HAMILTON–JACOBI–BELLMAN CONSTRAINED FLOW-BASED DISTRIBUTIONAL Q-LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Flow-based distributional critics learn expressive return distributions from transition-level Bellman supervision, but this supervision constrains the local derivatives of the induced value function only indirectly. We introduce HJB-Constrained Value Flows, which augments this learning process with a stationary Hamilton–Jacobi–Bellman (HJB)-inspired differential regularizer. The noise-averaged initial velocity provides an expected-return estimate, motivated by the population flow-matching regression, without integrating the flow to its terminal time. We use this interface to motivate a local prior from an auxiliary unit-cost control problem and implement it through a single-noise stochastic surrogate. The return-flow architecture, distributional targets, and policy-extraction procedure remain unchanged. Experiments on 25 `OGBench` manipulation tasks and 12 `D4RL Adroit` tasks show improvements over the base Value Flows critic, with task-dependent gains. Five offline-to-online evaluations compare the complete training pipelines. Value-landscape and execution analyses illustrate accompanying changes in the critic and policy, while return-spread diagnostics show that the learned distributions retain nontrivial variability. These comparisons support the utility of the complete regularizer for flow-based value learning; they do not isolate its HJB-specific mechanism.

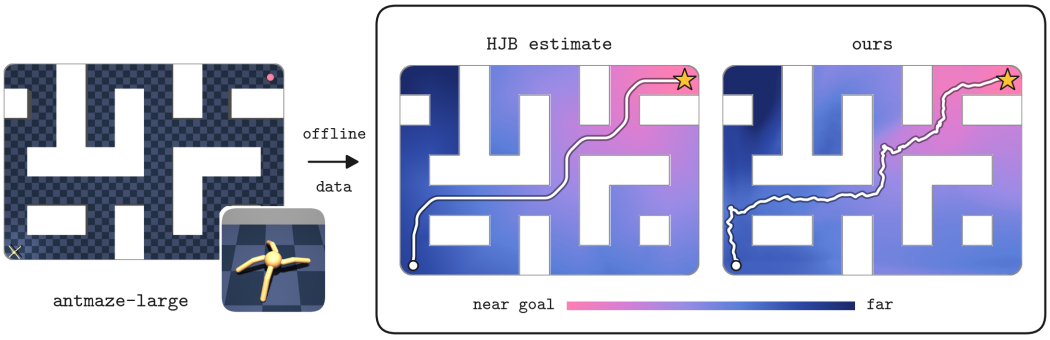


Figure 1: **HJB-inspired regularization of a distributional critic.** A stationary HJB reference for a simplified two-dimensional maze model (left) and the expected-return readout of the learned return flow (right). The comparison illustrates the local value structure targeted by the prior; the reference is not the task value function of the full Ant system.

1 INTRODUCTION

Distributional reinforcement learning models the full law of future returns rather than only their expectation (Bellemare et al., 2017; 2023). Recent generative approaches make this representation increasingly expressive. Value Flows (Dong et al., 2026), in particular, uses conditional flow matching to learn return distributions from observed transitions and bootstrapped distributional Bellman targets.

While representing the full return distribution provides a richer learning objective, its estimation remains dependent on the available transition data. Under finite coverage, Bellman supervision need

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.

not determine the local derivatives of the learned value function around observed state–action pairs. Prior work has shown that differential constraints derived from partial differential equations (PDEs) can provide useful inductive biases for value learning (Giammarino & Qureshi, 2026; Giammarino et al., 2026; Viswanath et al., 2026). The Hamilton–Jacobi–Bellman (HJB) equation supplies one such structure through the dynamic programming principle, relating an optimal value to its directional change under the dynamics and admissible controls (Kim et al., 2021; Seiffertt et al., 2008). This motivates supplementing transition-level supervision with an explicit relation among local value derivatives.

Conditional flow matching provides an interface between the distributional critic and scalar HJB structure: its noise-averaged initial velocity estimates the target mean (Lipman et al., 2022; Frans et al., 2025; Dong et al., 2026). This readout avoids generating complete returns. We use it to motivate a stationary differential prior and optimize a single-noise surrogate alongside the distributional losses.

Our prior is motivated by a canonical minimum-time control problem. Its residual balances value change along a transition with the improvement available through bounded action variation and a unit running cost. It favors a nonzero, dynamics-directed value slope coupled to the action gradient. We adopt this geometry as a soft inductive bias, while the Bellman objectives continue to learn the benchmark's discounted task returns. The auxiliary geometry need not be the governing equation of those returns; its usefulness is evaluated through the complete regularized learner.

We retain the Value Flows architecture and policy extraction, evaluating the method on 25 `OGBench` manipulation tasks (Park et al., 2025), 12 `D4RL Adroit` tasks (Fu et al., 2020), and five offline-to-online tasks. Value-landscape, spread, and execution analyses complement the task scores. Our contribution is this practical integration of distributional Bellman learning and HJB-inspired regularization, together with its empirical evaluation.

2 Related Work

Distributional RL. Distributional RL provides a framework for modeling the full distribution of future returns (Morimura et al., 2010; Bellemare et al., 2017; 2023). For a fixed policy, the distributional Bellman operator has the policy's return law as its fixed point. Common representations use categorical supports (Bellemare et al., 2017) or quantiles (Dabney et al., 2018). Value Flows (Dong et al., 2026) instead models the return law with a flow-matching model (Lipman et al., 2022; Albergo & Vanden-Eijnden, 2022). Its learned vector field provides estimates of return statistics, including the mean and a derivative-based measure of spread. Our approach retains its distributional Bellman objective and adds a differential regularizer motivated by the expected-return readout, with a sample-wise training objective over state–action space.

Hamilton–Jacobi–Bellman equations in RL. In optimal control, HJB equations characterize value functions under suitable assumptions on the dynamics, objective, and boundary conditions (Crandall & Lions, 1983; Bardi et al., 1997). This perspective has motivated continuous-time reinforcement learning for state-value functions (Doya, 2000; Munos, 2000; 1997). Kim et al. (2021) derive an action-dependent HJB equation by restricting controls to be Lipschitz continuous in time, and develop a DQN-like algorithm for the resulting control problem. Wiltzer et al. (2022) develop a distributional HJB framework using finite collections of statistics. More recently, physics-informed value learners have incorporated HJB or Eikonal structure into implicit value learning (Giammarino & Qureshi, 2026; Giammarino et al., 2026; Viswanath et al., 2026). We extend this line of work to flow-based distributional critics through their initial-velocity value readout.

Physics-informed RL and generative models. Physics-informed neural networks (PINNs) incorporate differential equations into function approximation by penalizing their residuals (Karniadakis et al., 2021; Raissi et al., 2019; 2024). Related approaches in offline RL and planning use Eikonal equations to regularize value or time fields (Giammarino & Qureshi, 2026; Giammarino et al., 2026; Viswanath et al., 2026; Ni & Qureshi, 2022; Ni et al., 2025). Physics constraints have also been incorporated into diffusion and flow-matching models (Utkarsh et al., 2026; Baldan et al., 2026; Tauberschmidt et al., 2026), often through generated samples or trajectories. Here, the statistic of interest is accessible at the initial return-flow velocity. This permits differential regularization without

unrolling the flow for the scalar readout. Our method uses a stochastic surrogate of this prior rather than a PDE for the full return distribution.

3 BACKGROUND

Notation and conventions. We consider an MDP with state $s \in \mathcal{S} \subset \mathbb{R}^n$, action $a \in \mathcal{A} \subset \mathbb{R}^m$, bounded reward r, transition kernel p, and discount factor $\gamma \in (0, 1)$. An offline dataset contains transitions $(s, a, r, s') \in \mathcal{D}$. We denote a policy's return by $Z^\pi(s, a)$ and its expectation by $Q^\pi(s, a)$. The learned flow readout is $\hat{Q}_\theta$. We use $\tau \in [0, 1]$ for generative flow time and t for physical time. The optimal value $Q^\star$ introduced in Section 3.3 refers specifically to the stationary unit-cost control problem used to motivate our prior.

3.1 DISTRIBUTIONAL RL

Given a policy π, distributional RL models the conditional return $Z^\pi(s, a) = \sum_{h=0}^{\infty} \gamma^h r(S_h, A_h)$, with $S_0 = s$, $A_0 = a$, $S_{h+1} \sim p(\cdot \mid S_h, A_h)$, and subsequent actions drawn from π, rather than only $Q^\pi(s, a) = \mathbb{E}[Z^\pi(s, a)]$ (Morimura et al., 2010; Bellemare et al., 2017). The distributional Bellman operator is

$$\mathcal{T}^\pi Z(s, a) \overset{d}{=} r(s, a) + \gamma Z(S', A'), \quad S' \sim p(\cdot \mid s, a), \quad A' \sim \pi(\cdot \mid S'), \tag{1}$$

where $\overset{d}{=}$ denotes equality in distribution. For a fixed policy and a uniformly bounded family of first moments, this operator is a γ-contraction in the supremal 1-Wasserstein metric and has the return law as its unique fixed point (Bellemare et al., 2017). In control, the policy used for the next action is updated using the critic; the fixed-policy contraction result alone is not a convergence statement for that joint learning procedure.

3.2 VALUE FLOWS

Value Flows (Dong et al., 2026) represents the conditional return law with a flow-matching model (Lipman et al., 2022; Albergo & Vanden-Eijnden, 2022), in place of categorical or quantile representations. A return vector field $v_\theta : \mathbb{R} \times [0, 1] \times \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ defines a flow ϕ_θ through

$$\frac{d}{d\tau}\phi_\theta(\epsilon \mid \tau, s, a) = v_\theta(\phi_\theta(\epsilon \mid \tau, s, a) \mid \tau, s, a)\,, \qquad \phi_\theta(\epsilon \mid 0, s, a) = \epsilon, \tag{2}$$

where $\epsilon \sim \mathcal{N}(0, 1)$. The law $p_\theta(\cdot \mid s, a)$ of $\phi_\theta(\epsilon \mid 1, s, a)$ is the learned return distribution. For a target sample z, linear conditional flow matching (CFM) uses $z_\tau = (1 - \tau)\epsilon + \tau z$ and the objective

$$\mathcal{L}_{\mathrm{CFM}}(v) = \mathbb{E}\Big[\big(v(z_\tau \mid \tau, s, a) - (z - \epsilon)\big)^2\Big]\,. \tag{3}$$

The expectation includes $\tau \sim \mathrm{U}[0, 1]$ and a joint target–noise distribution with the required marginals. The standard independent coupling is one choice; the bootstrapped construction below can couple z and ϵ.

Direct target-return samples require rollouts of the policy from (s, a), which are generally unavailable for a queried state–action pair in an offline dataset. Value Flows replaces them with bootstrapped targets from a target field $v_{\bar{\theta}}$, maintained by Polyak averaging. For a transition and next action a', let $\bar{z}_\tau = \phi_{\bar{\theta}}(\epsilon \mid \tau, s', a')$. Its distributional conditional flow-matching (DCFM) squared error is

$$\ell_{\mathrm{DCFM}}(\theta; \xi) = \big(v_\theta(r + \gamma \bar{z}_\tau \mid \tau, s, a) - v_{\bar{\theta}}(\bar{z}_\tau \mid \tau, s', a')\big)^2\,, \tag{4}$$

where $\xi = (s, a, r, s', a', \tau, \epsilon)$ collects the training samples. This is a temporal-difference loss on the return field. Because DCFM alone admits degenerate self-consistent solutions, Value Flows also uses bootstrapped CFM (BCFM):

$$\begin{aligned} z_{\mathrm{TD}} &= r + \gamma \phi_{\bar{\theta}}(\epsilon \mid 1, s', a'), & z_{\mathrm{TD},\tau} &= (1 - \tau)\epsilon + \tau z_{\mathrm{TD}}, \\ \ell_{\mathrm{BCFM}}(\theta; \xi) &= \big(v_\theta(z_{\mathrm{TD},\tau} \mid \tau, s, a) - (z_{\mathrm{TD}} - \epsilon)\big)^2\,. \end{aligned} \tag{5}$$

Here the same noise generates the target return and its interpolation (Dong et al., 2026, App. C.1). Together, these losses learn the return law from transition-level, bootstrapped supervision. Target flow states are obtained by Euler integration; the target values are held fixed when updating θ.

3.3 Stationary HJB Structure for Q-Functions

The HJB equation provides local optimal-control structure for a scalar value field. Following the action-rate construction of Kim et al. (2021), consider the extended system

$$\dot{s}(t) = f(s(t), a(t)), \qquad \dot{a}(t) = b(t), \qquad \|b(t)\|_2 \le M. \tag{6}$$

Here the action is part of the extended state and its rate b is the control. To motivate the stationary prior, consider an undiscounted goal-reaching problem with unit running cost. Let Γ be the goal set, t_Γ its first hitting time, and $T^\star(s,a) = \inf_{b(\cdot)} \int_0^{t_\Gamma} 1\, dt$ the minimum time-to-go, with $t_\Gamma = \inf\{t \ge 0 : s(t) \in \Gamma\}$. On reachable states where this value is finite, the dynamic programming principle gives, wherever the value is differentiable and no additional action-bound constraint is active,

$$\begin{aligned} 0 &= 1 + \nabla_s T^\star(s,a)^\top f(s,a) + \min_{\|b\|_2 \le M} \nabla_a T^\star(s,a)^\top b \\ &= 1 + \nabla_s T^\star(s,a)^\top f(s,a) - M\|\nabla_a T^\star(s,a)\|_2. \end{aligned} \tag{7}$$

The boundary condition is $T^\star = 0$ on the goal boundary. The viscosity formulation extends the equation to nonsmooth value functions under the appropriate control and boundary assumptions (Crandall & Lions, 1983; Bardi et al., 1997).

Writing $Q^\star = -T^\star$ in reward-maximization convention gives the stationary equation

$$\mathcal{H}[Q^\star](s,a) := \nabla_s Q^\star(s,a)^\top f(s,a) + M\|\nabla_a Q^\star(s,a)\|_2 - 1 = 0. \tag{8}$$

The first term measures value change along the dynamics; the second is the maximum local improvement allowed by the action-rate bound; the constant is the unit running cost in reward-maximization convention. This auxiliary problem is both stationary and undiscounted. Its $Q^\star$ is distinct from the discounted task-return mean Q^π learned by the Bellman objectives. We use its differential relation as a prior, not as an identification of the two objectives.

4 Method

Value Flows learns a conditional return distribution through transition-level Bellman supervision. Our goal is to complement this objective with structural information about the critic's local value landscape. The two formulations meet at a scalar interface: the initial return-flow velocity provides a first-moment estimate, whose state and action derivatives can be regularized without introducing a separate critic.

At a high level, our critic objective is

$$\mathcal{L}_{\text{critic}}(\theta) = \mathcal{L}_{\text{VF}}(\theta) + \lambda_{\text{H}} \mathcal{L}_{\text{HJB}}(\theta). \tag{9}$$

Here $\mathcal{L}_{\text{VF}}$ is the original weighted Value Flows objective and $\mathcal{L}_{\text{HJB}}$ is the weighted stochastic regularizer defined below. We first expose the mean readout, then derive the analytical prior, and finally specify the sample-wise loss actually optimized.

4.1 Exposing the First Moment of the Return Flow

A distributional critic needs a scalar statistic to connect to the HJB relation. Expected return is the natural choice because it is used to rank actions. Recall the CFM interpolation $z_\tau = (1-\tau)\epsilon + \tau z$ from Eq. (3). At the population squared-error solution,

$$v^\star(z_\tau \mid \tau, s, a) = \mathbb{E}[z - \epsilon \mid z_\tau, \tau, s, a]. \tag{10}$$

Taking the conditional-regression version of this field at $\tau = 0$, where $z_0 = \epsilon$, gives

$$\begin{aligned} v^\star(\epsilon \mid 0, s, a) &= \mathbb{E}[z - \epsilon \mid \epsilon, s, a], \\ \mathbb{E}_\epsilon[v^\star(\epsilon \mid 0, s, a)] &= \mathbb{E}[z \mid s, a]. \end{aligned} \tag{11}$$

The second equality follows from the law of total expectation and zero-mean base noise. It does not require z and ϵ to be independent, and therefore covers BCFM's coupled target construction. Since continuous-time CFM determines the regression field only almost everywhere in τ, this endpoint

identity requires the regression version at $\tau = 0$ or a continuous extension attaining it. Terminal-law matching alone does not imply the identity; see Section A.1.

Following Value Flows, we use the initial-velocity readout

$$\hat{Q}_\theta(s, a) := \mathbb{E}_{\epsilon\sim\mathcal{N}(0,1)}[v_\theta(\epsilon \mid 0, s, a)]. \tag{12}$$

At the corresponding population CFM solution, it equals the first moment of the fitted target. During joint learning, it serves as an expected-return estimate. Unlike averaging complete generated returns, evaluating the integrand requires no integration to $\tau = 1$. When differentiation and noise expectation can be interchanged, its derivatives are expectations of initial-velocity derivatives, giving an interface for differential regularization.

4.2 HJB as Differential Supervision

The Value Flows losses relate return predictions across observed transitions. The stationary HJB relation in Eq. (8) instead describes how a scalar value changes along the dynamics and through admissible action variation. Substituting the readout into that relation gives the analytical prior

$$\mathcal{H}[\hat{Q}_\theta](s, a) = \nabla_s \hat{Q}_\theta(s, a)^\top f(s, a) + M\|\nabla_a \hat{Q}_\theta(s, a)\|_2 - 1. \tag{13}$$

The state derivative is directional, not an isotropic smoothness penalty. The action-gradient term couples this directional change to the local improvement available in the auxiliary action-rate model. The unit term sets a nonzero progress scale. These ingredients motivate a structured regularizer rather than a second Bellman target.

We use this auxiliary geometry as a soft prior on the task critic. In particular, the environment need not impose the auxiliary bound $\|\dot{a}\|_2 \le M$, and M is a regularization coefficient rather than an identified physical limit. The unit cost does not replace the benchmark rewards in DCFM or BCFM. The resulting bias is therefore a modeling choice whose empirical utility is evaluated for the full objective.

Transition-based evaluation. An explicit dynamics model is unnecessary for the local state term. Under a first-order approximation, a transition over Δt gives $f(s, a)\Delta t \approx s' - s$. We measure time in environment steps, set $\Delta s := s' - s$, and use the corresponding per-step coefficient M. This yields

$$\mathcal{R}_\theta(s, a, s') = \nabla_s \hat{Q}_\theta(s, a)^\top \Delta s + M\|\nabla_a \hat{Q}_\theta(s, a)\|_2 - 1. \tag{14}$$

The displacement supplies a sample direction from observed data, without adding simulator access or a learned dynamics network. Finite transitions and hybrid contacts need not satisfy the ideal continuous model exactly; the residual is used locally as a regularizer.

4.3 Joint Distributional and Differential Learning

We retain the Value Flows targets in Eqs. (4) and (5). DCFM supplies field-level temporal-difference supervision, and BCFM anchors it to bootstrapped returns. The HJB-inspired addition is implemented with one Gaussian noise per transition:

$$q^\epsilon_\theta(s, a) := \epsilon + v_\theta(\epsilon \mid 0, s, a), \qquad \mathbb{E}_\epsilon[q^\epsilon_\theta(s, a)] = \hat{Q}_\theta(s, a). \tag{15}$$

This is a noise-conditioned scalar readout, not a terminal-flow sample. Its expectation identity holds by definition and zero-mean noise. Holding ϵ fixed when differentiating gives $\nabla_{s,a} q^\epsilon_\theta = \nabla_{s,a} v_\theta(\epsilon \mid 0, s, a)$. We compute

$$\begin{aligned} \widetilde{\mathcal{R}}_\theta(s, a, s'; \epsilon) &:= \nabla_s q^\epsilon_\theta(s, a)^\top \Delta s + M\|\nabla_a q^\epsilon_\theta(s, a)\|_2 - 1, \\ \ell_{\mathrm{HJB}}(\theta; \xi) &:= \widetilde{\mathcal{R}}_\theta(s, a, s'; \epsilon)^2. \end{aligned} \tag{16}$$

One draw estimates the sample-wise objective in Eq. (16); it does not unbiasedly estimate $\mathcal{R}^2_\theta$. Because the norm and square are nonlinear, it also penalizes dispersion of local residuals across noise-conditioned readouts. Section A.3 gives the exact unweighted decomposition and explains the effect of confidence weighting.

With $\xi = (s, a, r, s', a', \tau, \epsilon)$ denoting the sampled variables, let $w_{\bar{\theta}}(s, a, \epsilon)$ be the inherited Value Flows confidence weight, held fixed during the critic update. The complete objective is

$$\mathcal{L}_{\text{critic}}(\theta) = \mathbb{E}_{\xi}\Big[w_{\bar{\theta}}(s,a,\epsilon)\big(\lambda_{\text{D}}\ell_{\text{DCFM}}(\theta;\xi) + \lambda_{\text{B}}\ell_{\text{BCFM}}(\theta;\xi) + \lambda_{\text{H}}\ell_{\text{HJB}}(\theta;\xi)\big)\Big]. \tag{17}$$

Thus $\mathcal{L}_{\text{HJB}} := \mathbb{E}_{\xi}[w_{\bar{\theta}}\ell_{\text{HJB}}]$ in Eq. (9), and $\mathcal{L}_{\text{VF}}$ contains the first two weighted terms. Setting $\lambda_{\text{H}} = 0$ recovers the original critic objective.

The analytical prior is formulated on a first-moment estimate, but the implemented surrogate also acts on noise-conditioned derivatives. It supplies no explicit target for terminal-return variance or quantiles; nevertheless, both this sample-wise regularization and parameter sharing can change the learned distribution. DCFM and BCFM remain the objectives that directly supervise the return law.

Policy extraction is unchanged: offline actions are sampled from the behavior-cloning flow π_ω and ranked by the critic; online fine-tuning retains the one-step policy π_η (Dong et al., 2026). Only the auxiliary readout avoids flow integration; the inherited target-flow and confidence-weight computations are still used. No separate scalar critic or dynamics model is introduced.

Algorithm 1 HJB-Constrained Value Flows

Input: fields $v_\theta, v_{\bar{\theta}}$, policies π_ω, π_η, dataset $\mathcal{D}$
for each training iteration **do**
  Sample transitions, flow times τ, and one base noise ϵ per transition
  Select next actions and construct the Value Flows targets and weight $w_{\bar{\theta}}$
  Evaluate ℓ_{DCFM} and ℓ_{BCFM} using Eqs. (4) and (5)
  Form q_θ^ϵ and differentiate it with respect to s, a, holding ϵ fixed
  Compute ℓ_{HJB} using the sample-wise residual in Eq. (16)
  Update v_θ using Eq. (17); update $v_{\bar{\theta}}$ by Polyak averaging
  Update the policy components using the original Value Flows procedure
end for

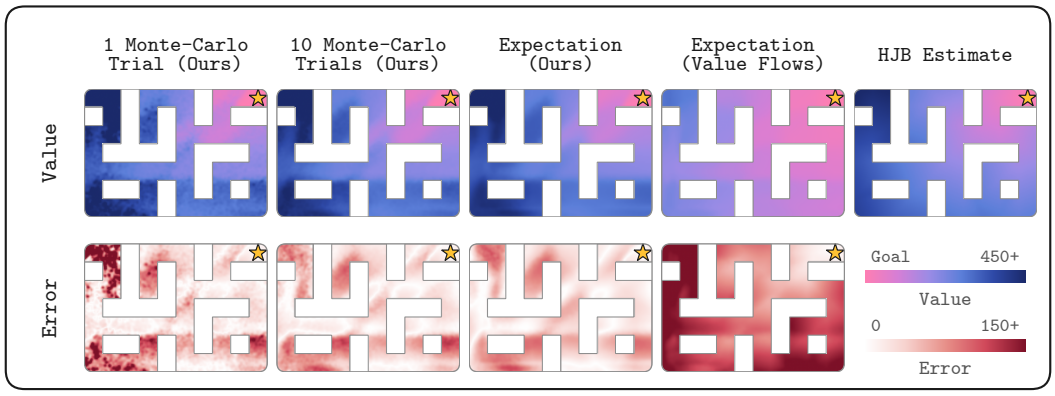


Figure 2: **Value readouts in a two-dimensional AntMaze slice.** Top: one terminal-flow sample, the mean of ten terminal-flow samples, the initial-velocity readout for our critic and Value Flows, and a stationary HJB reference for the simplified maze model. Bottom: pointwise differences from that reference. Other observation coordinates are held fixed. This visualization compares the displayed value landscapes, rather than evaluating the full Ant task against a ground-truth return function.

5 Experiments

We evaluate the regularized learner from four perspectives: offline task performance, final performance of the complete offline-to-online pipeline, the learned value landscape, and execution behavior. The comparisons test the combined method rather than isolating individual residual terms.

Experimental setup. The offline evaluation contains 25 state-based single-task OGBench manipulation tasks: five each from `scene-play`, `puzzle-3x3-play`, `cube-double-play`, `cube-triple-play`, and `puzzle-4x4-play`. We also evaluate 12 D4RL Adroit tasks. The offline-to-online comparison covers the five tasks in table 2, including two navigation tasks. We retain the Value Flows architecture, distributional targets, and policy extraction, adding the loss in Eq. (17) with $M = 10^{-3}$ and $\lambda_{\mathrm{H}} = 0.1$.

Offline tables summarize peak evaluation scores during training. OGBench uses success percentages, whereas D4RL uses normalized returns; these metrics are not pooled into a single overall score. The offline-to-online table reports each method before and after fine-tuning. Per-task offline results and learning curves are provided in Sections C and D. Section B documents the implementation and preliminary hyperparameter-selection convention.

Table 1: Offline RL by domain (Scene, Puzzle-n, and Cube-n abbreviate the corresponding `-play` environments). OGBench entries are success percentages; D4RL entries are normalized returns. Blue denotes results within 95% of the best in a column, and pink marks other improvements over Value Flows. Domain-level results are summarized here, with the corresponding task-level results reported in table 4.

	OGBench: success (%)					D4RL
Method	Scene 5 tasks	Puzzle-3 5 tasks	Cube-2 5 tasks	Cube-3 5 tasks	Puzzle-4 5 tasks	Adroit 12 tasks
BC	5±2	2±1	2±1	0±0	0±0	48
IQL	28±3	9±3	6±2	1±1	7±2	53
ReBRAC	41±7	22±2	12±3	0±0	14±3	**59**
FBRAC	45±5	14±5	15±6	0±0	13±5	50±3
IFQL	30±4	19±1	14±5	0±0	25±8	52±4
FQL	56±2	30±4	29±6	4±2	17±5	52±3
C51	4±1	1±0	2±0	0±0	0±0	48±2
IQN	40±1	15±1	42±8	6±0	27±4	50±3
CODAC	55±1	20±5	61±6	2±1	20±18	52±1
VF	59±4	87±13	**69±4**	14±3	27±4	50±2
+ HJB	**74±5**	**93±14**	**72±5**	**21±1**	**32±4**	55±3

5.1 Offline Reinforcement Learning

Table 1 summarizes domain-level performance, and table 4 gives the individual tasks. The regularized critic improves the reported aggregates on `scene-play`, both cube domains, `puzzle-4x4-play`, and D4RL Adroit. On `scene-play`, the largest task-level gain is task 4, from 7 ± 17 to 70 ± 12. The `puzzle-3x3-play` task entries reach 100 ± 0 on the first four tasks and improve from 58 ± 39 to 65 ± 0 on task 5. On `cube-triple-play`, task 1 increases from 59 ± 12 to 85 ± 1, although success remains low on several other tasks.

The gains are task-dependent rather than universal. For example, `puzzle-4x4-play` task 4 decreases from 28 ± 5 to 25 ± 2, and `scene-play` task 5 remains unsolved. Taken together, these comparisons support the utility of the complete regularizer across the evaluated domains, without establishing which derivative term accounts for the gains.

5.2 Offline-to-Online Fine-Tuning

Table 2 compares the complete pipelines at the end of offline training and after online fine-tuning, using the same reported interaction budget. The regularized variant finishes above Value Flows on four tasks and matches it on `scene-play`. It starts below the baseline on the two navigation tasks but reaches 99 ± 1 on `antmaze-large-navigate` and 87 ± 4 on `humanoidmaze-medium-navigate`. The final cube scores are 96 ± 3 and 100 ± 0.

Each pipeline begins online learning from its own offline-trained model. These results describe end-to-end performance; different offline initializations prevent attributing the final-score differences to the regularizer during online learning alone. The one-point difference on `humanoidmaze` is also small relative to the reported variation.

Table 2: Offline-to-online success percentages before and after fine-tuning. Baselines are from Dong et al. (2026); the comparison uses the same reported online interaction budget. Each method starts from its own offline-trained model, so these are complete-pipeline comparisons.

Task	IQN	IFQL	FQL	RLPD	IQL	VF	**+ HJB**
End of offline training							
antmaze-large-nav	55±5	11±3	87±7	0±0	41±5	72±4	32±8
humanoidmaze-med-nav	23±3	56±12	12±7	0±0	21±5	29±6	15±7
cube-double-play	29±4	12±3	40±11	0±0	0±0	65±7	78±6
cube-triple-play	25±5	2±1	4±1	0±0	3±1	29±8	83±5
scene-play	0±0	0±0	82±11	0±0	15±4	92±23	100±0
After online fine-tuning							
antmaze-large-nav	91±1	3±2	99±12	3±2	45±5	96±1	**99±1**
humanoidmaze-med-nav	14±2	82±7	22±12	8±3	16±3	86±3	**87±4**
cube-double-play	42±7	41±2	92±3	0±0	0±0	79±6	**96±3**
cube-triple-play	14±6	7±5	83±12	0±0	0±0	70±7	**100±0**
scene-play	0±0	60±14	**100±1**	**100±0**	10±3	**100±0**	**100±0**

5.3 The Learned Value Landscape

To examine the critic beyond task scores, we visualize a two-dimensional slice derived from `antmaze-large-navigate`, varying maze position while holding the remaining observation coordinates fixed. Figure 2 compares the readouts with a stationary HJB reference for a simplified maze model. This reference describes the auxiliary geometry, not the ground-truth discounted return of the full Ant system.

Within the displayed slice, averaging ten terminal-flow samples produces a field visually closer to the initial-velocity readout than a single sample does. The regularized readout also varies more clearly along the maze routes than the baseline readout. These observations illustrate the scalar interface and the accompanying geometric change; they are a local diagnostic, not a benchmark-wide validation of the endpoint identity.

Since the stochastic regularizer updates the shared return field, we also examine dispersion. Table 3 shows similar median spread but larger means for the regularized critic, particularly on `puzzle-3x3-play`. Thus the estimated spread remains nonzero while changing under regularization. This supports retained variability, not a claim that the return distributions are unchanged or better calibrated.

Table 3: Return-spread diagnostics using the flow-derivative estimator of Dong et al. (2026), over 2048 dataset states and 8 base noises per domain. Entries summarize the estimated standard deviation across states. Similar medians and larger means indicate retained, but changed, dispersion; these measurements do not assess distributional calibration.

Return s.d.	scene-play	puzzle-3x3-play	cube-double-play
Value Flows, avg	1.19	1.10	0.91
+ HJB, avg	1.82	3.43	1.30
Value Flows, median	0.98	0.82	0.71
+ HJB, median	0.97	0.92	0.72

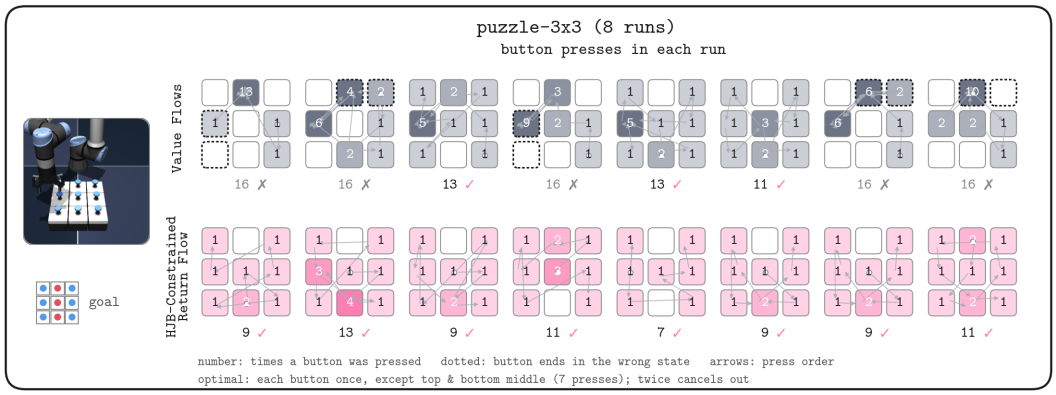


Figure 3: **Execution on `puzzle-3x3-play`.** Each board is one evaluation run. Shading counts presses, arrows give their order, and dotted borders mark buttons left in the wrong state. The task admits a seven-press solution. The regularized policy solves all eight displayed episodes; Value Flows solves three.

5.4 How the Policy Executes the Task

Task success alone does not distinguish direct completion from a long sequence of ineffective motions. Figure 3 shows eight evaluation runs on `puzzle-3x3-play`, where each press toggles a button and its neighbours and the displayed task admits a seven-press solution. The regularized policy solves all eight runs, whereas Value Flows solves three. The associated execution summaries are 9.8 ± 1.7 versus 14.6 ± 1.9 presses and median lengths of 278 versus 501 steps. Repeated presses on already-correct buttons are visible in the baseline trajectories.

Figure 4 extends the comparison to all five tasks of each named domain and includes every evaluation episode. Relative to Value Flows, the regularized pipeline uses 4% fewer button presses and 14% fewer spurious presses on `puzzle-3x3`; on `cube-double`, it grasps 8% less often and has 19% shorter median episodes. The corresponding median-length change on `scene-play` is 2%. Because completion rates and episode lengths both vary, these are descriptive execution costs, not matched-success estimates of efficiency.

Scope of the evidence. The experiments compare the full HJB-inspired learner with its baseline. They do not separate the contribution of the action-gradient term, the unit-cost geometry, or noise-dependent residual regularization from generic derivative regularization. Likewise, the slice visualization and spread statistics are not distributional calibration tests. The empirical claim is the usefulness of the complete proposed regularizer in these evaluations.

6 Conclusion

HJB-Constrained Value Flows combines distributional Bellman learning with a stationary differential prior through an initial-velocity value readout. Its single-noise surrogate regularizes local value derivatives without a separate critic or dynamics model. The resulting learner improves task scores in the reported offline and offline-to-online comparisons, with task-dependent gains and accompanying changes in value landscapes and execution. The results support the practical utility of this regularizer, while its component-level mechanisms remain open.

AI USE STATEMENT

Generative AI tools were used to improve the clarity and readability of the manuscript and to assist in searching for and identifying relevant literature. We did not use generative AI tools to develop the methodology or theoretical framework, formulate mathematical claims or proofs, propose hypotheses, design experiments, process experimental data, or interpret experimental results. All references identified with AI assistance were independently verified by the authors against the original sources. The authors reviewed and edited all AI-assisted text and take full responsibility for the final content of this work.

ETHICS STATEMENT

This work adheres to the ICLR Code of Ethics. The datasets and experimental resources used in this work are employed in accordance with their respective licenses and conditions of use. We do not collect new human-subject data. The authors take full responsibility for the accuracy and integrity of the methods, experiments, and conclusions presented in this work.

REPRODUCIBILITY STATEMENT

The main paper and appendices provide the information necessary to reproduce our method and experiments. The appendices include detailed derivations of the proposed objective, implementation details for the value-flow and HJB supervision terms, training and optimization settings, hyperparameters, benchmark configurations, and evaluation protocols. Additional implementation details, learning curves, and diagnostic analyses are provided to clarify the training procedure and evaluation conditions.

REFERENCES

Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. *arXiv preprint arXiv:2209.15571*, 2022.

Giacomo Baldan, Qiang Liu, Alberto Guardone, and Nils Thuerey. Physics vs distributions: Pareto optimal flow matching with physics constraints. In *International Conference on Learning Representations*, volume 2026, pp. 101215–101243, 2026.

Martino Bardi, Italo Capuzzo Dolcetta, et al. *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations*, volume 12. Springer, 1997.

Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International conference on machine learning*, pp. 449–458. Pmlr, 2017.

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

Michael G Crandall and Pierre-Louis Lions. Viscosity solutions of hamilton-jacobi equations. *Transactions of the American mathematical society*, 277(1):1–42, 1983.

Will Dabney, Georg Ostrovski, David Silver, and Rémi Munos. Implicit quantile networks for distributional reinforcement learning. In *International conference on machine learning*, pp. 1096–1105. PMLR, 2018.

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

Kenji Doya. Reinforcement learning in continuous time and space. *Neural computation*, 12(1): 219–245, 2000.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *International Conference on Learning Representations*, volume 2025, pp. 34668–34684, 2025.

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.

Vittorio Giammarino and Ahmed Hussain Qureshi. Goal reaching with eikonal-constrained hierarchical quasimetric reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 59145–59182, 2026.

Vittorio Giammarino, Ruiqi Ni, and Ahmed Qureshi. Physics-informed value learner for offline goal-conditioned reinforcement learning. *Advances in Neural Information Processing Systems*, 38: 137769–137796, 2026.

George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

Jeongho Kim, Jaeuk Shin, and Insoon Yang. Hamilton-jacobi deep q-learning for deterministic continuous-time systems with lipschitz continuous controls. *Journal of Machine Learning Research*, 22(206):1–34, 2021.

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

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

Rémi Munos. A convergent reinforcement learning algorithm in the continuous case based on a finite difference method. In *IJCAI (2)*, pp. 826–831, 1997.

Rémi Munos. A study of reinforcement learning in the continuous case by the means of viscosity solutions. *Machine Learning*, 40(3):265–299, 2000.

Ruiqi Ni and Ahmed H Qureshi. Ntfields: Neural time fields for physics-informed robot motion planning. *arXiv preprint arXiv:2210.00120*, 2022.

Ruiqi Ni, Ahmed Hussain Qureshi, et al. Physics-informed temporal difference metric learning for robot motion planning. In *International Conference on Learning Representations*, volume 2025, pp. 30632–30649, 2025.

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

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

Maziar Raissi, Paris Perdikaris, Nazanin Ahmadi, and George Em Karniadakis. Physics-informed neural networks and extensions. *arXiv preprint arXiv:2408.16806*, 2024.

John Seiffertt, Suman Sanyal, and Donald C Wunsch. Hamilton–jacobi–bellman equations and approximate dynamic programming on time scales. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 38(4):918–923, 2008.

Jan Tauberschmidt, Sophie Fellenz, Sebastian Vollmer, and Andrew Duncan. Physics-constrained fine-tuning of flow-matching models for generation and inverse problems. In *International Conference on Learning Representations*, volume 2026, pp. 143992–144037, 2026.

Utkarsh Utkarsh, Pengfei Cai, Alan Edelman, Rafael Gomez-Bombarelli, and Christopher Rackauckas. Physics-constrained flow matching: Sampling generative models with hard constraints. *Advances in Neural Information Processing Systems*, 38:160217–160252, 2026.

Hrishikesh Viswanath, Juanwu Lu, S Talha Bukhari, Damon Conover, Ziran Wang, and Aniket Bera. Physics informed viscous value representations. *arXiv e-prints*, pp. arXiv–2602, 2026.

Harley E Wiltzer, David Meger, and Marc G Bellemare. Distributional hamilton-jacobi-bellman equations for continuous-time reinforcement learning. In *International Conference on Machine Learning*, pp. 23832–23856. PMLR, 2022.

A Supporting Analysis

We state the conditions underlying the initial-velocity readout and distinguish the analytical mean-field prior from the sample-wise objective. These properties explain the construction; they do not establish convergence of the combined learner.

A.1 Initial-Velocity Readout and the Endpoint Convention

Fix (s, a) and let (z, ϵ) have finite second moments, with conditional base marginal $\epsilon \sim \mathcal{N}(0, 1)$. Define $z_\tau = (1 - \tau)\epsilon + \tau z$. Squared-error regression has the conditional-mean solution

$$v^\star(z_\tau \mid \tau, s, a) = \mathbb{E}[z - \epsilon \mid z_\tau, \tau, s, a]. \tag{18}$$

An integral objective with $\tau \sim \mathrm{U}[0, 1]$ determines this field only almost everywhere in flow time. We use the conditional-regression version at $\tau = 0$; the same conclusion holds for a continuous extension realizing this endpoint value. This is an endpoint condition on the regression field, not a consequence of matching the terminal law alone. Continuity of a finite network does not by itself establish that its endpoint matches this population regression value.

First-moment identity. At $\tau = 0$, $z_0 = \epsilon$, so the endpoint regression field is $v^\star(\epsilon \mid 0, s, a) = \mathbb{E}[z - \epsilon \mid \epsilon, s, a]$. Then

$$\begin{aligned}\mathbb{E}_\epsilon[v^\star(\epsilon \mid 0, s, a)] &= \mathbb{E}_\epsilon[\mathbb{E}[z - \epsilon \mid \epsilon, s, a]] \\ &= \mathbb{E}[z \mid s, a] - \mathbb{E}[\epsilon \mid s, a] = \mathbb{E}[z \mid s, a].\end{aligned}$$

This uses total expectation and the zero-mean base marginal, not target–noise independence. □

For an independent coupling, the stronger identity $v^\star(\epsilon \mid 0, s, a) = \mathbb{E}[z \mid s, a] - \epsilon$ also holds. BCFM couples the bootstrap target and base noise, so that stronger statement need not hold. The noise-averaged identity still applies at the corresponding CFM solution. At a fixed-policy distributional Bellman fixed point, the target mean is Q^π. During weighted, joint DCFM–BCFM–HJB optimization, Eq. (12) is an estimator and definition of a readout, not a guarantee that the network has attained that population solution.

A.2 Stationary Prior and Task Return

The auxiliary HJB derivation assumes the extended system in Eq. (6), a goal boundary, and finite minimum-time value on the states considered. For a small physical interval h, dynamic programming gives locally

$$T^\star(s, a) = h + \inf_{\|b\|_2 \le M} T^\star(s + hf(s, a), a + hb) + o(h).$$

A first-order expansion yields Eq. (7); using $Q^\star = -T^\star$ yields Eq. (8). At boundaries with additional restrictions on action rates, the infimum must use the feasible rate set. The regularizer adopts the interior ball expression as an approximation.

The unit cost, absence of continuous discounting, and action-rate model belong to this auxiliary problem. They do not alter the rewards or discrete discount in the Value Flows targets. Accordingly, task Q^π and auxiliary $Q^\star$ need not coincide. For a physical interval Δt, multiplying the auxiliary HJB equation by Δt gives a state-displacement term, an action-rate coefficient $M\Delta t$, and a running term $-\Delta t$. Equation (14) uses environment-step units ($\Delta t = 1$). Its scale therefore depends on the state/action coordinates and the return scale; it is not invariant to arbitrary rescaling. For stochastic transitions, $s' - s$ is a sampled displacement, not the deterministic drift itself. Hybrid state changes need not admit the same smooth dynamics. No additional goal-boundary penalty is imposed by the objective in Eq. (17).

A.3 What the Stochastic Residual Penalizes

Fix a transition (s, a, s') and average only over its Gaussian noise. Assume the required moments exist and state/action differentiation can be interchanged with this expectation. Let

$$g_s^\epsilon := \nabla_s q_\theta^\epsilon(s, a), \qquad g_a^\epsilon := \nabla_a q_\theta^\epsilon(s, a), \qquad J_\theta := \mathbb{E}_\epsilon\|g_a^\epsilon\|_2 - \|\mathbb{E}_\epsilon g_a^\epsilon\|_2 \ge 0.$$

The nonnegative J_θ is the Jensen gap of the action-gradient norm. For the residuals defined in Eqs. (14) and (16),

$$\mathbb{E}_\epsilon[\widetilde{\mathcal{R}}_\theta] = \mathcal{R}_\theta + MJ_\theta. \tag{19}$$

Applying the variance identity gives the exact unweighted decomposition

$$\mathbb{E}_\epsilon[\widetilde{\mathcal{R}}_\theta^2] = (\mathcal{R}_\theta + MJ_\theta)^2 + \mathrm{Var}_\epsilon(\widetilde{\mathcal{R}}_\theta). \tag{20}$$

Thus the sample-wise loss involves both a shifted mean residual and dispersion across noise-conditioned residuals. Even when $M = 0$, the variance term remains in general. It is therefore not solely a functional of the mean readout. Nor is it generally an upper bound on $\mathcal{R}_\theta^2$: the term $2MJ_\theta\mathcal{R}_\theta$ can be negative. This identity describes the objective; it does not establish which component drives the empirical gains.

The complete training loss also uses a noise-dependent confidence weight. Write $w = w_{\bar\theta}(s, a, \epsilon)$, $C_w = \mathbb{E}_\epsilon[w] > 0$, and define $\mathbb{E}_w[h] = \mathbb{E}_\epsilon[wh]/C_w$. Then

$$\mathbb{E}_\epsilon[w\widetilde{\mathcal{R}}_\theta^2] = C_w\left((\mathbb{E}_w[\widetilde{\mathcal{R}}_\theta])^2 + \mathrm{Var}_w(\widetilde{\mathcal{R}}_\theta)\right). \tag{21}$$

The weighted residual mean need not equal the residual of the unweighted $\hat{Q}_\theta$. Reweighting also does not commute with input differentiation: for $\bar q_w = \mathbb{E}_w[q_\theta^\epsilon]$ and $x \in \{s, a\}$, the product rule gives

$$\nabla_x \bar q_w = \mathbb{E}_w[\nabla_x q_\theta^\epsilon] + \frac{\mathbb{E}_\epsilon[(q_\theta^\epsilon - \bar q_w)\nabla_x w]}{C_w}.$$

Holding the weight fixed in the critic parameter update does not remove its dependence on the conditioning inputs in this identity. Equations (17) and (21) describe the implemented weighted surrogate. For the special exact independent-coupling endpoint regression solution, q_θ^ϵ is noise-independent and both the unweighted Jensen gap and residual variance vanish. This special case is not assumed for the coupled, learned critic.

B Training and Evaluation Conventions

B.1 Critic Update and Confidence Weight

The modification uses one base-noise draw per transition, the stationary -1 running term, $M = 10^{-3}$, and $\lambda_{\mathrm{H}} = 0.1$. Target values and confidence weights are held fixed when taking the critic gradient. The input derivatives in the residual remain differentiable with respect to θ, so its optimization requires mixed parameter–input derivatives. The target network is updated by Polyak averaging, as in Value Flows; its distributional and policy objectives are retained.

For completeness, let $D_{\bar\theta}(\epsilon; s, a) = \partial_\epsilon \phi_{\bar\theta}(\epsilon \mid 1, s, a)$. Using temperature $\beta_w > 0$ and logistic sigmoid σ, the inherited weight is

$$w_{\bar\theta}(s, a, \epsilon) = \sigma\left(-\frac{\beta_w}{|D_{\bar\theta}(\epsilon; s, a)|}\right) + \frac{1}{2}, \tag{22}$$

with limiting value $1/2$ at zero denominator. The derivative is computed along the target flow from

$$\frac{dD_\tau}{d\tau} = \partial_z v_{\bar\theta}(\phi_{\bar\theta}(\epsilon \mid \tau, s, a) \mid \tau, s, a)D_\tau, \qquad D_0 = 1.$$

The flow derivative supplies the spread proxy used by Value Flows (Dong et al., 2026); for a general nonlinear transport it is not an exact return-variance identity. Equation (22) gives $1/2 \le w_{\bar\theta} < 1$, increasing with $|D_{\bar\theta}|$. Thus the inherited weight emphasizes larger estimated spread; its name does not imply calibrated confidence in the predictions. The same target-flow computation is retained for weighting all three sample losses in Eq. (17).

B.2 Policy Extraction

The offline policy samples candidate actions from the behavior-cloning flow π_ω and executes the one with the highest value estimate. Offline-to-online fine-tuning retains the one-step actor π_η and the Value Flows policy objective (Dong et al., 2026). The architecture, target-flow integration, candidate-sampling procedure, and actor optimization follow that baseline; the only added training term is the critic regularizer.

B.3 Hyperparameter Convention

We retain $\lambda_{\mathrm{H}} = 0.1$ from preliminary development runs performed before the constant running-cost term was introduced. Figure 5 records those single-seed runs on one task from each of three domains at $M = 10^{-3}$. For that earlier residual, 0.1 gives the highest displayed peak score in each panel. These runs document the inherited parameter choice; they do not test the sensitivity of the final residual or isolate the role of M.

B.4 Supplementary Execution and Development Plots

Figure 4 displays the relative execution costs discussed in Section 5.4. Measurements include all evaluation episodes, so completion frequency and task duration both contribute to these summaries.

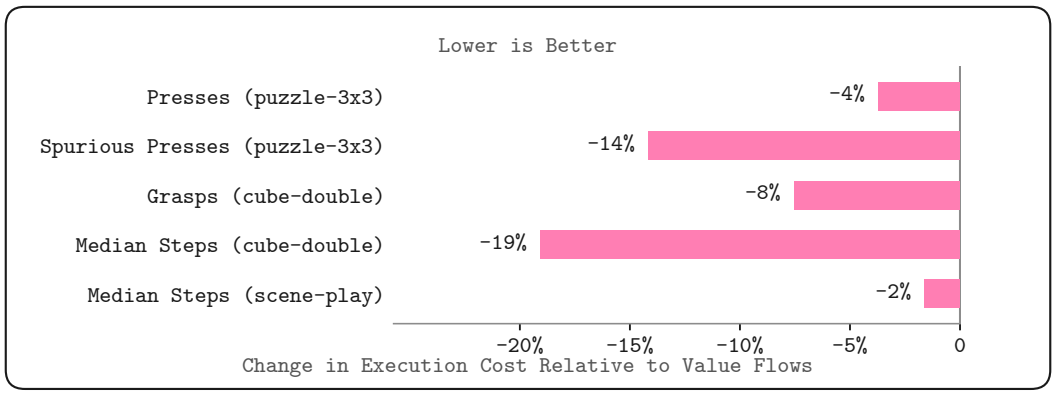


Figure 4: **Execution cost relative to Value Flows.** Changes in button presses, spurious presses, grasps, and median episode length, using all evaluation episodes and averaging over five tasks in each named domain. Motions are read from privileged object state. Relative changes place quantities with different units on a common scale.

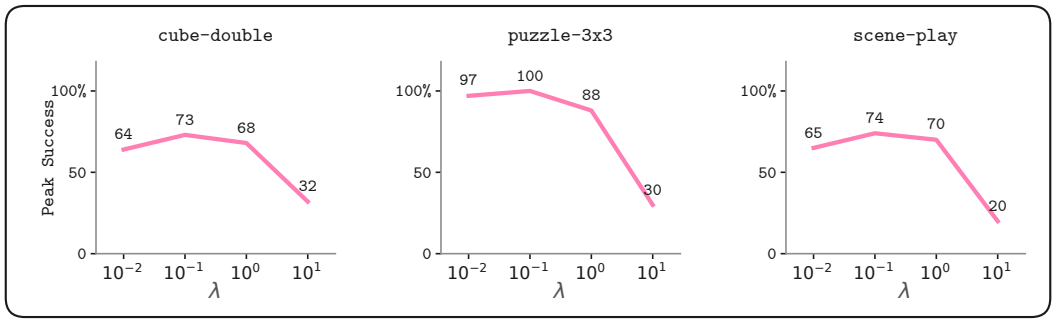


Figure 5: **Preliminary hyperparameter selection.** Peak evaluation scores for single-seed development runs on one task per domain, at $M = 10^{-3}$. These runs used an earlier residual without the constant running-cost term. The selected $\lambda_{\mathrm{H}} = 0.1$ was retained for the final residual; the plot is not a sensitivity study of that final objective. The figure label λ denotes λ_{H}.

C Full Offline Results

Tables 4 and 5 report the individual task results and all baselines. OGBench success rates and D4RL normalized returns retain their benchmark-specific units. The corresponding individual task results are reported in table 4.

Table 4: Offline RL, by task. OGBench entries are success percentages and D4RL entries are normalized returns. An asterisk identifies a designated tuning task. Colors use the best score across all methods in this table and its companion: blue is within 95% of the best, and pink marks other improvements over Value Flows. Distributional baselines and our method are shown here; additional policy baselines appear in table 5.

Environment	C51	IQN	CODAC	VF	**+ HJB**
scene-play					
task1	17±3	**100±0**	**99±0**	**99±0**	**100±0**
task2*	2±1	1±0	85±4	**97±1**	**100±0**
task3	0±1	**94±2**	90±3	**94±2**	**98±2**
task4	2±1	3±1	0±0	7±17	**70±12**
task5	0±0	0±0	0±0	0±0	0±0
puzzle-3x3-play					
task1	5±0	71±3	78±8	**99±0**	**100±0**
task2	0±0	2±2	5±2	**98±2**	**100±0**
task3	0±0	0±0	4±3	**97±1**	**100±0**
task4*	0±0	0±0	5±5	84±24	**100±0**
task5	0±0	0±0	6±5	58±39	**65±0**
cube-double-play					
task1	9±0	70±14	80±11	**97±1**	**99±1**
task2*	0±0	24±9	63±4	**76±7**	**76±2**
task3	0±0	25±6	66±9	73±4	**81±6**
task4	0±0	10±1	13±2	**30±5**	**31±3**
task5	0±0	**81±8**	**82±4**	69±5	72±8
cube-triple-play					
task1*	1±0	29±2	9±5	59±12	**85±1**
task2	0±0	0±0	**1±0**	0±0	0±0
task3	0±0	1±0	0±0	7±3	**15±1**
task4	0±0	0±0	0±0	0±0	**2±0**
task5	0±0	0±0	0±0	2±1	**4±3**
puzzle-4x4-play					
task1	0±0	41±2	37±32	36±3	39±3
task2	0±0	12±4	10±10	27±5	**31±4**
task3	0±0	45±7	33±29	30±4	42±7
task4*	0±0	23±2	12±10	**28±5**	25±2
task5	0±0	16±6	10±8	13±2	**21±2**
D4RL adroit					
pen-human	69±8	69±3	67±0	67±9	86±8
pen-cloned	67±9	80±11	76±2	73±5	82±6
pen-expert	110±3	118±19	136±2	117±3	126±2
door-human	0±0	0±0	3±1	7±2	**16±1**
door-cloned	0±0	0±0	0±0	0±0	1±0
door-expert	**104±1**	**105±0**	**104±0**	**104±1**	**106±0**
hammer-human	3±1	2±1	3±1	1±0	**4±2**
hammer-cloned	0±0	0±0	6±0	1±0	3±1
hammer-expert	122±1	121±7	126±1	125±5	127±2
relocate-human	0±0	0±0	0±0	0±0	**1±1**
relocate-cloned	0±0	0±0	0±0	0±0	0±0
relocate-expert	103±0	103±0	103±2	**105±3**	**109±1**

Table 5: Additional offline baselines from Dong et al. (2026), for the tasks in table 4. Metric units, task ordering, and color thresholds are the same as in that table.

Environment	BC	IQL	ReBRAC	FBRAC	IFQL	FQL
scene-play						
task1	19±6	94±3	**95±2**	**96±8**	**98±3**	**100±0**
task2*	1±1	12±3	50±13	46±10	0±0	76±9
task3	1±1	32±7	55±16	78±4	54±19	**98±1**
task4	2±2	0±1	3±3	4±4	0±0	5±1
task5	0±0	0±0	0±0	0±0	0±0	0±0
puzzle-3x3-play						
task1	5±2	33±6	**97±4**	63±19	94±3	90±4
task2	1±1	4±3	1±1	2±2	1±2	16±5
task3	1±1	3±2	3±1	1±1	0±0	10±3
task4*	1±1	2±1	2±1	2±2	0±0	16±5
task5	1±0	3±2	5±3	2±2	0±0	16±3
cube-double-play						
task1	8±3	27±5	45±6	47±11	35±9	61±9
task2*	0±0	1±1	7±3	22±12	9±5	36±6
task3	0±0	0±0	4±1	4±2	8±5	22±5
task4	0±0	0±0	1±1	0±1	1±1	5±2
task5	0±0	4±3	4±2	2±2	17±6	19±10
cube-triple-play						
task1*	1±1	4±4	1±2	0±0	2±2	20±6
task2	0±0	0±0	0±0	0±0	0±0	**1±2**
task3	0±0	0±0	0±0	0±0	0±0	0±0
task4	0±0	0±0	0±0	0±0	0±0	0±0
task5	0±0	1±1	0±0	0±0	0±0	0±0
puzzle-4x4-play						
task1	1±1	12±2	26±4	32±9	**49±9**	34±8
task2	0±0	7±4	12±4	5±3	4±4	16±5
task3	0±0	9±3	15±3	20±10	**50±14**	18±5
task4*	0±0	5±2	10±3	5±1	21±11	11±3
task5	0±0	4±1	7±3	2±2	2±2	7±3
D4RL adroit						
pen-human	71	78	**103**	77±7	71±12	53±6
pen-cloned	52	83	**103**	67±9	80±11	74±11
pen-expert	110	128	**152**	119±7	139±5	142±6
door-human	2	3	0	4±2	7±2	0±0
door-cloned	0	**3**	0	2±1	2±1	2±1
door-expert	**105**	**107**	**106**	**104±1**	**104±2**	**104±1**
hammer-human	3	2	0	2±1	3±1	1±1
hammer-cloned	1	2	5	2±1	2±1	**11±9**
hammer-expert	127	**129**	**134**	119±9	117±9	125±3
relocate-human	0	0	0	0±0	0±0	0±0
relocate-cloned	0	0	**2**	1±1	0±0	0±0
relocate-expert	**108**	**106**	**108**	**105±2**	**104±3**	**107±1**

D LEARNING CURVES

The following figures show evaluation performance over training for the regularized method, with a mean curve and one-standard-deviation band. The `scene-play`, `puzzle-3x3-play`, and `cube-double-play` evaluations use four seeds; `cube-triple-play`, `puzzle-4x4-play`, offline-to-online evaluations, and D4RL use three. Two `puzzle-3x3-play` task 5 runs without retained logs are excluded from the plotted averages. The tables report peak offline evaluations, whereas these curves show performance at each training step; no paired baseline learning curves are included.

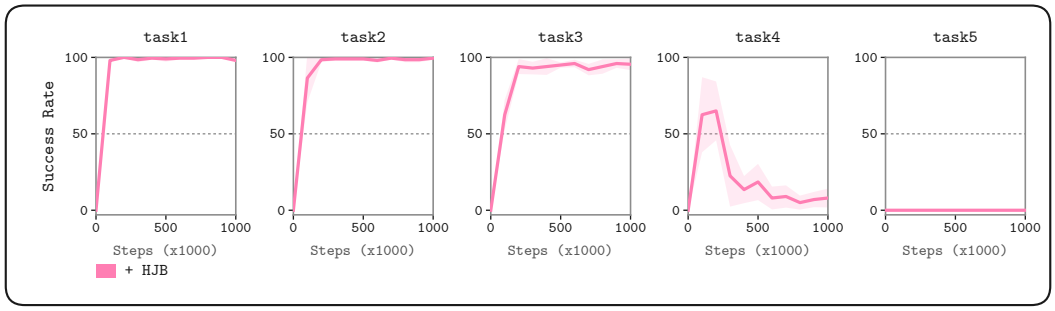


Figure 6: `scene-play`: success over training for the regularized critic. The curve shows the seed average at each step; the tables summarize peak evaluations.

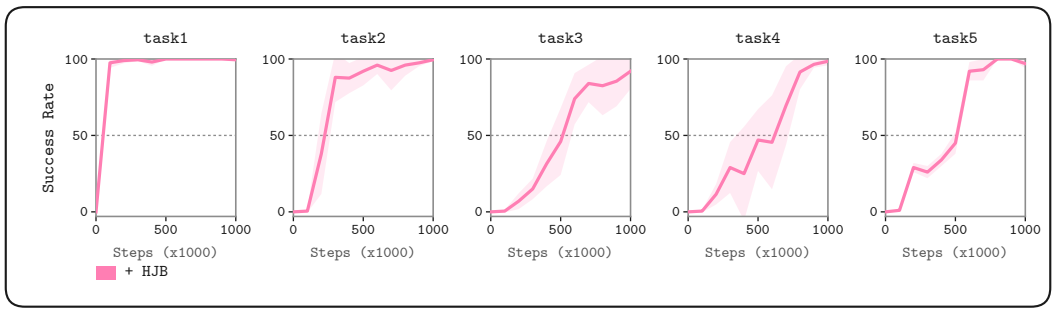


Figure 7: `puzzle-3x3-play`: success over training for the regularized critic, using the retained logs described above.

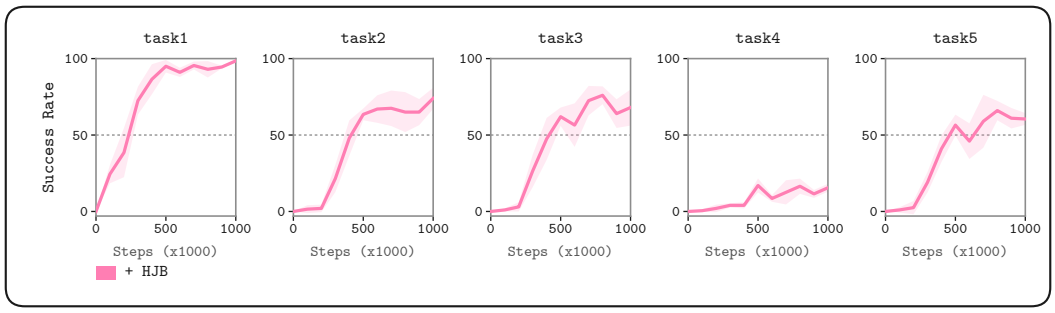


Figure 8: `cube-double-play`: evaluation success over training for the regularized critic.

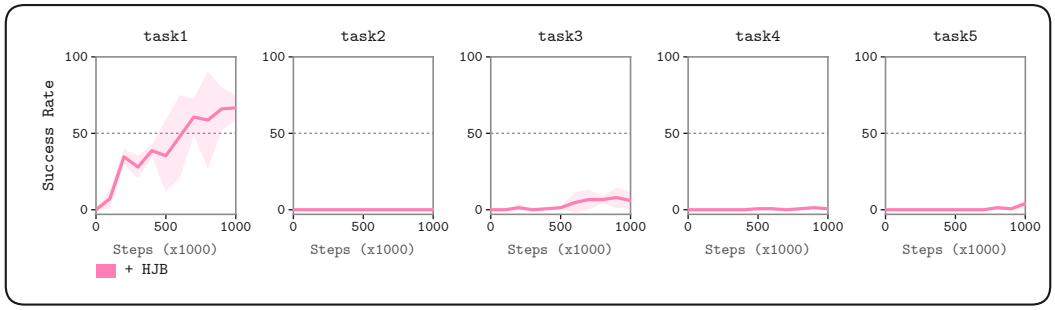


Figure 9: `cube-triple-play`: evaluation success over training for the regularized critic, with three seeds.

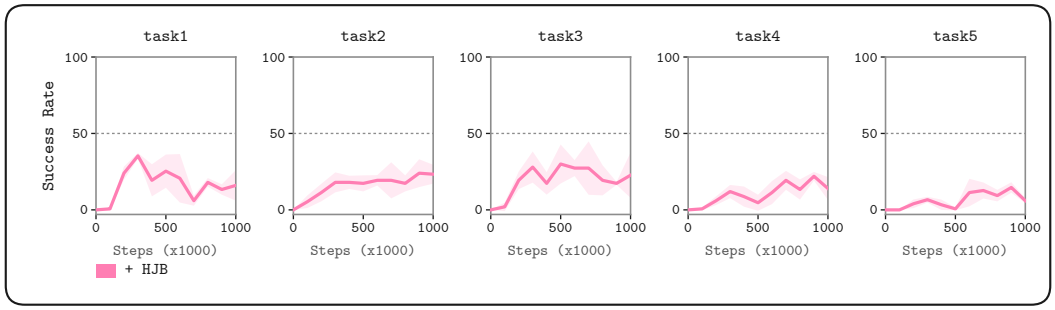


Figure 10: `puzzle-4x4-play`: evaluation success over training for the regularized critic, with three seeds.

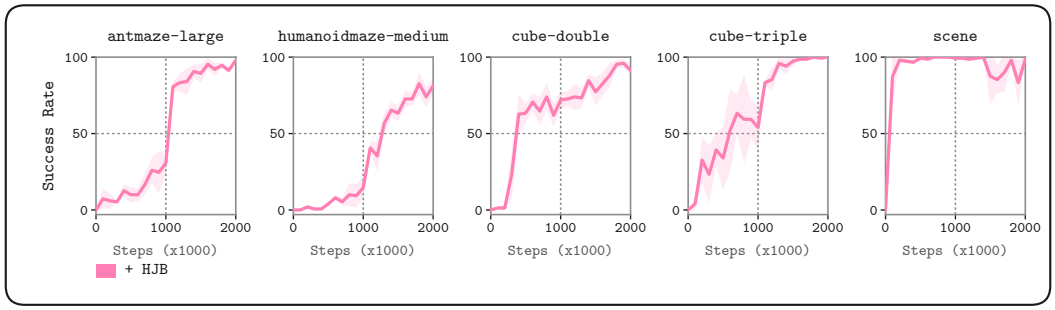


Figure 11: Offline-to-online fine-tuning on the five tasks in table 2, over three seeds. The vertical dotted line marks the transition to online interaction; the horizontal reference is 50% success.

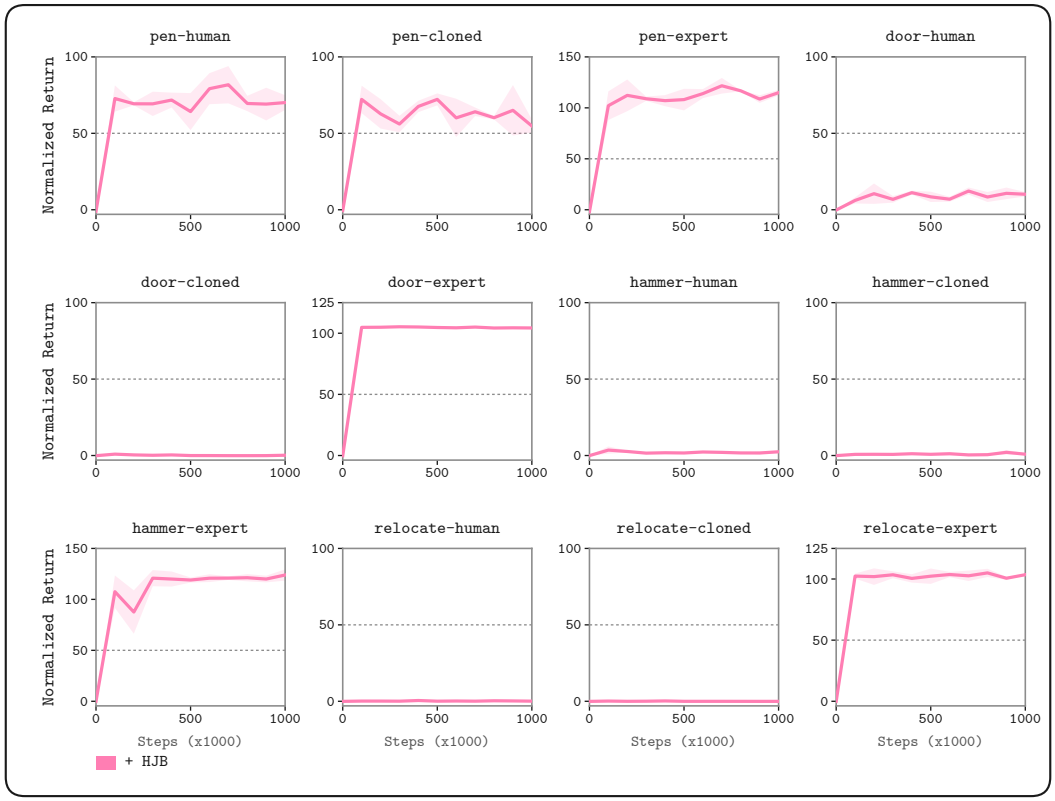


Figure 12: D4RL Adroit: normalized return over training for the regularized critic, over three seeds.