VALUE-GUIDED REPULSION FOR OFFLINE REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Drifting models have emerged as a highly efficient and stable paradigm for policy design in offline reinforcement learning (RL). However, standard drift policies determine repulsion solely from geometric relationships among generated actions and therefore cannot distinguish them by expected value. To address this limitation, we introduce Value-Guided Repulsion Q-Learning (VRQL) to integrate action value estimates into the particle interaction field. For each state, VRQL combines geometric kernel distance with a normalized Q value scaling. This mechanism assigns greater repulsive influence to particles with lower values while strictly retaining attraction toward behavior action. Consequently, this value-guided interaction field aligns behavior regularization with policy improvement without adding an auxiliary behavior model stage. On 50 OGBench task configurations and 6 D4RL AntMaze configurations, VRQL demonstrates highly competitive performance against established baselines, suggesting that value-guided particle interaction is an effective method for offline RL.

1 INTRODUCTION

Offline reinforcement learning (RL) aims to learn effective policies from static datasets (Levine et al., 2020). As modern datasets scale and aggregate heterogeneous sources of varying quality, they exhibit complex and multimodal behavioral distributions. This complexity requires expressive policies to accurately model the data support and enforce behavioral constraints (Fujimoto & Gu, 2021; Kostrikov et al., 2022; Wang et al., 2023).

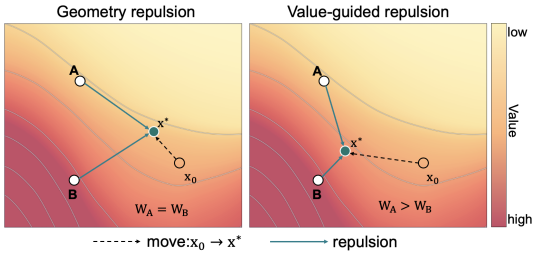


Figure 1: (Left) Standard drift policies rely on geometric distance, assigning identical repulsive weights ($W_A = W_B$) to equidistant neighbors particle A and B. (Right) VRQL incorporates critic estimates into the interaction field. The lower valued neighbor A exerts a stronger repulsive force ($W_A > W_B$), pushing the generated particle x_0 toward the higher value region at x^*. Background indicate the Q value landscape.

Generative models, such as diffusion and flow matching approaches, provide the expressiveness required to capture complex, multimodal action distributions (Ho et al., 2020; Lipman et al., 2023; Liu et al., 2023; Albergo & Vanden-Eijnden, 2023; Tong et al., 2024) in offline RL. However, the iterative sampling process incurs high inference costs, and directly training flow or diffusion policies proves difficult due to unstable backpropagation through time (Wang et al., 2023; Park et al., 2025b). A common strategy adopts a two stage pipeline: first fitting a generative behavior model, and then using its samples to extract a separate actor (Chen et al., 2023; Hansen-Estruch et al., 2023; Park et al., 2025b). Furthermore, because offline datasets contain suboptimal trajectories, these decoupled methods weight the training data using estimated Q values to bias the actor toward high return actions (Peng et al., 2020; Ding et al., 2024; NGUIMATSIA TIOFACK et al., 2026). While effective at injecting value guidance, these two stage pipelines are inherently indirect and risk introducing compounding approximation errors.

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

By directly constructing the policy without relying on iterative sampling or auxiliary models, drifting models provide a promising alternative that sidesteps both inference inefficiencies and two stage approximation errors (Deng et al., 2026; Houssaini et al., 2026). These models train a conditional generator using an interaction field defined by attraction toward dataset actions and repulsion among generated particles to maintain diversity. However, while drifting models address the structural bottlenecks of two stage pipelines, they currently lack a mechanism for value-aware target shaping. Standard drift policies compute repulsive influence relying solely on geometric relationships. Consequently, particles at similar distances exert identical repulsive forces, even if their corresponding critic estimates differ significantly (Figure 1, left). **Thus, a fundamental limitation of existing drift policies is their inability to incorporate estimated action values into the policy constraint, thereby hindering policy improvement.**

To address this limitation, we propose VRQL, which integrates action values into the generated action interaction field. For each state, VRQL modulates the repulsion logits by combining kernel distance with a normalized value bias. Under this design, particles with lower values exert stronger repulsive influence, whereas those with higher values exert less forces (Figure 1, right), while the original attraction toward behavior action remains strictly preserved. To ensure stable optimization, we adaptively calibrate this value signal using a shared scale derived from the typical within state value variation. Thus, VRQL maintains a direct value maximization objective while injecting value awareness into the particle interaction. This integration dynamically reshapes the behavioral constraint within the generative process, avoiding the need for an auxiliary behavior model.

In summary, our primary contributions are:

- **Value-Guided Repulsion:** We propose a value-guided repulsion mechanism. By leveraging estimated action values and geometric proximity, VRQL modulates the influence of generated neighbors particles within the drifting policy.
- **Adaptive Value Calibration:** We introduce adaptive value calibration to control the strength of value guidance using the typical within state variation of particle values.
- **Extensive Empirical Validation:** We evaluate VRQL across 50 OGBench tasks and 6 D4RL AntMaze environments (Fu et al., 2021; Park et al., 2025a), demonstrating that our method yields highly competitive performance in the offline RL setting.

2 PRELIMINARIES

2.1 POLICY CONSTRAINED OFFLINE REINFORCEMENT LEARNING

We consider a discounted Markov decision process $(\mathcal{S}, \mathcal{A}, P, r, \gamma)$ (Sutton et al., 1998), where $\mathcal{S}$ and $\mathcal{A} \subseteq \mathbb{R}^d$ are the state and action spaces, P is the transition kernel, r is the reward function, and $\gamma \in [0, 1)$ is the discount factor. We assume $\mathcal{A}$ is a bounded action box and denote coordinatewise projection onto it by $\Pi_{\mathcal{A}}$. Given an initial state distribution, a policy π seeks to maximize $J(\pi) = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)]$. Its action value function $Q^\pi(s, a)$ is the expected discounted return conditioned on starting with $(s_0, a_0) = (s, a)$ and subsequently following π.

Offline learning uses only a fixed dataset $\mathcal{D} = \{(s, a, r, s', m)\}$ collected by a behavior policy μ, where m is a continuation mask (Levine et al., 2020). A policy constrained offline objective combines value improvement with a soft behavioral regularizer:

$$\max_{\pi} \quad \mathbb{E}_{s\sim\mathcal{D},\, a\sim\pi(\cdot|s)} \left[Q^\pi(s, a)\right] - \lambda\, \mathcal{R}_{\mathcal{D}}(\pi, \mu), \qquad \lambda > 0. \tag{1}$$

Here $\mathcal{R}_{\mathcal{D}}$ denotes a behavioral regularizer supported by the dataset; it need not be a hard equality or a fixed divergence to μ. This soft formulation permits controlled movement within the learned action distribution while discouraging unsupported actions whose critic estimates may be unreliable (Fujimoto et al., 2019; Kumar et al., 2020).

In practice, the value term is estimated with an ensemble of H critics, with $[H] = \{1, \ldots, H\}$, online parameters $\{\phi_h\}_{h\in[H]}$, and target parameters $\{\bar{\phi}_h\}_{h\in[H]}$. For a next-state action sampled from the current actor, $x_\theta(s', z')$, we define the target as

$$y = r + \gamma m \min_{h\in[H]} Q_{\bar{\phi}_h}(s', x_\theta(s', z')). \tag{2}$$

The corresponding Bellman regression is

$$\mathcal{L}_{\text{critic}} = \frac{1}{H} \sum_{h \in [H]} \mathbb{E}_{(s,a,r,s',m) \sim \mathcal{D},\, z'} \left[(Q_{\phi_h}(s,a) - y)^2 \right]. \tag{3}$$

For actor updates, let $\mathcal{Q}_\phi(s,a) = \text{Agg}_{h \in [H]}\, Q_{\phi_h}(s,a)$ denote the chosen ensemble aggregation. The target critics are updated by Polyak averaging. VRQL leaves these standard value estimation components unchanged and modifies the behavioral regularizer $\mathcal{R}_\mathcal{D}$.

2.2 Drifting Models

Drifting models train a direct conditional generator $x_\theta(s,z) = \Pi_\mathcal{A}(f_\theta(s,z))$, where $z \sim \mathcal{N}(0, I)$, by regressing its outputs toward stopped targets constructed from attractive and repulsive sample interactions (Deng et al., 2026). The noise induces the stochastic policy $\pi_\theta(\cdot \mid s)$, although the generator is deterministic for fixed (s,z). Given generated particles x_i and a set of positive samples $\mathcal{P}(s)$, a positive kernel k_τ with bandwidth parameter τ produces normalized attraction and repulsion weights. A common choice is a Laplace or Gaussian kernel scaled by the action dimension d,

$$k_\tau^{\text{L}}(x,y) = \exp\left(-\frac{\|x-y\|_2}{\tau\sqrt{d}} \right), \qquad k_\tau^{\text{G}}(x,y) = \exp\left(-\frac{\|x-y\|_2^2}{2\tau^2 d} \right). \tag{4}$$

With $w_{iu}^+ \propto k_\tau(x_i, u)$ normalized over $u \in \mathcal{P}(s)$ and $w_{ij}^- \propto k_\tau(x_i, x_j)$ normalized over $j \neq i$, the total drift vector V_i aggregates the attraction vector A_i pulling toward the behavior data, and the repulsion vector R_i pushing away from neighboring particles:

$$A_i = \sum_{u \in \mathcal{P}(s)} w_{iu}^+ (u - x_i), \qquad R_i = \sum_{j \neq i} w_{ij}^- (x_j - x_i), \qquad V_i = A_i - R_i. \tag{5}$$

The generator is trained toward a projected, stopped target t_i:

$$t_i = \text{sg}[\Pi_\mathcal{A}(x_i + \eta V_i)], \qquad \mathcal{L}_{\text{drift}}(\theta) = \mathbb{E}\left[\frac{1}{d} \|x_i - t_i\|_2^2 \right], \tag{6}$$

where $\eta > 0$ is a drift step size. The targets are recomputed as training proceeds and held fixed when differentiating the regression loss. Standard drifting models use geometry alone in w_{ij}^-; VRQL changes this negative side weighting while retaining the attraction construction.

3 Methods

Our method VRQL builds on drifting policies: it retains the actor network and behavior action anchor while redefining how generated particles repel each other. Algorithm 1 summarizes the complete training procedure. Specifically, for a given state, the actor generates a particle set, and the critic evaluates these particles. Then, a value signal combines with pairwise geometric distances to determine the repulsion weights. The resulting interaction field forms the drift target for actor optimization. Thus, VRQL reshapes the behavioral target through value-guided interactions while preserving the value maximization objective (Houssaini et al., 2026). In this section, we consider a drift minibatch containing B_d transitions and use $b \in \{1, \cdots, B_d\}$ to index the transitions .

3.1 Behavior Action Anchor and Particle Generation

For each state s_b, we sample N independent noise vectors z_b^i and generate the particle set $\{x_b^i\}$:

$$x_b^i = x_\theta(s_b, z_b^i), \qquad i \in \{1, \cdots, N\}. \tag{7}$$

The dataset behavior action provides a positive anchor $a_b^+ = \Pi_\mathcal{A}(a_b)$. The positive set in Eq. 5 reduces to the singleton $\mathcal{P}(s_b) = \{a_b^+\}$, yielding the normalized attraction field A_b^i:

$$A_b^i = a_b^+ - x_b^i. \tag{8}$$

At inference, the actor outputs actions directly. Value guidance acts only on generated neighbor interactions during training, leaving the attraction mechanism and inference architecture unchanged.

3.2 Adaptive Value Calibration

We define the value used to weight particle j as:

$$q_b^j = \mathrm{sg}\Big[\mathrm{Agg}_{h\in[H]}\, Q_{\bar{\phi}_h}(s_b, x_b^j)\Big]. \tag{9}$$

VRQL uses the target critic ensemble with minimum aggregation to compute this signal. The critic aggregation rules are configurable, as specified in Appendix A.

Because raw Q value magnitudes vary significantly across states and environments, using uncalibrated values makes the relative strength of value guidance sensitive. To ensure stable optimization, we center the Q values within each state using the mean $\bar{q}_b = \frac{1}{N}\sum_{j=1}^{N} q_b^j$ and apply a shared scale for the centered differences across the minibatch. The default scale, $S_{\mathrm{norm}} = S_{\mathrm{std}}$, is the average within state standard deviation:

$$S_{\mathrm{std}} = \frac{1}{B_d}\sum_{b=1}^{B_d}\sqrt{\frac{1}{N}\sum_{j=1}^{N}(q_b^j - \bar{q}_b)^2 + \epsilon}, \tag{10}$$

where ϵ is a constant added for numerical stability. Centering subtracts the same value from all particle logits within each state and therefore leaves the softmax weights unchanged. We use it to remove common value offsets before computing the logits. The shared scale S_{std} calibrates the strength of value guidance relative to typical within state value variation. Recomputing this scale at each training step adapts the calibration as the critic evolves. A shared scale also preserves differences in value dispersion across states, whereas normalizing each state separately can amplify small value differences in states with low variance. Section 4.3 evaluates alternative scaling schemes.

3.3 Value-Guided Repulsion

We measure geometric proximity using the Laplace kernel in Eq. 4 by default, denoted as k_τ throughout. This geometric term is combined with the calibrated value signal to define the repulsion logit for each distinct pair $j \neq i$:

$$\ell_b^{i,j} = \underbrace{-\frac{\|x_b^i - x_b^j\|_2}{\tau\sqrt{d}}}_{\text{geometry}} \underbrace{-\frac{q_b^j - \bar{q}_b}{\beta S_{\mathrm{norm}}}}_{\text{value}}, \qquad w_b^{i,j} = \frac{\exp(\ell_b^{i,j})}{\sum_{j'\neq i}\exp(\ell_b^{i,j'})}, \quad w_b^{i,i} = 0, \tag{11}$$

where $\beta > 0$ is the guidance temperature. With this scaling, β controls the strength of value guidance relative to typical within state value variation. By substituting the standard geometric weights w_{ij}^- in Eq. 5 with these value-guided weights $w_b^{i,j}$, we define the value-guided repulsive component R_b^i and the combined drift vector V_b^i as:

$$R_b^i = \sum_{j\neq i} w_b^{i,j}(x_b^j - x_b^i), \qquad V_b^i = A_b^i - R_b^i. \tag{12}$$

The diagonal mask prevents self interaction. Since the normalized weights sum to one, the combined drift simplifies to $V_b^i = a_b^+ - \sum_{j\neq i} w_b^{i,j} x_b^j$. Value guidance shifts the neighbor barycenter that determines the update direction. Although the repulsion weights sum to one, value-guided reweighting can change both the direction and magnitude of the resulting drift. Specifically, consider two eligible neighbors $(j, k \neq i)$, their relative repulsive influence on particle i is:

$$\frac{w_b^{i,j}}{w_b^{i,k}} = \frac{k_\tau(x_b^i, x_b^j)}{k_\tau(x_b^i, x_b^k)}\exp\left(-\frac{q_b^j - q_b^k}{\beta S_{\mathrm{norm}}}\right). \tag{13}$$

This formulation combines geometric proximity with value differences. At equal distances, the lower valued neighbor receives greater repulsive weight. At unequal distances, spatial proximity and critic evaluations determine the influence. When particle values are identical or as $\beta \to \infty$, the method recovers geometric repulsion.

Algorithm 1 Value-Guided Repulsion Q-Learning (VRQL)

Input: Dataset $\mathcal{D}$; policy x_θ; critic ensemble $\{Q_{\phi_h}\}_{h\in[H]}$; Laplace kernel temperature τ; guidance temperature β; particle count $N \geq 3$

for each training step **do**
  Sample a minibatch from $\mathcal{D}$ and update the critics $\{\phi_h\}_{h\in[H]}$
  Generate particles $x_b^i \leftarrow x_\theta(s_b, z_b^i),\ z_b^i \sim \mathcal{N}(0, I),\ i \in \{1, \cdots, N\}$
  Compute q_b^j, $\bar{q}_b$, and S_{norm} as in Section 3.2
  $\ell_b^{i,j} \leftarrow -\|x_b^i - x_b^j\|_2/(\tau\sqrt{d}) - (q_b^j - \bar{q}_b)/(\beta S_{\text{norm}}),\ j \neq i$ ▷ value guidance
  $w_b^{i,j} \leftarrow \text{softmax}_{j\neq i}(\ell_b^{i,j}),\ w_b^{i,i} \leftarrow 0$
  $V_b^i \leftarrow (a_b^+ - x_b^i) - \sum_{j\neq i} w_b^{i,j}(x_b^j - x_b^i)$ ▷ attraction − repulsion
  $t_b^i \leftarrow \text{sg}\big[\Pi_{\mathcal{A}}(x_b^i + \eta V_b^i)\big]$
  $\mathcal{L}_{\text{drift}} \leftarrow \mathbb{E}_{b,i}\big[\frac{1}{d}\|x_b^i - t_b^i\|_2^2\big]$
  Update θ by minimizing $\alpha\,\mathcal{L}_{\text{drift}} - \mathbb{E}[\mathcal{Q}_\phi(s, x_\theta(s, z))]$
  Update the target critics $\{\bar{\phi}_h\}_{h\in[H]}$ by Polyak averaging
end for

Output: One step policy π_θ with action $a = x_\theta(s, z)$

Actor optimization objective. The projected drift target $t_b^i = \Pi_{\mathcal{A}}(x_b^i + \eta V_b^i)$ is stopped as in Eq. 6. The actor loss is:

$$\mathcal{L}_{\text{actor}}(\theta) = \alpha\,\mathcal{L}_{\text{drift}}(\theta) - \mathbb{E}_{s\sim\mathcal{D}, z\sim\mathcal{N}(0,I)}\left[\mathcal{Q}_\phi(s, x_\theta(s, z))\right], \tag{14}$$

where $\alpha > 0$ balances behavioral regularization and value improvement. The Q maximization term computes gradients through the online critic ensemble, whereas the values in Eq. 9 are detached before entering the repulsion weights. This stop gradient operation is necessary to prevent gradient paths from flowing through the drift interaction field or its value bias. The direct value objective improves actions through critic gradients, whereas value-guided repulsion uses relative critic estimates to reshape the stopped behavioral target. The two terms therefore incorporate value information through different update mechanisms.

Computational overhead. Value guidance adds one additional batch of critic evaluations for the $B_d N$ generated actions to the $\mathcal{O}(B_d N^2 d)$ pairwise interactions.

4 Experiments

In this section, we evaluate the policy performance against published baselines and investigate how repulsion assignment, normalization, guidance temperature, and particle count affect VRQL.

4.1 Experimental Setup

Benchmarks. We evaluate VRQL on OGBench and D4RL AntMaze (Fu et al., 2021; Park et al., 2025a). OGBench offers 50 single task configurations across 10 categories, each comprising 5 tasks. These environments cover diverse domains spanning locomotion and navigation (*humanoidmaze-medium/large-navigate*, *antmaze-large/giant-navigate*, *antsoccer-arena-navigate*), high dimensional object manipulation (*cube-single-play*, *cube-double-play*, and *scene-play*), and long horizon puzzle solving (*puzzle-3x3-play* and *puzzle-4x4-play*). D4RL AntMaze adds 6 configurations covering three maze sizes (*umaze*, *medium*, *large*) and two offline dataset variants (*play*, *diverse*), explicitly testing the ability to stitch suboptimal trajectories. Detailed hyperparameter configurations are provided in Appendix A.2.

Baselines. We compare VRQL against five classes of offline algorithms: (i) behavior cloning and Gaussian policies, including BC, IQL (Kostrikov et al., 2022), and ReBRAC (Tarasov et al., 2023); (ii) diffusion and consistency models, represented by IDQL (Hansen-Estruch et al., 2023), SRPO, and CAC; (iii) standard flow policies, comprising FQL (Park et al., 2025b), FAWAC, FBRAC, and IFQL; (iv) value-guided policies, such as Q-Flow (Doo et al., 2026) and GFP (NGUIMATSIA TIOFACK et al., 2026), to benchmark alternative value integration strategies; and (v) the direct drift

Table 1: **Offline reinforcement learning results.** VRQL achieves the best or tied for best performance across the OGBench categories and the D4RL AntMaze benchmark. The reported $\pm$ values aggregate the seed standard deviations across individual tasks. This table presents one representative method from each policy class, while Appendix C provides complete results for all fourteen methods. Boldface indicates the best score in each row.

	Gaussian	Diffusion	Flow	Value-Guide	Drift	
Task category	IQL	IDQL	FQL	Q-Flow	DriftQL	**VRQL**
OGBench						
antmaze-large (5 tasks)	53 ± 3	21 ± 5	79 ± 3	88 ± 5	92 ± 4	$\mathbf{97} \pm 1$
antmaze-giant (5 tasks)	4 ± 1	0 ± 0	9 ± 6	40 ± 6	60 ± 2	$\mathbf{74} \pm 4$
humanoidmaze-medium (5 tasks)	33 ± 2	1 ± 0	58 ± 5	83 ± 4	62 ± 2	$\mathbf{88} \pm 8$
humanoidmaze-large (5 tasks)	2 ± 1	1 ± 0	4 ± 2	$\mathbf{8} \pm 2$	5 ± 8	$\mathbf{8} \pm 1$
antsoccer-arena (5 tasks)	8 ± 2	12 ± 4	60 ± 2	55 ± 5	65 ± 2	$\mathbf{71} \pm 2$
cube-single (5 tasks)	83 ± 3	95 ± 2	96 ± 1	95 ± 2	94 ± 3	$\mathbf{99} \pm 1$
cube-double (5 tasks)	6 ± 1	15 ± 6	29 ± 2	36 ± 5	25 ± 2	$\mathbf{58} \pm 3$
scene (5 tasks)	28 ± 1	46 ± 3	56 ± 2	60 ± 2	74 ± 4	$\mathbf{75} \pm 1$
puzzle-3x3 (5 tasks)	9 ± 1	10 ± 2	30 ± 1	49 ± 5	35 ± 3	$\mathbf{54} \pm 2$
puzzle-4x4 (5 tasks)	7 ± 1	29 ± 3	17 ± 2	30 ± 3	27 ± 3	$\mathbf{32} \pm 2$
D4RL						
antmaze (6 tasks)	57	79	84 ± 3	59 ± 1	84 ± 3	$\mathbf{90} \pm 3$

predecessor, DriftQL (Houssaini et al., 2026), which relies strictly on geometric repulsion to isolate the empirical contribution of our value-guided mechanism. Owing to space constraints, Table 1 reports only one representative method from each baseline class: IQL, IDQL, FQL, Q-Flow, and DriftQL. Appendix C presents the per task scores for these together with fourteen methods (i.e., BC, ReBRAC, SRPO, CAC, FAWAC, FBRAC, IFQL and GFP). Baseline scores follow the corresponding papers (Houssaini et al., 2026; Doo et al., 2026; NGUIMATSIA TIOFACK et al., 2026).

Evaluation. For all VRQL evaluations, we train with five independent random seeds per task and run 50 evaluation episodes per checkpoint. OGBench scores correspond to average success rates computed at 800k, 900k, and 1M training steps within each seed; D4RL AntMaze scores use normalized returns measured at 500k training steps. We report per task means and standard deviations across the five seed level scores. Table 1 equally weights all tasks per OGBench category and D4RL AntMaze configuration. Category standard deviations aggregate task-level seed variances. For D4RL IQL and IDQL, only means are reported, as original papers do not standard deviations.

4.2 Main Results

Table 1 evaluates VRQL on OGBench and D4RL, demonstrating its empirical advantages across three distinct aspects. First, compared with purely geometric predecessor DriftQL, VRQL yields substantial gains, raising success rates from 60 to 74 on antmaze-giant and from 25 to 58 on cube-double. Moreover, VRQL achieves higher mean scores than IQL, IDQL and FQL across all evaluated task categories. These results indicate that value guidance improving performance in these environments. Then, VRQL outperforms the alternative value-guided method Q-Flow on most task categories. These comparisons suggest that modulating the particle interaction field is a promising strategy for incorporating value guidance into policy optimization. Finally, on the established D4RL AntMaze benchmark, VRQL reaches a mean score of 90, exceeding all evaluated policy classes, which verifies its efficacy across classical offline continuous control domains.

4.3 Ablation Studies

We evaluate four ablation studies, i.e., repulsion weight assignment, value normalization, guidance temperature, and particle count. To validate that VRQL generalizes across diverse environments, we distribute these evaluations across various tasks from both OGBench and D4RL, following the eval-

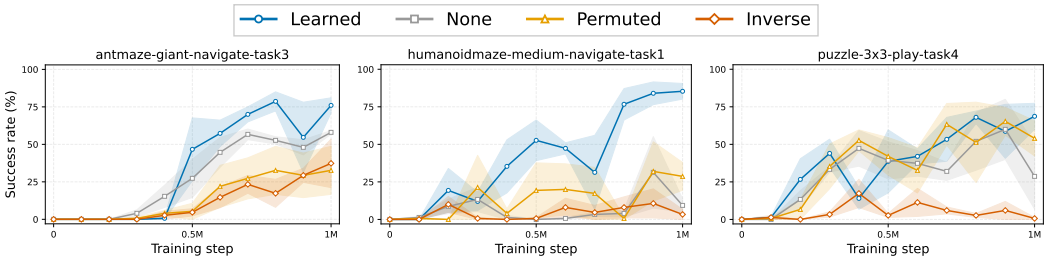


Figure 2: **Effect of repulsion weight assignment.** The *learned* assignment used by VRQL is compared with geometric repulsion (*none*), shuffled value biases (*permuted*), and reversed value biases (*inverse*). Curves show mean success rates, and shaded regions denote one standard deviation.

uation protocol of Section 4.1. The first three use OGBench tasks, the repulsion weight assignment and guidance temperature studies each cover three tasks. While the particle count study adopts three D4RL AntMaze configurations and additionally reports relative training time. All configuration is trained with five independent seeds, and all figures show seed means with shading denoting one standard deviation. Each ablation study varies a single quantity and inherits all remaining hyperparameters from the corresponding task configuration.

Repulsion weight assignment. Figure 2 evaluates the *learned* assignment used by VRQL on three single tasks (*antmaze-giant-navigate*, *humanoidmaze-medium-navigate*, *puzzle-3x3-play*). We compare this mechanism against three control variants: *none* (pure geometric repulsion), *permuted* (shuffled value biases among particles within each state before the softmax operation), and *inverse* (reversed value biases).

This ablation investigates whether the correspondence between estimated action quality and repulsive influence drives policy performance. The Figure 2 shows that the *learned* guidance achieves the highest success rates and fastest convergence across all three tasks. Moreover, disrupting the correct value relationship actively degrades learning, and the *inverse* assignment cause severe performance collapse, plunging success rates near zero in humanoid navigation and puzzle solving. Meanwhile, the *permuted* assignment performs worse than the *none* baseline, demonstrating that random value noise actively disrupts the beneficial geometric structure. These outcomes confirm that the precise alignment of high repulsion with low value actions is essential.

Value normalization. Figure 3 compares three scaling strategies for S_{norm}: the default within state standard deviation S_{std} defined in Eq. 10, the absolute mean S_{abs} define in Eq. 15, and the unnormalized baseline S_{none} (i.e., $S_{\text{none}} = 1$).

$$S_{\text{abs}} = \frac{1}{B_d N} \sum_{b=1}^{B_d} \sum_{j=1}^{N} |q_b^j| + \epsilon. \tag{15}$$

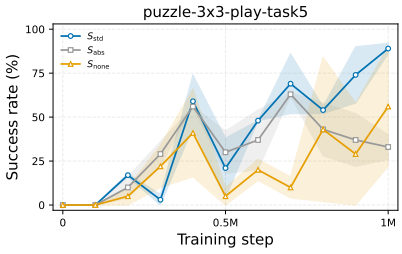


Figure 3: **Value normalization ablation.** Success rates under different normalization strategies.

The results show that calibration by relative value variation using S_{std} yields the highest and most stable success rate. Without normalization S_{none}, training exhibits severe instability and variance, demonstrating that raw value scales disrupt the guidance signal. While absolute value scaling S_{abs} initially matches S_{std}, it suffers from performance degradation later in training state. Thus, calibrating the value term with the within state value spread provides a more reliable guidance scale.

Guidance temperature. Figure 4 evaluates VRQL sensitivity to the guidance temperature $\beta \in \{0.05, 0.25, 1, 4\}$ on three single tasks (*antmaze-giant-navigate*, *humanoidmaze-medium-navigate*, *antsoccer-arena-navigate*). Results indicate that low temperatures ($\beta = 0.05$) disrupt geometric diversity and degrade success rates in antmaze-giant. Conversely, high temperatures ($\beta = 4$) weaken the value signal and reduce performance in humanoid navigation. Thus, a moderate β balances geometric diversity with value guidance.

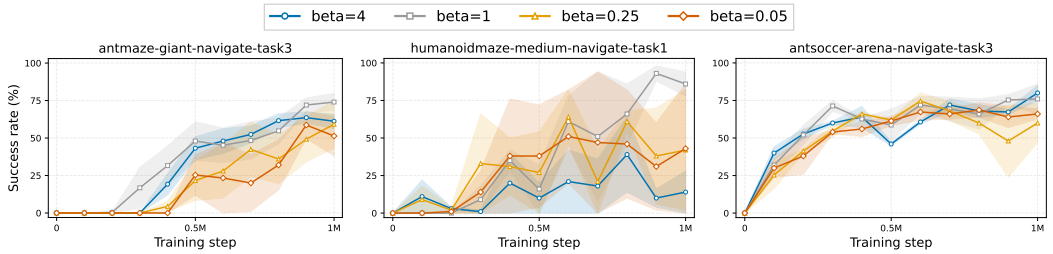


Figure 4: **Sensitivity to the guidance temperature β.** Training curves for varying β on three OGBench tasks. Curves show mean success rates, and shaded regions denote one standard deviation.

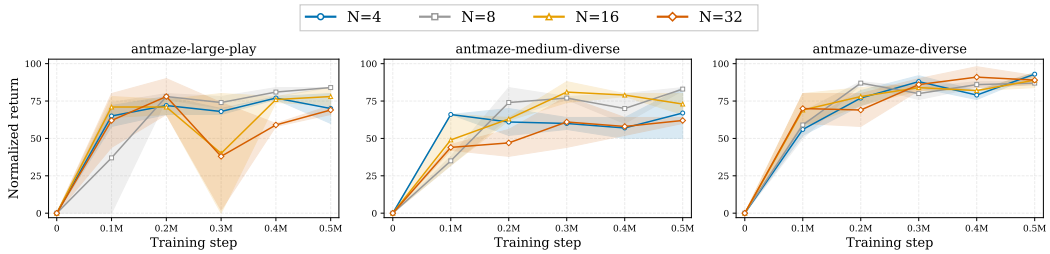


Figure 5: **Sensitivity to the particle count N.** Normalized return on three D4RL AntMaze tasks evaluating sensitivity to the generated particle count N. Curves show mean normalized returns, and shaded regions denote one standard deviation.

Generated action particles. Figure 5 evaluates VRQL sensitivity to the generated particle count N. While $N \geq 3$ is strictly necessary for value guidance to function when self-interactions are excluded, increasing N further expands the repulsive neighborhood at the cost of training computation. The learning curves demonstrate that VRQL sensitivity to the particle count varies by environment. Specifically, performance on *antmaze-large-play* and *antmaze-medium-diverse* exhibits noticeable sensitivity to N, whereas the *antmaze-umaze-diverse* task remains robust across all tested configurations ($N \in \{4, 8, 16, 32\}$). Concurrently, the timing analysis in Figure 6 reveals that VRQL requires selecting the particle count based on specific task complexity to properly balance performance optimization and computational efficiency.

5 Related Work

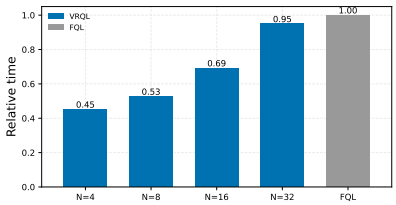


Figure 6: **Relative training time.** VRQL training time scales with particle count.

Offline reinforcement learning. Offline reinforcement learning improves a policy from a fixed dataset without environment interaction, which makes extrapolation error on out of distribution actions the central challenge: the critic assigns unreliable values to actions the data never covers, and unconstrained maximization exploits precisely those errors (Levine et al., 2020). One family of methods constrains the learned policy toward the behavior policy, either by restricting the action support through a generative model of the data or by adding an explicit divergence penalty to the actor objective (Fujimoto et al., 2019; Fujimoto & Gu, 2021). A second family regularizes the value function, penalizing estimates at unobserved actions so that the induced policy stays conservative (Kumar et al., 2020). A third family avoids querying out of distribution actions altogether, fitting values with in sample targets and extracting a policy by weighted regression (Kostrikov et al., 2022; Tarasov et al., 2023), and recent analyses indicate that policy extraction is often the binding constraint on offline performance (Park et al., 2024). VRQL adopts the policy constrained formulation of Eq. 1 and contributes to the regularizer, leaving the value estimation components unchanged.

Generative policies for offline reinforcement learning. Diffusion and flow policies provide expressive conditional action distributions for behavioral modeling and policy extraction (Florence et al., 2022; Janner et al., 2022; Wang et al., 2023; Hansen-Estruch et al., 2023; Park et al., 2025b). A first line of work applies value improvement through an iterative sampler, which can require differentiating through multiple denoising or flow integration steps (Wang et al., 2023; Lu et al., 2023; Doo et al., 2026). A second line first fits an expressive behavior model and then uses its samples or predictions to regularize a separately trained policy (Chen et al., 2023; Hansen-Estruch et al., 2023; Park et al., 2025b). This two stage design separates the policy objective from the behavior model, but introduces an additional model and an approximation error between the two stages. VRQL instead modifies the interactions among actions generated by the policy regularizer itself, within a direct policy training objective. Sampling cost has also been reduced by distilled or few step policy classes (Kang et al., 2023; Ding & Jin, 2024), and entropy or ensemble regularization has been used to preserve action diversity (Zhang et al., 2024).

Value-aware behavioral regularization. Offline methods with value-aware behavioral regularization use estimated action quality to make behavioral fitting selective. Advantage weighted regression and actor-critic policy extraction methods emphasize behavior action with higher estimated advantage, while implicit value based extraction combines in sample value learning with a similarly selective fitting objective (Peters & Schaal, 2007; Peng et al., 2020; Nair et al., 2021; Kostrikov et al., 2022). Value-aware fitting has also been applied to expressive generative policies, where value or advantage estimates reweight flow matching or diffusion targets (Ajay et al., 2023; Ding et al., 2024; Zhang et al., 2025). Auxiliary behavior model methods can use the same signal when fitting a behavioral model and then transfer that model's samples or targets to a separate actor (NGUIMATSIA TIOFACK et al., 2026). These approaches establish that behavioral regularization need not treat all examples equally, but they place the value weight on behavior action, training examples, or model targets. Value-aware fitting methods use critic estimates to prioritize behavioral examples or auxiliary model targets; VRQL uses critic estimates to assign interaction weights to the generated neighbors that define repulsion.

Drifting models and policies. Drifting models train a direct generator by regressing toward stopped targets constructed from attractive and repulsive sample interactions (Deng et al., 2026). A conditional offline policy construction specializes this principle to action generation by combining attraction toward a behavior action, repulsion among generated actions, and a direct actor value objective (Houssaini et al., 2026). VRQL retains this geometric foundation and introduces a negative value dependent bias into the normalized repulsive weights. Kernel similarity determines geometric influence, while relative critic values determine which generated neighbors exert more or less of that influence. The resulting contribution is a value-aware interaction rule inside the drifting regularizer.

6 Conclusion

We introduced VRQL, an offline algorithm that incorporates estimated action values directly into the repulsive interactions of a drifting policy. By combining geometric similarity with value dependent weights, VRQL modulates the influence of generated neighbors while preserving attraction toward behavioral data and maintaining a separate value maximization objective. Furthermore, batch level scaling effectively calibrates this guidance against typical within state value variations. Empirical evaluations across the OGBench and D4RL benchmarks demonstrate that VRQL establishes strong performance and outperforms representative baselines. Ablation studies suggest that these empirical gains are associated with pairing low action values with high geometric repulsion. Disrupting this learned assignment severely degrades policy optimization, whereas proper within state calibration and moderate guidance temperatures ensure stable learning.

However, a primary limitation of this approach stems from its inherent reliance on learned critic estimates. In environments where the critic produces incorrect value orderings, the guidance mechanism can inadvertently repel optimal actions and misguide the policy constraint. Future work will investigate integrating uncertainty estimation into the particle interaction field, allowing the algorithm to dynamically attenuate value guidance when critic reliability degrades. Additionally, future research will explore extending this value-aware mechanism to the behavioral attraction phase, allowing the policy to preferentially anchor to high return data points.

Reproducibility statement

To support reproducibility, the main paper describes the method and training objective, Appendix A provides the network architectures, hyperparameters, benchmark definitions, and compute resources, and the supplementary materials contain anonymized source code.

AI use statement

AI tools were used solely to improve the language and readability of this article. They were not used to generate the method, experimental results, scientific claims, or citations. All AI-assisted edits were reviewed by the authors, who take full responsibility for the final content of this work.

References

Anurag Ajay, Yilun Du, Abhi Gupta, Joshua B. Tenenbaum, Tommi S. Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision making? In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=sP1fo2K9DFG.

Michael Samuel Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=li7qeBbCR1t.

Huayu Chen, Cheng Lu, Chengyang Ying, Hang Su, and Jun Zhu. Offline reinforcement learning via high-fidelity generative behavior modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=42zs3qa2kpy.

Mingyang Deng, He Li, Tianhong Li, Yilun Du, and Kaiming He. Generative modeling via drifting, 2026. URL https://arxiv.org/abs/2602.04770.

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. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 53945–53968. Curran Associates, Inc., 2024. doi: 10.52202/079017-1708. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/6111371a868af8dcfba0f96ad9e25ae3-Paper-Conference.pdf.

Zihan Ding and Chi Jin. Consistency models as a rich and efficient policy class for reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8jdwkUNXb.

JaeHyeok Doo, Byeongguk Jeon, Seonghyeon Ye, Kimin Lee, and Minjoon Seo. Q-flow: Stable and expressive reinforcement learning with flow-based policy. In *Decision-Making from Offline Datasets to Online Adaptation: Black-Box Optimization to Reinforcement Learning*, 2026. URL https://openreview.net/forum?id=payPql0Pk7.

Pete Florence, Corey Lynch, Andy Zeng, Oscar A Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Conference on robot learning*, pp. 158–168. PMLR, 2022.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4{rl}: Datasets for deep data-driven reinforcement learning, 2021. URL https://openreview.net/forum?id=px0-N3_KjA.

Scott Fujimoto and Shixiang (Shane) Gu. A minimalist approach to offline reinforcement learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 20132–20145. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/a8166da05c5a094f7dc03724b41886e5-Paper.pdf.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2052–2062. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/fujimoto19a.html.

Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies, 2023. URL https://arxiv.org/abs/2304.10573.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

Anas Houssaini, Mohamad H. Danesh, Amin Abyaneh, Scott Fujimoto, Hsiu-Chin Lin, and David Meger. Drift q-learning, 2026. URL https://arxiv.org/abs/2606.00350.

Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9902–9915. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/janner22a.html.

Bingyi Kang, Xiao Ma, Chao Du, Tianyu Pang, and Shuicheng Yan. Efficient diffusion policies for offline reinforcement learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 67195–67212. Curran Associates, Inc., 2023. doi: 10.52202/075280-2937. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/d45e0bfb5a39477d56b55c0824200008-Paper-Conference.pdf.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=68n2s9ZJWF8.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1179–1191. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/0d2b2061826a5df3221116a5085a6052-Paper.pdf.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems, 2020. URL https://arxiv.org/abs/2005.01643.

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

Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

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 Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 22825–22855. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/lu23d.html.

Ashvin Nair, Murtaza Dalal, Abhishek Gupta, and Sergey Levine. {AWAC}: Accelerating online reinforcement learning with offline datasets, 2021. URL https://openreview.net/forum?id=OJiM1R3jAtZ.

Franki NGUIMATSIA TIOFACK, Théotime Le Hellard, Fabian Schramm, Nicolas Perrin-Gilbert, and Justin Carpentier. Guided flow policy: Learning from high-value actions in offline reinforcement learning. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 30601–30625, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/33f131b806a93d376cf0ce1a456464bb-Paper-Conference.pdf.

Seohong Park, Kevin Frans, Sergey Levine, and Aviral Kumar. Is value learning really the main bottleneck in offline rl? In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 79029–79056. Curran Associates, Inc., 2024. doi: 10.52202/079017-2509. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/8ffb4e3118280a66b192b6f06e0e2596-Paper-Conference.pdf.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. OGBench: Benchmarking offline goal-conditioned RL. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=M992mjgKzI.

Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 48104–48127. PMLR, 13–19 Jul 2025b. URL https://proceedings.mlr.press/v267/park25f.html.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage weighted regression: Simple and scalable off-policy reinforcement learning, 2020. URL https://openreview.net/forum?id=H1gdF34FvS.

Jan Peters and Stefan Schaal. Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pp. 745–750, New York, NY, USA, 2007. Association for Computing Machinery. ISBN 9781595937933. doi: 10.1145/1273496.1273590. URL https://doi.org/10.1145/1273496.1273590.

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

Denis Tarasov, Vladislav Kurenkov, Alexander Nikulin, and Sergey Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 11592–11620. Curran Associates, Inc., 2023. doi: 10.52202/075280-0511. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/26cce1e512793f2072fd27c391e04652-Paper-Conference.pdf.

Alexander Tong, Kilian FATRAS, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=CD9Snc73AW. Expert Certification.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=AHvFDPi-FA.

Ruoqi Zhang, Ziwei Luo, Jens Sjölund, Thomas B. Schön, and Per Mattsson. Entropy-regularized diffusion policy with q-ensembles for offline reinforcement learning. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 98871–98897. Curran Associates, Inc., 2024. doi: 10.52202/

079017-3138. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/b319c0e8092ba726cb22e718d7d9a95e-Paper-Conference.pdf.

Shiyuan Zhang, Weitong Zhang, and Quanquan Gu. Energy-weighted flow matching for offline reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=HA0oLUvuGI.

A Implementation Details

This section describes the implementation of VRQL, including the network architectures, hyperparameter settings, and benchmark environments used in the experiments.

A.1 Network Architectures

We parameterize the policy generator f_θ and the critic ensemble $\{Q_{\phi_h}\}_{h=1}^{H}$ using multilayer perceptrons (MLPs). The actor and critics have separate parameters and are trained with Adam.

Actor network (f_θ). The actor consists of four hidden layers with 512 units each and GELU activations, without layer normalization. Its inputs are the state s and a Gaussian noise vector $z \sim \mathcal{N}(0, I)$ whose dimension equals the action dimension. The output is clipped coordinatewise to the action box $\mathcal{A} = [-1, 1]^d$, giving $x_\theta(s, z) = \Pi_{\mathcal{A}}(f_\theta(s, z))$. Sampling z induces the policy $\pi_\theta(\cdot \mid s)$.

Critic networks (Q_{ϕ_h}). We use an ensemble of $H = 2$ critics. Each critic takes a state and an action as input and consists of four hidden layers with 512 units each, GELU activations, and layer normalization. The actor objective uses the mean of the online critic estimates, whereas Bellman targets use the minimum of the target critic estimates, as specified in Eqs. 14 and 2. By default, particle value guidance also uses the minimum of the target critic estimates. Target parameters are updated by Polyak averaging with coefficient ρ.

A.2 Hyperparameters

The shared training settings for VRQL are summarized in Table 2, and the value guidance settings are listed in Table 3. All runs use the Laplace kernel defined in Eq. 4. We use τ for the kernel temperature and ρ for the target network update coefficient.

Table 2: Shared hyperparameters. Symbols refer to the main text where one is defined.

Hyperparameter	**Symbol**	**Value**
General training settings		
Optimizer	–	Adam
Actor / critic learning rate	–	3×10^{-4}
Batch size	–	256
Hidden dimensions (actor and critic)	–	$(512, 512, 512, 512)$
Activation	–	GELU
Critic ensemble size	H	2
Target update coefficient	ρ	0.005
Drifting policy settings		
Particle count	N	32
Drift step size	η	1.0
Drift batch size	B_d	256
Drift distance stabilizer	–	10^{-12}
Training duration		
Offline gradient steps (OGBench)	–	1,000,000
Offline gradient steps (D4RL AntMaze)	–	500,000

Task specific hyperparameters. Following the hyperparameter protocol of DriftQL (Houssaini et al., 2026), we retain its general training conventions and adapt the settings needed for value-guided repulsion. The behavioral regularization coefficient α controls the balance between the drift regression loss and direct value maximization in Eq. 14. We tune the guidance temperature β and the kernel temperature τ by task. Across OGBench, the settings span $\beta \in [0.05, 10]$, and $\tau \in [0.02, 0.8]$. D4RL AntMaze uses $\beta = 1$. The per task configurations are released with the code.

Table 3: Value-guided repulsion settings. These are the defaults of Section 3.3; the ablations of Section 4.3 vary them one at a time.

Hyperparameter	**Default**	**Alternatives**
Value scale S_{norm}	S_{std} (Eq. 10)	S_{abs}, S_{none} (Eq. 15)
Ensemble aggregation Agg	$\min$	mean
Critic for the value bias	target	online
Value scale stabilizer ϵ	10^{-6}	–

A.3 Environment Details

We evaluate VRQL on 56 task configurations: 50 from OGBench and 6 from D4RL AntMaze. All experiments train on fixed offline datasets.

OGBench. The benchmark covers navigation, object manipulation, and puzzle solving. Navigation tasks include *antmaze-large-navigate*, *antmaze-giant-navigate*, *humanoidmaze-medium-navigate*, *humanoidmaze-large-navigate*, and *antsoccer-arena-navigate*. Manipulation and puzzle tasks include *cube-single-play*, *cube-double-play*, *scene-play*, *puzzle-3x3-play*, and *puzzle-4x4-play*. Each of these ten families has five single task configurations. The full identifier appends *-singletask-task$i$-v0*, for $i \in \{1, 2, 3, 4, 5\}$, to the family prefix. Table 1 abbreviates these prefixes by omitting the *-navigate* or *-play* suffix.

D4RL AntMaze. The six configurations cover three maze sizes and different dataset variants: *antmaze-umaze-v2*, *antmaze-umaze-diverse-v2*, *antmaze-medium-play-v2*, *antmaze-medium-diverse-v2*, *antmaze-large-play-v2*, and *antmaze-large-diverse-v2*. These tasks evaluate navigation using offline trajectories with different maze layouts and dataset compositions.

Both benchmarks use a bounded action box $\mathcal{A} = [-1, 1]^d$, so the projection $\Pi_{\mathcal{A}}$ is a coordinatewise clip. We report success rate on OGBench and normalized return on D4RL AntMaze, following the protocol of Section 4.1 and the conventions of the respective benchmarks (Fu et al., 2021; Park et al., 2025a).

B Compute Resources

All experiments run on a machine equipped with an AMD 9950 CPU and a single NVIDIA RTX 5090 GPU, with one task per device, in a JAX implementation. A single OGBench run takes approximately 0.5 GPU hours for $1{,}000{,}000$ gradient steps, and a single D4RL AntMaze run approximately 0.2 GPU hours for $500{,}000$ steps; both figures include periodic policy evaluation. With five seeds per configuration, the 50 OGBench and 6 D4RL configurations of Table 1 amount to roughly 125 and 6 GPU hours respectively, excluding the ablations of Section 4.3 and the hyperparameter search.

C Detailed benchmark results

The tables below give the complete baseline inventory and the per task results underlying the main comparison. OGBench rows are grouped by the identifier prefixes of Appendix A.3 and named by their *task$i$-v0* suffix; D4RL rows are the 6 AntMaze v2 tasks listed above, named by their dataset qualifier (*div* abbreviates *diverse*). The fourteen methods are split across two tables, seven each.

Table 4: Full results across all evaluated OGBench and D4RL tasks, methods one to seven, Boldface marks the best of all fourteen methods on a task, a row's best may appear in Table 5, which gives the remaining seven methods.

	Gaussian			Diffusion			Flow
Task	BC	IQL	ReBRAC	IDQL	SRPO	CAC	FAWAC
antmaze-large-navigate-singletask							
task1-v0 ⋆	0 ± 0	48 ± 9	91 ± 10	0 ± 0	0 ± 0	42 ± 7	1 ± 1
task2-v0	6 ± 3	42 ± 6	88 ± 4	14 ± 8	4 ± 4	1 ± 1	0 ± 1
task3-v0	29 ± 5	72 ± 7	51 ± 18	26 ± 8	3 ± 2	49 ± 10	12 ± 4
task4-v0	8 ± 3	51 ± 9	84 ± 7	62 ± 25	45 ± 19	17 ± 6	10 ± 3
task5-v0	10 ± 3	54 ± 22	90 ± 2	2 ± 2	1 ± 1	55 ± 6	9 ± 5
antmaze-giant-navigate-singletask							
task1-v0 ⋆	0 ± 0	0 ± 0	27 ± 22	0 ± 0	0 ± 0	0 ± 0	0 ± 0
task2-v0	0 ± 0	1 ± 1	16 ± 17	0 ± 0	0 ± 0	0 ± 0	0 ± 0
task3-v0	0 ± 0	0 ± 0	34 ± 22	0 ± 0	0 ± 0	0 ± 0	0 ± 0
task4-v0	0 ± 0	0 ± 0	5 ± 12	0 ± 0	0 ± 0	0 ± 0	0 ± 0
task5-v0	1 ± 1	19 ± 7	49 ± 22	0 ± 1	0 ± 0	0 ± 0	0 ± 0
humanoidmaze-medium-navigate-singletask							
task1-v0 ⋆	1 ± 0	32 ± 7	16 ± 9	1 ± 1	0 ± 0	38 ± 19	6 ± 2
task2-v0	1 ± 0	41 ± 9	18 ± 16	1 ± 1	1 ± 1	47 ± 35	40 ± 2
task3-v0	6 ± 2	25 ± 6	36 ± 13	0 ± 1	2 ± 1	83 ± 18	19 ± 2
task4-v0	0 ± 0	0 ± 1	15 ± 16	1 ± 1	1 ± 1	5 ± 4	1 ± 1
task5-v0	2 ± 1	66 ± 4	24 ± 20	1 ± 1	3 ± 3	91 ± 5	31 ± 7
humanoidmaze-large-navigate-singletask							
task1-v0 ⋆	0 ± 0	3 ± 1	2 ± 1	0 ± 0	0 ± 0	1 ± 1	0 ± 0
task2-v0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0
task3-v0	1 ± 1	7 ± 3	8 ± 4	3 ± 1	1 ± 1	2 ± 3	1 ± 1
task4-v0	1 ± 0	1 ± 0	1 ± 1	0 ± 0	0 ± 0	0 ± 1	0 ± 0
task5-v0	0 ± 1	1 ± 1	2 ± 2	0 ± 0	0 ± 0	0 ± 0	0 ± 0
antsoccer-arena-navigate-singletask							
task1-v0	2 ± 1	14 ± 6	0 ± 0	44 ± 12	2 ± 1	1 ± 3	22 ± 2
task2-v0	2 ± 2	17 ± 7	0 ± 1	15 ± 12	3 ± 1	0 ± 0	8 ± 1
task3-v0	0 ± 0	6 ± 4	0 ± 0	0 ± 0	0 ± 0	8 ± 19	11 ± 5
task4-v0 ⋆	1 ± 0	3 ± 2	0 ± 0	0 ± 1	0 ± 0	0 ± 0	12 ± 3
task5-v0	0 ± 0	2 ± 2	0 ± 0	0 ± 0	0 ± 0	0 ± 0	9 ± 2
cube-single-play-singletask							
task1-v0	10 ± 5	88 ± 3	89 ± 5	95 ± 2	89 ± 7	77 ± 28	81 ± 9
task2-v0 ⋆	3 ± 1	85 ± 8	92 ± 4	96 ± 2	82 ± 16	80 ± 30	81 ± 9
task3-v0	9 ± 3	91 ± 5	93 ± 3	99 ± 1	96 ± 2	98 ± 1	87 ± 4
task4-v0	2 ± 1	73 ± 6	92 ± 3	93 ± 4	70 ± 18	91 ± 2	79 ± 6
task5-v0	3 ± 3	78 ± 8	90 ± 8	90 ± 6	61 ± 12	80 ± 20	78 ± 10
cube-double-play-singletask							
task1-v0	8 ± 3	27 ± 5	45 ± 6	39 ± 19	7 ± 6	21 ± 8	21 ± 7
task2-v0 ⋆	0 ± 0	1 ± 1	7 ± 3	16 ± 10	0 ± 0	2 ± 2	2 ± 1
task3-v0	0 ± 0	0 ± 0	4 ± 1	17 ± 8	0 ± 1	3 ± 1	1 ± 1
task4-v0	0 ± 0	0 ± 0	1 ± 1	0 ± 1	0 ± 0	0 ± 0	0 ± 0
task5-v0	0 ± 0	4 ± 3	4 ± 2	1 ± 1	0 ± 0	3 ± 2	2 ± 1

Continued on next page

Table 4 (continued)

	Gaussian			Diffusion			Flow
Task	BC	IQL	ReBRAC	IDQL	SRPO	CAC	FAWAC
scene-play-singletask							
task1-v0	19 ± 6	94 ± 3	95 ± 2	**100** ± 0	94 ± 4	**100** ± 1	87 ± 8
task2-v0 $\star$	1 ± 1	12 ± 3	50 ± 13	33 ± 14	2 ± 2	50 ± 40	18 ± 8
task3-v0	1 ± 1	32 ± 7	55 ± 16	94 ± 4	4 ± 4	49 ± 16	38 ± 9
task4-v0	2 ± 2	0 ± 1	3 ± 3	4 ± 3	0 ± 0	0 ± 0	6 ± 1
task5-v0	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
task2-v0	1 ± 1	4 ± 3	1 ± 1	0 ± 1	0 ± 1	0 ± 0	4 ± 2
task3-v0	1 ± 1	3 ± 2	3 ± 1	0 ± 0	0 ± 0	0 ± 0	1 ± 1
task4-v0 $\star$	1 ± 1	2 ± 1	2 ± 1	0 ± 0	0 ± 0	0 ± 0	1 ± 1
task5-v0	1 ± 0	3 ± 2	5 ± 3	0 ± 0	0 ± 0	0 ± 0	1 ± 1
puzzle-4x4-play-singletask							
task1-v0	1 ± 1	12 ± 2	26 ± 4	48 ± 5	24 ± 9	44 ± 10	1 ± 2
task2-v0	0 ± 0	7 ± 4	12 ± 4	14 ± 5	0 ± 1	0 ± 0	0 ± 1
task3-v0	0 ± 0	9 ± 3	15 ± 3	34 ± 5	21 ± 10	29 ± 12	1 ± 1
task4-v0 $\star$	0 ± 0	5 ± 2	10 ± 3	**26** ± 6	7 ± 4	1 ± 1	0 ± 0
task5-v0	0 ± 0	4 ± 1	7 ± 3	**24** ± 11	1 ± 1	0 ± 0	0 ± 1
antmaze-$*$-v2 (D4RL)							
umaze	55	77	98	94	97	66 ± 5	90 ± 6
umaze-div	47	54	84	80	82	66 ± 11	55 ± 7
medium-play	0	66	**90**	84	81	49 ± 24	52 ± 12
medium-div	1	74	84	**85**	75	0 ± 1	44 ± 15
large-play	0	42	52	64	54	0 ± 0	10 ± 6
large-div	0	30	64	68	54	0 ± 0	16 ± 10

Table 5: Full results across all evaluated OGBench and D4RL tasks, methods eight to fourteen, continuing Table 4 under the same conventions; boldface again marks the best of all fourteen methods on a task.

	Flow			Value-Guide		Drift	
Task	FBRAC	IFQL	FQL	Q-Flow	GFP	DriftQL	**VRQL**
antmaze-large-navigate-singletask							
task1-v0 ⋆	70 ± 20	24 ± 17	80 ± 8	95 ± 4	95 ± 0	95 ± 2	$\mathbf{99} \pm 1$
task2-v0	35 ± 12	8 ± 3	57 ± 10	85 ± 6	$\mathbf{92} \pm 3$	85 ± 9	91 ± 0
task3-v0	83 ± 15	52 ± 17	93 ± 3	93 ± 4	95 ± 2	97 ± 1	$\mathbf{99} \pm 1$
task4-v0	37 ± 18	18 ± 8	80 ± 4	79 ± 23	90 ± 2	91 ± 1	$\mathbf{97} \pm 2$
task5-v0	76 ± 8	38 ± 18	83 ± 4	90 ± 6	95 ± 1	92 ± 1	$\mathbf{97} \pm 1$
antmaze-giant-navigate-singletask							
task1-v0 ⋆	0 ± 1	0 ± 0	4 ± 5	15 ± 10	12 ± 15	32 ± 2	$\mathbf{56} \pm 14$
task2-v0	4 ± 7	0 ± 0	9 ± 7	34 ± 22	52 ± 26	$\mathbf{79} \pm 1$	69 ± 9
task3-v0	0 ± 0	0 ± 0	0 ± 1	9 ± 5	13 ± 10	43 ± 2	$\mathbf{67} \pm 7$
task4-v0	9 ± 4	0 ± 0	14 ± 23	68 ± 13	17 ± 19	64 ± 3	$\mathbf{87} \pm 4$
task5-v0	6 ± 10	13 ± 9	16 ± 28	73 ± 12	43 ± 35	85 ± 5	$\mathbf{89} \pm 3$
humanoidmaze-medium-navigate-singletask							
task1-v0 ⋆	25 ± 8	69 ± 19	19 ± 12	$\mathbf{87} \pm 5$	83 ± 3	28 ± 2	$\mathbf{87} \pm 12$
task2-v0	76 ± 10	85 ± 11	94 ± 3	95 ± 4	91 ± 6	87 ± 2	$\mathbf{100} \pm 0$
task3-v0	27 ± 11	49 ± 49	74 ± 18	95 ± 3	86 ± 10	56 ± 3	$\mathbf{99} \pm 2$
task4-v0	1 ± 2	1 ± 1	3 ± 4	39 ± 21	3 ± 6	39 ± 3	$\mathbf{56} \pm 37$
task5-v0	63 ± 9	98 ± 2	97 ± 2	98 ± 2	95 ± 1	99 ± 2	$\mathbf{100} \pm 0$
humanoidmaze-large-navigate-singletask							
task1-v0 ⋆	0 ± 0	6 ± 2	7 ± 6	14 ± 7	$\mathbf{57} \pm 23$	2 ± 1	13 ± 3
task2-v0	0 ± 0	0 ± 0	0 ± 0	0 ± 1	0 ± 0	0 ± 0	0 ± 0
task3-v0	10 ± 2	$\mathbf{48} \pm 10$	11 ± 7	16 ± 5	14 ± 16	23 ± 1	24 ± 1
task4-v0	1 ± 1	1 ± 1	2 ± 3	$\mathbf{5} \pm 5$	3 ± 4	1 ± 1	0 ± 1
task5-v0	1 ± 1	0 ± 0	1 ± 3	$\mathbf{5} \pm 4$	3 ± 15	1 ± 0	1 ± 1
antsoccer-arena-navigate-singletask							
task1-v0	17 ± 3	61 ± 25	77 ± 4	73 ± 9	77 ± 1	79 ± 4	$\mathbf{81} \pm 2$
task2-v0	8 ± 2	75 ± 3	88 ± 3	94 ± 5	91 ± 2	91 ± 2	$\mathbf{98} \pm 3$
task3-v0	16 ± 3	14 ± 22	61 ± 6	58 ± 9	51 ± 4	60 ± 3	$\mathbf{66} \pm 1$
task4-v0 ⋆	24 ± 4	16 ± 9	39 ± 6	27 ± 11	40 ± 4	48 ± 5	$\mathbf{50} \pm 1$
task5-v0	15 ± 4	0 ± 1	36 ± 9	25 ± 15	29 ± 8	48 ± 9	$\mathbf{58} \pm 8$
cube-single-play-singletask							
task1-v0	73 ± 33	79 ± 4	97 ± 2	95 ± 3	99 ± 0	94 ± 3	$\mathbf{100} \pm 1$
task2-v0 ⋆	83 ± 13	73 ± 3	97 ± 2	98 ± 2	$\mathbf{99} \pm 0$	93 ± 2	$\mathbf{99} \pm 1$
task3-v0	82 ± 12	88 ± 4	98 ± 2	98 ± 2	99 ± 0	95 ± 2	$\mathbf{100} \pm 0$
task4-v0	79 ± 20	79 ± 6	94 ± 3	90 ± 5	$\mathbf{99} \pm 0$	92 ± 3	97 ± 2
task5-v0	76 ± 33	77 ± 7	93 ± 3	92 ± 5	97 ± 1	90 ± 3	$\mathbf{98} \pm 1$
cube-double-play-singletask							
task1-v0	47 ± 11	35 ± 9	61 ± 9	67 ± 11	76 ± 4	49 ± 1	$\mathbf{95} \pm 3$
task2-v0 ⋆	22 ± 12	9 ± 5	36 ± 6	27 ± 10	53 ± 8	23 ± 5	$\mathbf{68} \pm 1$
task3-v0	4 ± 2	8 ± 5	22 ± 5	28 ± 10	43 ± 8	9 ± 3	$\mathbf{58} \pm 11$
task4-v0	0 ± 1	1 ± 1	5 ± 2	$\mathbf{9} \pm 5$	7 ± 3	3 ± 1	$\mathbf{9} \pm 1$
task5-v0	2 ± 2	17 ± 6	19 ± 10	48 ± 20	56 ± 11	43 ± 2	$\mathbf{60} \pm 10$

Continued on next page

Table 5 (continued)

	Flow			Value-Guide		Drift	
Task	FBRAC	IFQL	FQL	Q-Flow	GFP	DriftQL	**VRQL**
scene-play-singletask							
task1-v0	96 ± 8	98 ± 3	$\mathbf{100} \pm 0$	$\mathbf{100} \pm 0$	99 ± 0	$\mathbf{100} \pm 0$	$\mathbf{100} \pm 0$
task2-v0 ⋆	46 ± 10	0 ± 0	76 ± 9	$\mathbf{96} \pm 6$	89 ± 4	89 ± 3	91 ± 3
task3-v0	78 ± 14	54 ± 19	$\mathbf{98} \pm 1$	96 ± 4	78 ± 13	93 ± 3	95 ± 5
task4-v0	4 ± 4	0 ± 0	5 ± 1	7 ± 9	0 ± 0	83 ± 10	$\mathbf{87} \pm 2$
task5-v0	0 ± 0	0 ± 0	0 ± 0	0 ± 1	0 ± 0	$\mathbf{2} \pm 2$	$\mathbf{2} \pm 2$
puzzle-3x3-play-singletask							
task1-v0	63 ± 19	94 ± 3	90 ± 4	94 ± 4	94 ± 4	87 ± 9	$\mathbf{100} \pm 0$
task2-v0	2 ± 2	1 ± 2	16 ± 5	$\mathbf{60} \pm 17$	0 ± 0	39 ± 2	10 ± 5
task3-v0	1 ± 1	0 ± 0	10 ± 3	$\mathbf{28} \pm 7$	0 ± 0	20 ± 3	6 ± 0
task4-v0 ⋆	2 ± 2	0 ± 0	16 ± 5	35 ± 9	5 ± 2	10 ± 6	$\mathbf{84} \pm 2$
task5-v0	2 ± 2	0 ± 0	16 ± 3	29 ± 9	14 ± 8	19 ± 1	$\mathbf{69} \pm 11$
puzzle-4x4-play-singletask							
task1-v0	32 ± 9	49 ± 9	34 ± 8	54 ± 9	50 ± 8	72 ± 8	$\mathbf{83} \pm 4$
task2-v0	5 ± 3	4 ± 4	16 ± 5	$\mathbf{17} \pm 5$	9 ± 2	4 ± 2	5 ± 0
task3-v0	20 ± 10	50 ± 14	18 ± 5	47 ± 8	46 ± 3	47 ± 10	$\mathbf{61} \pm 7$
task4-v0 ⋆	5 ± 1	21 ± 11	11 ± 3	19 ± 5	17 ± 3	10 ± 3	6 ± 2
task5-v0	4 ± 3	2 ± 2	7 ± 3	11 ± 5	7 ± 3	2 ± 1	3 ± 1
antmaze-∗-v2 (D4RL)							
umaze	94 ± 2	92 ± 6	96 ± 2	98 ± 0	96 ± 1	96 ± 2	$\mathbf{100} \pm 1$
umaze-div	82 ± 6	62 ± 12	89 ± 5	67 ± 0	91 ± 2	86 ± 9	$\mathbf{96} \pm 5$
medium-play	77 ± 7	56 ± 15	78 ± 7	76 ± 2	81 ± 5	81 ± 5	89 ± 5
medium-div	77 ± 6	60 ± 25	71 ± 13	66 ± 1	61 ± 20	75 ± 8	84 ± 10
large-play	32 ± 21	55 ± 9	84 ± 7	17 ± 4	82 ± 5	83 ± 6	$\mathbf{89} \pm 8$
large-div	20 ± 17	64 ± 8	83 ± 4	26 ± 7	84 ± 5	84 ± 6	$\mathbf{85} \pm 5$