 2.5	**98.9** ± 0.5	60.2 ± 7.4
humanoidmaze-large-navigate-singletask-task1-v0	*3* ± 1	27.8 ± 13.4	19.8 ± 13.1	**57.2** ± 23.7	4.5 ± 2.9
humanoidmaze-large-navigate-singletask-task2-v0	*0* ± 0	**0.5** ± 0.7	0.0 ± 0.1	0.1 ± 0.2	0.0 ± 0.0
humanoidmaze-large-navigate-singletask-task3-v0	7 ± 3	**25.9** ± 8.5	8.8 ± 4.0	14.6 ± 16.2	5.5 ± 2.8
humanoidmaze-large-navigate-singletask-task4-v0	*1* ± 0	**8.3** ± 14.4	1.6 ± 1.8	3.7 ± 4.1	0.7 ± 0.5
humanoidmaze-large-navigate-singletask-task5-v0	*1* ± 1	1.9 ± 1.5	2.2 ± 3.9	**13.1** ± 15.5	1.5 ± 0.9
antsoccer-arena-navigate-singletask-task1-v0	*14* ± 5	62.1 ± 3.6	***77*** ± 4	**77.0** ± 1.7	17.0 ± 2.8
antsoccer-arena-navigate-singletask-task2-v0	*17* ± 7	78.5 ± 2.8	***88*** ± 3	**91.2** ± 2.2	16.8 ± 2.6
antsoccer-arena-navigate-singletask-task3-v0	*6* ± 4	55.5 ± 1.7	***61*** ± 6	51.9 ± 4.9	7.8 ± 2.9
antsoccer-arena-navigate-singletask-task4-v0	*3* ± 2	34.8 ± 5.0	***39*** ± 6	**40.2** ± 4.2	5.2 ± 2.1
antsoccer-arena-navigate-singletask-task5-v0	*2* ± 2	**48.5** ± 6.1	*36* ± 9	29.1 ± 8.9	4.6 ± 2.1
antsoccer-arena-stitch-singletask-task1-v0	5.3 ± 3.3	44.6 ± 5.0	**53.4** ± 3.5	**51.8** ± 3.5	3.0 ± 1.2
antsoccer-arena-stitch-singletask-task2-v0	5.6 ± 1.9	30.0 ± 6.0	49.1 ± 8.1	**53.0** ± 7.6	3.0 ± 1.4
antsoccer-arena-stitch-singletask-task3-v0	1.3 ± 1.7	15.9 ± 2.4	**19.3** ± 2.7	**18.7** ± 1.4	0.4 ± 0.3
antsoccer-arena-stitch-singletask-task4-v0	0.4 ± 0.5	14.8 ± 4.2	20.0 ± 6.4	**26.1** ± 4.7	0.1 ± 0.2
antsoccer-arena-stitch-singletask-task5-v0	1.3 ± 1.8	**4.8** ± 1.6	1.2 ± 0.4	2.9 ± 2.9	0.6 ± 0.4
cube-single-play-singletask-task1-v0	*88* ± 3	*89* ± 5	***97*** ± 2	**99.1** ± 0.4	42.1 ± 5.9
cube-single-play-singletask-task2-v0	*85* ± 8	*92* ± 4	***97*** ± 2	**99.4** ± 0.7	38.8 ± 6.4
cube-single-play-singletask-task3-v0	*91* ± 5	*93* ± 3	***98*** ± 2	**99.4** ± 0.5	48.5 ± 9.1
cube-single-play-singletask-task4-v0	*73* ± 6	*92* ± 3	*94* ± 3	**99.1** ± 0.7	32.8 ± 9.3
cube-single-play-singletask-task5-v0	*78* ± 9	*87* ± 8	***93*** ± 3	**97.0** ± 1.6	36.3 ± 5.5
cube-single-noisy-singletask-task1-v0	52.3 ± 7.2	**99.2** ± 1.1	**100.0** ± 0.0	**100.0** ± 0.0	**99.9** ± 0.2
cube-single-noisy-singletask-task2-v0	55.3 ± 8.0	**96.0** ± 3.5	**100.0** ± 0.1	**100.0** ± 0.1	**99.9** ± 0.2
cube-single-noisy-singletask-task3-v0	34.3 ± 8.1	**97.4** ± 1.6	**100.0** ± 0.0	**100.0** ± 0.0	**100.0** ± 0.0
cube-single-noisy-singletask-task4-v0	63.2 ± 7.5	**99.7** ± 0.5	**100.0** ± 0.1	**100.0** ± 0.0	**99.9** ± 0.1
cube-single-noisy-singletask-task5-v0	60.9 ± 11.7	**99.8** ± 0.2	**100.0** ± 0.1	**99.9** ± 0.2	**99.8** ± 0.3

Table 9: **Offline RL full results on OGBench, part 2/2.** Models were trained with 8 random seeds and evaluated over 100 episodes, following the setup of prior work (Park et al., 2024a; 2025). Scores are averaged across seeds; values within 95% of the best performance are shown in bold, while *italics* indicate scores reported from prior work (Park et al., 2025). GFP actor π_θ is our primary policy, while GFP VaBC π_ω is reported as a byproduct of training. The $\pm$ symbol denotes the standard deviation over seeds. See Tab 8 for the other tasks.

Task	Offline RL algorithms				
	IQL	ReBRAC	FQL	GFP actor π_ω	GFP VaBC π_θ
cube-double-play-singletask-task1-v0	*27* ± 5	43.0 ± 8.9	*61* ± 9	**76.1** ± 4.6	28.5 ± 5.0
cube-double-play-singletask-task2-v0	*1* ± 1	16.2 ± 5.0	*36* ± 6	**53.3** ± 8.9	1.8 ± 1.0
cube-double-play-singletask-task3-v0	*0* ± 0	1.3 ± 0.4	*22* ± 5	**43.3** ± 8.9	0.4 ± 0.4
cube-double-play-singletask-task4-v0	*0* ± 0	0.4 ± 0.3	*5* ± 2	**7.1** ± 3.1	0.8 ± 0.6
cube-double-play-singletask-task5-v0	*4* ± 3	2.0 ± 1.0	*19* ± 10	**56.3** ± 11.3	0.7 ± 0.6
cube-double-noisy-singletask-task1-v0	20.8 ± 3.4	51.3 ± 9.5	77.1 ± 8.0	**89.5** ± 4.5	32.2 ± 3.9
cube-double-noisy-singletask-task2-v0	0.0 ± 0.1	21.1 ± 4.3	43.1 ± 10.5	75.7 ± 7.6	5.8 ± 1.3
cube-double-noisy-singletask-task3-v0	0.8 ± 1.0	8.0 ± 3.4	26.3 ± 5.8	**75.0** ± 4.4	3.2 ± 1.2
cube-double-noisy-singletask-task4-v0	0.2 ± 0.2	6.5 ± 1.8	15.5 ± 3.9	**41.8** ± 4.6	1.6 ± 0.9
cube-double-noisy-singletask-task5-v0	0.5 ± 0.5	11.2 ± 3.3	29.0 ± 7.9	**33.4** ± 7.6	3.9 ± 1.5
cube-triple-play-singletask-task1-v0	0.4 ± 0.3	14.0 ± 5.8	17.2 ± 7.3	**54.8** ± 6.2	32.4 ± 8.4
cube-triple-play-singletask-task2-v0	0.0 ± 0.0	0.1 ± 0.1	0.8 ± 0.2	**6.6** ± 6.3	0.6 ± 0.7
cube-triple-play-singletask-task3-v0	1.3 ± 0.6	0.3 ± 0.3	1.3 ± 0.6	**14.9** ± 9.9	3.2 ± 2.1
cube-triple-play-ingletask-task4-v0	0.0 ± 0.0	0.0 ± 0.0	0.3 ± 0.4	**2.5** ± 1.7	0.8 ± 0.2
cube-triple-play-singletask-task5-v0	0.1 ± 0.2	0.3 ± 0.5	0.1 ± 0.2	**0.6** ± 0.5	**1.0** ± 0.9
cube-triple-noisy-singletask-task1-v0	24.0 ± 6.0	25.3 ± 13.9	17.5 ± 8.0	**90.7** ± 5.2	41.0 ± 6.6
cube-triple-noisy-singletask-task2-v0	0.0 ± 0.1	0.4 ± 0.4	0.1 ± 0.2	**8.9** ± 3.4	0.3 ± 0.2
cube-triple-noisy-singletask-task3-v0	0.0 ± 0.0	0.1 ± 0.2	0.0 ± 0.0	**11.8** ± 7.5	0.8 ± 0.7
cube-triple-noisy-ingletask-task4-v0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.1	**10.8** ± 6.2	0.8 ± 0.5
cube-triple-noisy-singletask-task5-v0	0.0 ± 0.0	0.0 ± 0.1	0.0 ± 0.0	**0.1** ± 0.2	**0.0** ± 0.0
puzzle-3×3-play-singletask-task1-v0	*33* ± 6	***97*** ± 4	*90* ± 4	**94.8** ± 4.4	54.6 ± 7.4
puzzle-3×3-play-singletask-task2-v0	*4* ± 3	*1* ± 1	***16*** ± 5	0.3 ± 0.3	11.2 ± 2.4
puzzle-3×3-play-singletask-task3-v0	*3* ± 2	*3* ± 1	***10*** ± 3	0.9 ± 0.6	**9.7** ± 1.5
puzzle-3×3-play-singletask-task4-v0	*2* ± 1	*2* ± 1	***16*** ± 5	5.4 ± 2.1	9.4 ± 3.1
puzzle-3×3-play-singletask-task5-v0	*3* ± 2	*5* ± 3	***16*** ± 3	**14.1** ± 8.4	10.9 ± 4.8
puzzle-4×4-play-singletask-task1-v0	*12* ± 2	45.4 ± 3.7	*34* ± 8	**50.0** ± 8.1	16.2 ± 3.3
puzzle-4×4-play-singletask-task2-v0	*7* ± 4	2.7 ± 1.0	***16*** ± 5	9.9 ± 2.5	7.0 ± 1.8
puzzle-4×4-play-singletask-task3-v0	*9* ± 3	27.8 ± 4.0	*18* ± 5	**46.2** ± 3.8	10.2 ± 2.9
puzzle-4×4-play-singletask-task4-v0	*5* ± 2	9.1 ± 2.1	*11* ± 3	**17.2** ± 2.5	7.4 ± 1.9
puzzle-4×4-play-singletask-task5-v0	*4* ± 1	0.8 ± 0.7	***7*** ± 3	**7.3** ± 3.6	**6.6** ± 1.8
puzzle-4×4-noisy-singletask-task1-v0	0.1 ± 0.1	3.9 ± 1.2	**41.0** ± 3.8	38.5 ± 3.6	**39.6** ± 4.8
puzzle-4×4-noisy-singletask-task2-v0	0.0 ± 0.1	0.4 ± 0.4	**5.9** ± 1.7	0.7 ± 0.5	3.5 ± 1.1
puzzle-4×4-noisy-singletask-task3-v0	0.1 ± 0.2	0.9 ± 0.5	20.8 ± 2.7	**51.1** ± 6.5	44.0 ± 4.7
puzzle-4×4-noisy-singletask-task4-v0	0.0 ± 0.0	0.4 ± 0.4	**6.5** ± 1.6	3.0 ± 1.6	**6.3** ± 2.2
puzzle-4×4-noisy-singletask-task5-v0	0.0 ± 0.0	0.0 ± 0.1	**3.7** ± 1.7	0.7 ± 0.7	**3.1** ± 2.0
scene-play-singletask-task1-v0	*94* ± 3	***95*** ± 2	***100*** ± 0	**99.8** ± 0.4	**99.8** ± 0.2
scene-play-singletask-task2-v0	*12* ± 3	*50* ± 13	*76* ± 9	**89.0** ± 4.1	**93.0** ± 5.1
scene-play-singletask-task3-v0	*32* ± 7	*55* ± 16	***98*** ± 1	78.0 ± 13.2	**93.5** ± 5.1
scene-play-singletask-task4-v0	*0* ± 0	*3* ± 3	***5*** ± 1	0.6 ± 0.6	1.8 ± 1.3
scene-play-singletask-task5-v0	*0* ± 0	*0* ± 0	*0* ± 0	0.0 ± 0.0	0.0 ± 0.0
scene-noisy-singletask-task1-v0	74.2 ± 5.4	94.8 ± 3.7	**100.0** ± 0.0	**99.9** ± 0.2	**99.9** ± 0.2
scene-noisy-singletask-task2-v0	0.1 ± 0.2	18.1 ± 5.9	87.4 ± 3.7	**94.2** ± 2.0	**97.4** ± 1.9
scene-noisy-singletask-task3-v0	5.7 ± 1.1	81.1 ± 5.5	**94.4** ± 3.7	**93.3** ± 4.3	**95.2** ± 3.2
scene-noisy-singletask-task4-v0	0.0 ± 0.1	5.6 ± 3.4	**14.8** ± 4.6	0.0 ± 0.0	0.1 ± 0.2
scene-noisy-singletask-task5-v0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
visual-cube-single-play-singletask-task1-v0[1]	*70* ± 12	***83*** ± 6	***81*** ± 12	**82.3** ± 4.2	–[1]
visual-cube-double-play-singletask-task1-v0[1]	***34*** ± 23	*4* ± 4	*21* ± 11	19.7 ± 8.7	–
visual-scene-play-singletask-task1-v0[1]	***97*** ± 2	***98*** ± 4	***98*** ± 3	**99.3** ± 0.5	–
visual-puzzle-3x3-play-singletask-task1-v0[1]	*7* ± 15	*88* ± 4	***94*** ± 1	**92.8** ± 2.0	–
visual-puzzle-4x4-play-singletask-task1-v0[1]	*0* ± 0	*26* ± 6	***33*** ± 6	19.7 ± 1.7	–

[1] Following Park et al. (2025) to reduce the computational cost of pixel-based tasks, for these tasks we use 4 seeds and 50 episodes per seed; moreover, we chose not to evaluate the VaBC policy, as it is only a byproduct.

Table 10: **Offline RL Results on OGBench Tasks (only task evaluated in FQL):** Performance comparison across different offline RL algorithms. All prior work results are taken from Park et al. (2025) to compare a wide range of previous methods against GFP.

Task	BC	IQL	ReBRAC	IDQL	SRPO	CAC	FAWAC	FBRAC	IFQL	FQL	**GFP** actor π_ω
antmaze-large-navigate-singletask											
task1-v0	0 ± 0	48 ± 9	91 ± 10	0 ± 0	0 ± 0	42 ± 7	1 ± 1	70 ± 20	24 ± 17	80 ± 8	**95.4** ± 0.8
task2-v0	6 ± 3	42 ± 6	88 ± 4	14 ± 8	4 ± 4	1 ± 1	0 ± 1	35 ± 12	8 ± 3	57 ± 10	**92.2** ± 3.0
task3-v0	29 ± 5	72 ± 7	51 ± 18	26 ± 8	3 ± 2	49 ± 10	12 ± 4	83 ± 15	52 ± 17	**93** ± 3	**95.6** ± 2.7
task4-v0	8 ± 3	51 ± 9	84 ± 7	62 ± 25	45 ± 19	17 ± 6	10 ± 3	37 ± 18	18 ± 8	80 ± 4	**90.6** ± 2.6
task5-v0	10 ± 3	54 ± 22	90 ± 2	2 ± 2	1 ± 1	55 ± 6	9 ± 5	76 ± 8	38 ± 18	83 ± 4	**95.0** ± 1.3
antmaze-giant-navigate-singletask-task1											
task1-v0	0 ± 0	0 ± 0	**27** ± 22	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 1	0 ± 0	4 ± 5	12.6 ± 15.1
task2-v0	0 ± 0	1 ± 1	16 ± 17	0 ± 0	0 ± 0	0 ± 0	0 ± 0	4 ± 7	0 ± 0	9 ± 7	**52.2** ± 26.5
task3-v0	0 ± 0	0 ± 0	**34** ± 22	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 1	13.7 ± 10.2
task4-v0	0 ± 0	0 ± 0	5 ± 12	0 ± 0	0 ± 0	0 ± 0	0 ± 0	9 ± 4	0 ± 0	14 ± 23	**17.8** ± 19.9
task5-v0	1 ± 1	19 ± 7	**49** ± 22	0 ± 1	0 ± 0	0 ± 0	0 ± 0	6 ± 10	13 ± 9	16 ± 28	43.2 ± 35.7
humanoidmaze-medium-navigate-singletask											
task1-v0	1 ± 0	32 ± 7	16 ± 9	1 ± 1	0 ± 0	38 ± 19	6 ± 2	25 ± 8	69 ± 19	19 ± 12	**83.5** ± 3.7
task2-v0	1 ± 0	41 ± 9	18 ± 16	1 ± 1	1 ± 1	47 ± 35	40 ± 2	76 ± 10	85 ± 11	**94** ± 3	**91.2** ± 6.3
task3-v0	6 ± 2	25 ± 5	36 ± 13	0 ± 1	2 ± 1	**83** ± 18	19 ± 2	27 ± 11	49 ± 49	74 ± 18	**86.3** ± 10.7
task4-v0	0 ± 0	0 ± 1	15 ± 16	1 ± 1	1 ± 1	5 ± 4	1 ± 1	1 ± 2	1 ± 1	3 ± 4	3.0 ± 6.6
task5-v0	2 ± 1	66 ± 4	24 ± 20	1 ± 1	3 ± 3	91 ± 5	31 ± 7	63 ± 9	**98** ± 2	**97** ± 2	**95.8** ± 1.3
humanoidmaze-large-navigate-singletask											
task1-v0	0 ± 0	3 ± 1	2 ± 1	0 ± 0	0 ± 0	1 ± 1	0 ± 0	0 ± 1	6 ± 2	7 ± 6	**57.2** ± 23.7
task2-v0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	**0.1** ± 0.2
task3-v0	1 ± 1	7 ± 3	8 ± 4	3 ± 1	1 ± 1	2 ± 3	1 ± 1	10 ± 2	**48** ± 10	11 ± 7	14.6 ± 16.2
task4-v0	1 ± 0	1 ± 0	1 ± 1	0 ± 0	0 ± 0	0 ± 1	0 ± 0	0 ± 0	1 ± 1	2 ± 3	**3.7** ± 4.1
task5-v0	0 ± 1	1 ± 1	2 ± 2	0 ± 0	0 ± 0	0 ± 0	0 ± 0	1 ± 1	0 ± 0	1 ± 3	**13.1** ± 15.5
antsoccer-arena-navigate-singletask											
task1-v0	2 ± 1	14 ± 5	0 ± 0	44 ± 12	2 ± 1	1 ± 3	22 ± 2	17 ± 3	61 ± 25	**77** ± 4	**77.0** ± 1.7
task2-v0	2 ± 2	17 ± 7	0 ± 1	15 ± 12	3 ± 1	0 ± 0	8 ± 1	8 ± 2	75 ± 3	**88** ± 3	**91.2** ± 2.2
task3-v0	0 ± 0	6 ± 4	0 ± 0	0 ± 0	0 ± 0	8 ± 19	11 ± 5	16 ± 3	14 ± 22	**61** ± 6	51.9 ± 4.9
task4-v0	1 ± 0	3 ± 2	0 ± 0	0 ± 1	0 ± 0	0 ± 0	12 ± 3	24 ± 4	16 ± 9	**39** ± 6	**40.2** ± 4.2
task5-v0	0 ± 0	2 ± 2	0 ± 0	0 ± 0	0 ± 0	0 ± 0	9 ± 2	15 ± 4	0 ± 1	**36** ± 9	29.1 ± 8.9
cube-single-play-singletask											
task1-v0	10 ± 5	88 ± 3	89 ± 5	**95** ± 2	89 ± 7	77 ± 28	81 ± 9	73 ± 33	79 ± 4	**97** ± 2	**99.1** ± 0.4
task2-v0	3 ± 1	85 ± 8	92 ± 4	**96** ± 2	82 ± 16	80 ± 30	81 ± 9	83 ± 13	73 ± 3	**97** ± 2	**99.4** ± 0.7
task3-v0	9 ± 3	91 ± 5	93 ± 3	**99** ± 1	**96** ± 2	**98** ± 1	87 ± 4	82 ± 12	88 ± 4	**98** ± 2	**99.4** ± 0.5
task4-v0	2 ± 1	73 ± 6	92 ± 3	93 ± 4	70 ± 18	91 ± 2	79 ± 6	79 ± 20	79 ± 6	94 ± 3	**99.1** ± 0.7
task5-v0	3 ± 3	78 ± 9	87 ± 8	90 ± 6	61 ± 12	80 ± 20	78 ± 10	76 ± 33	77 ± 7	**93** ± 3	**97.0** ± 1.6
cube-double-play-singletask											
task1-v0	8 ± 3	27 ± 5	45 ± 6	39 ± 19	7 ± 6	21 ± 8	21 ± 7	47 ± 11	35 ± 9	61 ± 9	**76.1** ± 4.6
task2-v0	0 ± 0	1 ± 1	7 ± 3	16 ± 10	0 ± 0	2 ± 2	2 ± 1	22 ± 12	9 ± 5	36 ± 6	**53.3** ± 8.9
task3-v0	0 ± 0	0 ± 0	4 ± 1	17 ± 8	0 ± 1	3 ± 1	1 ± 1	4 ± 2	8 ± 5	22 ± 5	**43.3** ± 8.9
task4-v0	0 ± 0	0 ± 0	1 ± 1	0 ± 1	0 ± 0	0 ± 1	0 ± 0	0 ± 1	1 ± 1	5 ± 2	**7.1** ± 3.1
task5-v0	0 ± 0	4 ± 3	4 ± 2	1 ± 1	0 ± 0	3 ± 2	2 ± 1	2 ± 2	17 ± 6	19 ± 10	**56.3** ± 11.3
scene-play-singletask											
task1-v0	19 ± 6	94 ± 3	**95** ± 2	**100** ± 0	94 ± 4	**100** ± 1	87 ± 8	**96** ± 8	**98** ± 3	**100** ± 0	**99.8** ± 0.4
task2-v0	1 ± 1	12 ± 3	50 ± 13	33 ± 14	2 ± 2	50 ± 40	18 ± 8	46 ± 10	0 ± 0	76 ± 9	**89.0** ± 4.1
task3-v0	1 ± 1	32 ± 7	55 ± 16	**94** ± 4	4 ± 4	49 ± 16	38 ± 9	78 ± 14	54 ± 19	**98** ± 1	78.0 ± 13.2
task4-v0	2 ± 2	0 ± 1	3 ± 3	4 ± 3	0 ± 0	0 ± 0	**6** ± 1	4 ± 4	0 ± 0	5 ± 1	0.6 ± 0.6
task5-v0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0.0 ± 0.0
puzzle-3x3-play-singletask											
task1-v0	5 ± 2	33 ± 6	**97** ± 4	52 ± 12	89 ± 5	**97** ± 2	25 ± 9	63 ± 19	**94** ± 3	90 ± 4	**94.8** ± 4.4
task2-v0	1 ± 1	4 ± 3	1 ± 1	0 ± 1	0 ± 1	0 ± 0	4 ± 2	2 ± 2	1 ± 2	**16** ± 5	0.3 ± 0.3
task3-v0	1 ± 1	3 ± 2	3 ± 1	0 ± 0	0 ± 0	0 ± 0	1 ± 0	1 ± 1	0 ± 0	**10** ± 3	0.9 ± 0.6
task4-v0	1 ± 1	2 ± 1	2 ± 1	0 ± 0	0 ± 0	0 ± 0	1 ± 1	2 ± 2	0 ± 0	**16** ± 5	5.4 ± 2.1
task5-v0	1 ± 0	3 ± 2	5 ± 3	0 ± 0	0 ± 0	0 ± 0	1 ± 1	2 ± 2	0 ± 0	**16** ± 3	14.1 ± 8.4
puzzle-4x4-play-singletask											
task1-v0	1 ± 1	12 ± 2	26 ± 4	**48** ± 5	24 ± 9	44 ± 10	1 ± 2	32 ± 9	**49** ± 9	34 ± 8	**50.0** ± 8.1
task2-v0	0 ± 0	7 ± 4	12 ± 4	14 ± 5	0 ± 1	0 ± 0	0 ± 1	5 ± 3	4 ± 4	**16** ± 5	9.9 ± 2.5
task3-v0	0 ± 0	9 ± 3	15 ± 3	34 ± 5	21 ± 10	29 ± 12	1 ± 1	20 ± 10	**50** ± 14	18 ± 5	46.2 ± 3.8
task4-v0	0 ± 0	5 ± 2	10 ± 3	**26** ± 6	7 ± 4	1 ± 1	0 ± 0	5 ± 1	21 ± 11	11 ± 3	17.2 ± 2.5
task5-v0	0 ± 0	4 ± 1	7 ± 3	**24** ± 11	1 ± 1	0 ± 0	0 ± 1	4 ± 3	2 ± 2	7 ± 3	7.3 ± 3.6
Average (50 tasks)	2.8	23.4	30.9	22.8	14.2	24.7	16.1	28.0	30.0	43.5	**51.8**

Table 11: **Offline RL full results on D4RL.** For each task, models were trained with 8 random seeds and evaluated at the end of training. Reported values are the average normalized scores over the final 100 evaluation episodes, with ± denoting the standard deviation across seeds. *Italics* indicate scores from prior work (Fu et al., 2020; Fujimoto & Gu, 2021; Tarasov et al., 2023; Park et al., 2025), and bold denotes values within 95% of the best performance. GFP actor π_θ is our primary policy, while GFP VaBC π_ω is reported as a byproduct of training.

Task	Offline RL algorithms							
	BC	CQL	IQL	TD3 + BC	ReBRAC	FQL	GFP actor π_ω	GFP VaBC π_θ
D4RL antmaze-umaze	*55*	*74.0*	*87.5*	*78.6*	***97.8*** ± 1.0	***96*** ± 2	**96.8** ± 1.9	**94.9** ± 2.0
D4RL antmaze-umaze-diverse	*47*	*84.0*	*62.2*	*71.4*	***88.3*** ± 13.0	***89*** ± 5	**91.9** ± 2.7	90.1 ± 3.8
D4RL antmaze-medium-play	*0*	*61.2*	*71.2*	*3.0*	***84.0*** ± 4.2	*78* ± 7	**81.9** ± 5.2	57.4 ± 9.1
D4RL antmaze-medium-diverse	*1*	*53.7*	*70.0*	*10.6*	***76.3*** ± 13.5	*71* ± 13	61.6 ± 20.9	45.6 ± 9.5
D4RL antmaze-large-play	*0*	*15.8*	*39.6*	*0.0*	*60.4* ± 26.1	***84*** ± 7	**82.6** ± 5.4	62.6 ± 8.8
D4RL antmaze-large-diverse	*0*	*14.9*	*47.5*	*0.2*	*54.4* ± 25.1	***83*** ± 4	**84.1** ± 5.4	70.6 ± 4.7
D4RL pen-human-v1	*71*	*37.5*	*71.5*	*81.8*	***103.5***	*53*	64.6 ± 5.4	67.4 ± 6.9
D4RL pen-cloned-v1	*52*	*39.2*	*37.3*	*61.4*	***91.8***	*74*	77.1 ± 10.4	70.5 ± 4.2
D4RL pen-expert-v1	*110*	*107.0*	*133.6*	*146*	***154.1***	*142*	140.4 ± 4.7	123.2 ± 5.4
D4RL door-human-v1	*2*	***9.9***	*4.3*	*−0.1*	*0.0*	*0.0*	0.3 ± 0.3	4.1 ± 2.4
D4RL door-cloned-v1	*−0*	*0.4*	***1.6***	*0.1*	*1.1*	*2*	**1.6** ± 1.9	0.6 ± 0.6
D4RL door-expert-v1	***105***	***101.5***	***105.3***	*84.6*	***104.6***	***104***	**104.1** ± 0.6	**103.1** ± 0.9
D4RL hammer-human-v1	*3*	***4.4***	*1.4*	*0.4*	*0.2*	*1*	**4.4** ± 4.9	2.5 ± 1.1
D4RL hammer-cloned-v1	*1*	*2.1*	*2.1*	*0.8*	*6.7*	***11***	**12.4** ± 5.4	2.5 ± 0.9
D4RL hammer-expert-v1	*127*	*86.7*	***129.6***	*117.0*	***133.8***	*125*	123.6 ± 2.0	116.6 ± 4.1
D4RL relocate-human-v1	*0*	*0.20*	*0.1*	*−0.2*	*0.0*	*0*	**0.5** ± 0.3	0.0 ± 0.0
D4RL relocate-cloned-v1	*−0*	*−0.1*	*−0.2*	*−0.1*	***1.9***	*−0*	1.6 ± 0.7	0.1 ± 0.1
D4RL relocate-expert-v1	***108***	*95.0*	***106.5***	***107.3***	***106.6***	***107***	**103.2** ± 3.7	**104.0** ± 3.1

Table 12: **Offline RL full results on Minari.** For each task, models were trained using 8 different random seeds, and evaluation was performed at the end of training. The reported values represent the average normalized score, computed over the final 100 evaluation episodes and averaged across the 8 seeds. GFP actor π_θ represents our primary policy, while GFP VaBC π_ω is reported as a byproduct of our training procedure.

Task	Offline RL algorithms		
	FQL	GFP actor π_θ	GFP VaBC π_ω
Minari pen-human-v2	11.5 ± 4.9	50.1 ± 6.3	**54.4** ± 6.7
Minari pen-cloned-v2	41.8 ± 3.7	**60.4** ± 4.2	54.0 ± 5.5
Minari pen-expert-v2	92.7 ± 6.2	**115.9** ± 4.5	108.7 ± 2.8
Minari door-human-v2	1.1 ± 0.5	0.4 ± 0.1	**2.7** ± 1.9
Minari door-cloned-v2	**0.4** ± 0.2	0.3 ± 0.3	0.1 ± 0.1
Minari door-expert-v2	**102.0** ± 0.9	94.1 ± 20.9	99.0 ± 10.3
Minari hammer-human-v2	1.0 ± 0.6	2.7 ± 0.8	**3.1** ± 0.8
Minari hammer-cloned-v2	1.0 ± 0.6	**22.9** ± 19.5	5.9 ± 4.1
Minari hammer-expert-v2	121.2 ± 4.2	**130.2** ± 5.4	119.0 ± 6.7
Minari relocate-human-v2	−0.0 ± 0.0	**0.1** ± 0.2	−0.0 ± 0.1
Minari relocate-cloned-v2	0.0 ± 0.0	**0.3** ± 0.2	0.0 ± 0.0
Minari relocate-expert-v2	**103.7** ± 1.2	**102.1** ± 5.3	**105.9** ± 1.2
Minari halfcheetah-simple-v0	59.2 ± 0.3	**72.5** ± 0.5	64.4 ± 0.4
Minari halfcheetah-medium-v0	100.2 ± 6.7	**121.3** ± 5.8	108.6 ± 5.3
Minari halfcheetah-expert-v0	**134.1** ± 2.3	**133.4** ± 1.3	**136.4** ± 0.8
Minari hopper-simple-v0	57.2 ± 5.9	**91.6** ± 4.3	**87.4** ± 6.6
Minari hopper-medium-v0	**81.9** ± 23.9	**79.6** ± 14.5	**78.2** ± 24.2
Minari hopper-expert-v0	99.6 ± 10.9	**103.9** ± 10.6	**108.8** ± 13.1
Minari walker2d-simple-v0	**89.4** ± 0.7	**90.0** ± 0.9	**89.7** ± 0.9
Minari walker2d-medium-v0	**127.6** ± 3.0	**133.7** ± 1.1	126.1 ± 3.5
Minari walker2d-expert-v0	**148.0** ± 1.8	**149.9** ± 2.0	**150.8** ± 0.4

Table 13: **Task-specific hyperparameters for offline RL on OGBench.**

Task Category	Offline RL algorithms				
	IQL α	ReBRAC $(\alpha_1,\ \alpha_2)$	FQL α	GFP (**ours**) $(\alpha,\ \eta)$	GFP-AWR Sec. B.3 $(\alpha,\ \eta)$
antmaze-large-navigate-singletask-task{1,2,3,4,5}-v0	1e+1	(1e−2, 1e−2)	1e+1	(3e−1, 1e−4)	–
antmaze-large-stitch-singletask-task{1,2,3,4,5}-v0	1e+1	(1e−2, 1e−2)	3e+0	(3e−2, 1e−6)	(1e−1, 3e−1)
antmaze-large-explore-singletask-task{1,2,3,4,5}-v0	1e+0	(1e−3, 1e−1)	1e+0	(1e−2, 1e−6)	–
antmaze-large-giant-singletask-task{1,2,3,4,5}-v0	–	(1e−2, 1e−2)	3e+1	(1e−1, 1e−1)	(1e−1, 3e−1)
humanoidmaze-medium-navigate-singletask-task{1,2,3,4,5}-v0	–	(1e−2, 1e−2)	3e+1	(3e−1, 1e−3)	(3e−1, 3e−1)
humanoidmaze-medium-stitch-singletask-task{1,2,3,4,5}-v0	1e+1	(1e−2, 1e−2)	1e+2	(3e−1, 1e−3)	(3e−1, 1e−1)
humanoidmaze-large-navigate-singletask-task{1,2,3,4,5}-v0	–	(1e−2, 0e+0)	1e+2	(3e−1, 1e−4)	(3e−1, 1e−1)
antsoccer-arena-navigate-singletask-task{1,2,3,4,5}-v0	–	(1e−2, 0e+0)	–	(1e−1, 1e−2)	–
antsoccer-arena-stitch-singletask-task{1,2,3,4,5}-v0	1e+0	(1e−2, 1e−3)	1e+1	(1e−1, 1e−2)	(1e−1, 1e−1)
cube-single-play-singletask-task{1,2,3,4,5}-v0	1	–	–	(1e+1, 1e−1)	–
cube-single-noisy-singletask-task{1,2,3,4,5}-v0	3e+0	(1e−1, 1e−1)	3e+1	(1e+1, 1e−3)	–
cube-double-play-singletask-task{1,2,3,4,5}-v0	–	(1e−1, 3e−1)	–	(1e+0, 1e−2)	(1e+0, 1e−1)
cube-double-noisy-singletask-task{1,2,3,4,5}-v0	3e−1	(1e−2, 1e−2)	1e+1	(1e−1, 1e−4)	(1e−1, 1e−1)
cube-triple-play-singletask-task{1,2,3,4,5}-v0	1e+0	(1e−1, 1e−3)	3e+2	(1e+0, 1e−5)	(1e+0, 3e+0)
cube-triple-noisy-singletask-task{1,2,3,4,5}-v0	3e+0	(1e−2, 0e+0)	1e+1	(1e−1, 1e−5)	(3e−2, 1e+0)
scene-play-singletask-task{1,2,3,4,5}-v0	–	(1e−1, 1e−3)	–	(1e+1, 1e−3)	–
scene-noisy-singletask-task{1,2,3,4,5}-v0	1e+1	(3e−3, 0e+0)	3e+1	(1e+0, 1e−4)	–
puzzle-3×3-play-singletask-task{1,2,3,4,5}-v0	–	–	–	(3e+0, 1e−3)	(3e+0, 1e+0)
puzzle-4×4-play-singletask-task{1,2,3,4,5}-v0	–	(1e−1, 0e+0)	–	(3e+0, 1e−5)	(1e+0, 3e−1)
puzzle-4×4-noisy-singletask-task{1,2,3,4,5}-v0	1e+0	(3e−2, 1e−2)	3e+2	(3e+0, 1e−3)	(3e−1, 1e−1)
visual-cube-single-play-singletask-task1-v0	–	–	–	(1e+1, 1e−1)	–
visual-cube-double-play-singletask-task1-v0	–	–	–	(3e−1, 1e−2)	–
visual-scene-play-singletask-task1-v0	–	–	–	(1e+1, 1e−3)	–
visual-puzzle-3x3-play-singletask-task1-v0	–	–	–	(3e+0, 1e−2)	–
visual-puzzle-4x4-play-singletask-task1-v0	–	–	–	(1e+0, 1e−4)	–

Table 14: **Task-specific hyperparameters for offline RL on D4RL and Minari.**

Task	GFP (**ours**) $(\alpha,\ \eta)$
D4RL antmaze-umaze	$(1e{-}1, 1e{-}3)$
D4RL antmaze-umaze-diverse	$(1e{-}1, 1e{-}3)$
D4RL antmaze-medium-play	$(3e{-}2, 1e{-}3)$
D4RL antmaze-medium-diverse	$(3e{-}2, 1e{-}3)$
D4RL antmaze-large-play	$(3e{-}2, 1e{-}5)$
D4RL antmaze-large-diverse	$(3e{-}2, 1e{-}5)$
D4RL pen-human-v1	$(3e{+}0, 1e{-}4)$
D4RL pen-cloned-v1	$(3e{+}0, 1e{-}5)$
D4RL pen-expert-v1	$(1e{+}0, 1e{-}3)$
D4RL door-human-v1	$(1e{+}1, 1e{-}2)$
D4RL door-cloned-v1	$(1e{+}1, 1e{-}2)$
D4RL door-expert-v1	$(1e{+}1, 1e{-}2)$
D4RL hammer-human-v1	$(1e{+}1, 1e{-}5)$
D4RL hammer-cloned-v1	$(1e{+}1, 1e{-}5)$
D4RL hammer-expert-v1	$(1e{+}1, 1e{-}2)$
D4RL relocate-human-v1	$(1e{+}1, 1e{-}4)$
D4RL relocate-cloned-v1	$(1e{+}1, 1e{-}4)$
D4RL relocate-expert-v1	$(1e{+}1, 1e{-}4)$

(a) D4RL

Task	FQL α	GFP (**ours**) $(\alpha,\ \eta)$
Minari pen-human-v2	1e+4	$(3e{+}0, 1e{-}6)$
Minari pen-cloned-v2	3e+3	$(1e{+}0, 1e{-}4)$
Minari pen-expert-v2	1e+3	$(3e{-}1, 1e{-}6)$
Minari door-human-v2	3e+4	$(1e{-}2, 1e{-}2)$
Minari door-cloned-v2	3e+4	$(3e{-}2, 1e{-}6)$
Minari door-expert-v2	3e+4	$(3e{+}0, 1e{-}3)$
Minari hammer-human-v2	3e+4	$(1e{+}0, 1e{-}2)$
Minari hammer-cloned-v2	3e+4	$(3e{+}0, 1e{-}5)$
Minari hammer-expert-v2	1e+4	$(1e{+}0, 1e{-}5)$
Minari relocate-human-v2	3e+3	$(3e{+}0, 1e{-}5)$
Minari relocate-cloned-v2	3e+4	$(3e{+}0, 1e{-}4)$
Minari relocate-expert-v2	3e+4	$(3e{+}0, 1e{-}3)$
Minari halfcheetah-simple-v0	1e+2	$(3e{-}2, 1e{-}6)$
Minari halfcheetah-medium-v0	1e+2	$(1e{-}1, 1e{-}3)$
Minari halfcheetah-expert-v0	1e+3	$(3e{+}0, 1e{-}2)$
Minari hopper-simple-v0	3e+2	$(3e{+}0, 1e{-}3)$
Minari hopper-medium-v0	3e+2	$(3e{+}0, 1e{-}3)$
Minari hopper-expert-v0	1e+3	$(3e{+}0, 1e{-}2)$
Minari walker2d-simple-v0	1e+3	$(3e{+}0, 1e{-}1)$
Minari walker2d-medium-v0	3e+2	$(3e{-}1, 1e{-}2)$
Minari walker2d-expert-v0	3e+2	$(3e{+}0, 1e{-}3)$

(b) Minari