REINFORCEMENT LEARNING VIA FIXED-POINT FLOW MATCHING

Anonymous authors
Paper under double-blind review

ABSTRACT

In maximum entropy and behavior-constrained reinforcement learning (RL), the optimal policy is given by a Boltzmann distribution. Generating samples from this distribution can be challenging, as it is often complex and multimodal. While flow policies trained with flow matching provide a powerful framework for representing such distributions, they traditionally rely on access to training data, which is generally unavailable in online RL. To address this limitation, we propose fixed-point flow matching (FFM), a data-free procedure for training and fine-tuning flow policies in RL. FFM leverages the target flow identity to formulate the actor update as a fixed-point iteration, utilizing a flow-matching-style regression objective where the critic's action gradients serve as regression targets. We show that the optimal policy is a fixed point of this iteration. Through extensive experiments, we demonstrate that FFM effectively fine-tunes pretrained flow policies in the offline-to-online setting and enables efficient and accurate training of flow policies from scratch in online RL. These results establish fixed-point flow matching as a principled approach for learning expressive flow-based policies without requiring training data.

1 INTRODUCTION

Across a range of reinforcement learning (RL) formulations, the improved policy is available in closed form as a Boltzmann distribution of the current critic. In maximum entropy RL, $\pi^\star(x \mid s) \propto \exp(Q(s,x)/\lambda)$ is optimal for the policy improvement step of soft policy iteration (Ziebart et al., 2008; Haarnoja et al., 2017; 2018); in KL-regularized and behavior-constrained RL, which includes offline-to-online fine-tuning, the same expression appears tilted by a reference policy (Peters et al., 2010; Abdolmaleki et al., 2018; Peng et al., 2019; Nair et al., 2020). This target distribution can be complex and multimodal whenever several distinct action choices are viable, but the classical actor–critic recipe projects it onto a Gaussian due to the simplicity of the resulting optimization problem and the tractability of the policy's log-likelihood (Haarnoja et al., 2018). Flow and diffusion models (Lipman et al., 2023; Liu et al., 2023; Albergo & Vanden-Eijnden, 2023; Ho et al., 2020; Song et al., 2021) remove this representational restriction and have become the policy class of choice wherever a dataset is available (Chi et al., 2023; Reuss et al., 2023; Janner et al., 2022; Park et al., 2025b). Yet, the stable training objectives that are used to train these models are not directly usable for online RL due to a lack of samples from $\pi^\star$. Existing methods for training flow and diffusion models in the on-policy RL setting often require backpropagating the critic through the integrator (Wang et al., 2023; Ding & Jin, 2024; Wang et al., 2024; Zhang et al., 2025b; Celik et al., 2025; Le et al., 2026), whose memory cost grows with every integration step and quickly becomes prohibitive, or rely on importance-weighted regression (Ding et al., 2024; Ma et al., 2025; Dong et al., 2025), whose exponential critic weights have high variance and lead to unstable training updates. Moreover, methods for fine-tuning flow models typically use stochastic optimal control, which requires converting the flow into a diffusion model in order to use the adjoint method (Domingo-Enrich et al., 2025; Li & Levine, 2026; Dong et al., 2026b). The adjoint method stores each sampled trajectory and integrates an adjoint state backward along it, leading to complex and expensive update schemes.

In this work we ask whether a flow policy can be trained against a Boltzmann target with the same simple regression that makes flow matching stable, but without samples from the target. Our starting point is the *target flow identity*: for a Boltzmann target, the velocity field that generates the noised target is, up to a rescaling of the noisy action, a conditional average of the critic's action gradients

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.

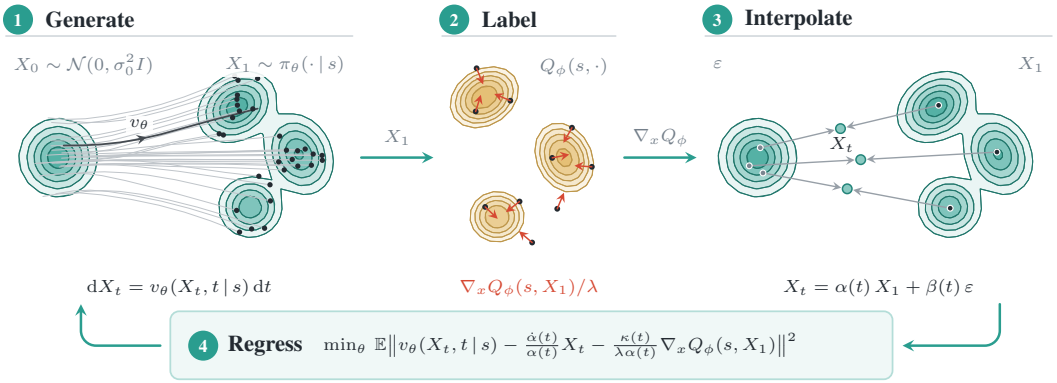


Figure 1: **The FFM policy-improvement step.** FFM repeats: **(1)** Clean actions X_1 are generated from the current policy by integrating Equation (5) (forward only, no gradients); the grey curves are sampling trajectories. **(2)** Critic action gradients are required for the FFM regression target of Equation (12). **(3)** Fresh noise ε and the clean action X_1 are used to sample a noised action X_t via Equation (3). **(4)** A flow-matching-style regression pulls v_θ toward the FFM target. The optimal policy $\pi^\star \propto e^{Q/\lambda}$ is a fixed point of this loop.

$\nabla_x Q(s, x)$ at clean actions. It replaces the data-derived regression target of flow matching by a critic-derived one, and the resulting objective, *target flow matching*, has the same minimizer as flow matching while reading its targets off the critic instead of a dataset. Since the objective still averages over samples of the target, we evaluate it with samples of the current policy and iterate. The resulting *fixed-point flow matching* (FFM) update, illustrated in Figure 1, generates actions with the current flow, labels them with critic gradients, noises them, and regresses the velocity field onto the labels. We show that the optimal policy is a fixed point of this iteration. The whole procedure is conceptually simple, easy to implement and requires neither backpropagation through the integrator nor importance weights or conversion into a diffusion model.

Contributions.

- **The target flow identity and target flow matching** (Section 4). We derive an identity that expresses the velocity field of a Boltzmann target as a conditional expectation of the gradient of its log-density, and turn it into target flow matching, an alternative training objective to flow matching that shares its population minimizer but regresses onto the gradient of the target rather than onto the displacement of the data.
- **Fixed-point flow matching** (Section 4). We propose a fixed-point iteration that evaluates the target flow matching objective with samples of the current policy instead of the optimal one, and prove that the optimal policy is a fixed point of this iteration (Theorem 3).
- **Extensions** (Section 5). We further introduce several extensions for flow policies: a trust-region loss in velocity space, a loss for behavior-constrained fine-tuning that reads the reference score off the pretrained flow, and correlated exploration through temporally correlated prior draws.
- **Two RL instantiations** (Section 5). We instantiate FFM for maximum entropy RL and for offline-to-online fine-tuning. We validate both across 50 offline-to-online tasks and 24 online tasks (Section 6), where FFM is competitive with or better than the strongest available baselines.

2 Related Work

Reinforcement learning with diffusion and flow policies. Diffusion and flow models entered RL through imitation learning and offline RL, where a dataset makes their standard training objectives applicable and the critic enters through weighting, guidance, or gradients through the sampling chain (Janner et al., 2022; Chi et al., 2023; Wang et al., 2023; Chen et al., 2023; Hansen-Estruch et al., 2023; Lu et al., 2023; Kang et al., 2023; Chen et al., 2024; Ding & Jin, 2024; Park et al., 2025b). In online RL there are no samples of the target policy, and existing methods differ in how the critic

drives the actor update. One line backpropagates the critic value through the sampling chain (Wang et al., 2023; Ding & Jin, 2024; Wang et al., 2024; Zhang et al., 2025b), which is memory-intensive and unstable for many integration steps. A second line uses critic-weighted regression, interpreting the denoising loss as a variational bound or as a mirror-descent step (Ding et al., 2024; Ma et al., 2025; Dong et al., 2025), and a third estimates policy gradients through the denoising process, typically under a trust region (Ren et al., 2025; McAllister et al., 2026; Zhang et al., 2025a; Le et al., 2026; Lv et al., 2026). Closest to FFM are methods that use the action gradient of the critic: DIPO (Yang et al., 2023) clones actions improved by gradient ascent on Q, and QSM (Psenka et al., 2024) regresses the score onto $\nabla_x Q$ evaluated at the noisy action, a plug-in approximation of the target score identity that is exact only at the clean end. Maximum entropy RL with generative policies additionally requires an entropy estimate: DACER (Wang et al., 2024) fits a Gaussian mixture, DIME (Celik et al., 2025) and MaxEntDP (Dong et al., 2025) use a lower bound and numerical integration, respectively, Sanokowski & Patil (2025) augment the MDP with the denoising steps, FLAG (Kim et al., 2026) uses importance sampling in the latent space of the flow, and AMDP (Thilges et al., 2026) amortizes the entropy-regularized update with adjoint matching. FFM obtains the entropy from the divergence of the velocity field it trains, using the instantaneous change of variables.

Fine-tuning and offline-to-online RL. Reward fine-tuning of pretrained diffusion and flow models has been cast as policy gradient over the denoising chain (Fan et al., 2023; Black et al., 2024; Liu et al., 2025b) and as entropy-regularized stochastic optimal control (Uehara et al., 2024; Domingo-Enrich et al., 2025), with the KL divergence to the pretrained model as regularizer; Uehara et al. (2025) review the corresponding inference-time methods. The same behavior-tilted target governs offline-to-online RL with flow policies. FQL (Park et al., 2025b) distills the flow into a one-step policy that is trained by reparameterized critic gradients, DSRL (Wagenmaker et al., 2025) keeps the flow fixed and learns a policy over its latent noise, ReinFlow and FINO (Zhang et al., 2025a; Shin et al., 2026b) inject noise into the flow to obtain tractable likelihoods and exploration, and Q-Flow (Doo et al., 2026) propagates terminal values to intermediate states of the flow. QAM (Li & Levine, 2026) imports adjoint matching: the pretrained flow is converted into a memoryless SDE and critic gradients are transported backward along sampled trajectories by the lean adjoint ODE, and TRQAM (Dong et al., 2026b) stabilizes it with a path-space trust region. Concurrent work regresses velocity corrections onto critic gradients (Wang et al., 2026; Shi et al., 2026; Ziakas et al., 2026) or steers a fixed flow policy at test time (Zhou et al., 2026; Ruan et al., 2026). FFM fine-tunes the pretrained flow directly in velocity space, requires neither SDE conversion nor adjoint simulation, and its behavior-tilted target has the KL-regularized optimum as a fixed point (Section E).

Beyond the RL literature, FFM shares the fixed-point structure of the matching objectives used to train diffusion and flow models from unnormalized densities, and of adjoint matching; Section C discusses these methods, together with further critic-driven policy updates.

3 Preliminaries

Reinforcement learning. We consider a Markov decision process with states $s \in \mathcal{S}$, continuous actions $x \in \mathcal{A} \subseteq \mathbb{R}^d$, transitions $P(s' \mid s, x)$, reward $r(s, x)$, and discount $\gamma \in [0, 1)$. The state-action value function of the current policy π is the expected discounted return $Q^\pi(s, x) = \mathbb{E}_\pi\big[\sum_{k\geq 0} \gamma^k r(s_k, x_k) \mid s_0 = s,\ x_0 = x\big]$ when taking action x in state s and following π thereafter. Maximizing this return alone drives the policy towards a deterministic optimum, which often suffers from premature convergence. The two settings we consider therefore both regularize the policy improvement step, which leads to closed-form solutions that are known only up to an intractable normalization constant.

In *behavior-constrained* RL (Peters et al., 2010; Abdolmaleki et al., 2018; Peng et al., 2019; Nair et al., 2020), which includes offline-to-online fine-tuning, the improvement step is penalized for deviating from a reference policy p_{ref}, typically a policy pretrained on the offline dataset. Maximizing $\mathbb{E}_{x\sim\pi}[Q(s, x)] - \lambda\, D_{\mathrm{KL}}(\pi(\cdot \mid s) \,\|\, p_{\mathrm{ref}}(\cdot \mid s))$ over all policies yields the *behavior-tilted* optimum

$$\pi^\star(x \mid s) \;=\; \frac{p_{\mathrm{ref}}(x \mid s)\exp\big(Q(s, x)/\lambda\big)}{\mathcal{Z}(s)}, \qquad \mathcal{Z}(s) = \int_{\mathcal{A}} p_{\mathrm{ref}}(x \mid s)\exp\big(Q(s, x)/\lambda\big)\,\mathrm{d}x, \quad (1)$$

where $\lambda > 0$ trades off the penalty for deviating from p_{ref} against maximizing the expected return.

In *maximum entropy* RL, the reference is dropped and the rewards are instead augmented by the entropy of the current policy $\mathcal{H}(\pi(\cdot \mid s)) = -\mathbb{E}_{x\sim\pi}[\log \pi(x \mid s)]$, which leads to the soft state-action value function $Q^\pi(s,x) = \mathbb{E}_\pi\big[\sum_{k\geq 0}\gamma^k\big(r(s_k,x_k) + \lambda\,\mathcal{H}(\pi(\cdot \mid s_k))\big) \mid s_0 = s,\, x_0 = x\big]$. The improvement step of soft policy iteration is the *Boltzmann policy* (Ziebart et al., 2008; Haarnoja et al., 2017; Levine, 2018; Haarnoja et al., 2018)

$$\pi^\star(x \mid s) \;=\; \frac{\exp\big(Q(s,x)/\lambda\big)}{\mathcal{Z}(s)}, \qquad \mathcal{Z}(s) = \int_{\mathcal{A}} \exp\big(Q(s,x)/\lambda\big)\,\mathrm{d}x, \tag{2}$$

with a temperature $\lambda > 0$, which now trades off maximizing the expected return against the policy's stochasticity.

In both cases, the normalization constant $\mathcal{Z}(s)$ is intractable, and generating samples from the potentially complex and multimodal target $\pi^\star$ requires fitting a tractable policy parameterization to its density. FFM optimizes a flow-based policy to this end.

Flow policies from flow matching. Assume for now that we have access to samples $X_1 \sim \pi^\star(\cdot \mid s)$ from the optimal policy. By drawing independent Gaussian noise $\varepsilon \sim \mathcal{N}(\cdot;\,0,\,I)$, one can construct a stochastic interpolant (Albergo et al., 2025) as

$$X_t \;=\; \alpha(t)\,X_1 \;+\; \beta(t)\,\varepsilon, \qquad t \in [0,1], \tag{3}$$

with smooth schedules satisfying $\alpha(0) = 0$, $\alpha(1) = 1$, $\beta(0) = \sigma_0$, $\beta(1) = 0$ (e.g. $\alpha(t) = t$, $\beta(t) = \sigma_0(1-t)$). Hence, $X_0 \sim \mathcal{N}\big(\cdot;\,0,\,\sigma_0^2 I\big)$ is pure noise, X_1 is a clean action from $\pi^\star(\cdot \mid s)$, and X_t in between is a partially noised action with marginal density ρ_t. The interpolant defines the *velocity field*

$$v(x,t \mid s) \;:=\; \mathbb{E}_{X_1\sim\pi^\star,\,\varepsilon}\big[\dot{X}_t \,\big|\, X_t = x\big], \qquad \dot{X}_t = \dot{\alpha}(t)\,X_1 + \dot{\beta}(t)\,\varepsilon, \tag{4}$$

which represents the mean direction in which interpolating actions move. Here and in the following, the subscript of the expectation lists the random variables of the interpolant it is taken over: $\mathbb{E}_{X_1\sim\pi^\star,\,\varepsilon}$ is the expectation over the pair (X_1,ε) of Equation (3) with clean actions $X_1 \sim \pi^\star(\cdot \mid s)$. Conditioning on $X_t = x$ hence averages over all pairs (X_1,ε) that are consistent with the noised action x at time t. Integrating the ordinary differential equation (ODE)

$$\mathrm{d}X_t \;=\; v(X_t,t \mid s)\,\mathrm{d}t, \quad X_0 \sim \mathcal{N}\big(\cdot;\,0,\,\sigma_0^2 I\big)\,, \tag{5}$$

transports the prior along the marginals ρ_t, so that the terminal samples X_1 are distributed according to $\pi^\star(\cdot \mid s)$ (Albergo et al., 2025). To learn the velocity, one can use *flow matching* (Lipman et al., 2023; Liu et al., 2023; Albergo & Vanden-Eijnden, 2023), which provides a simple and scalable regression objective, i.e.,

$$\mathcal{L}_{\mathrm{FM}}(\theta) \;=\; \mathbb{E}_{t,\,X_1\sim\pi^\star,\,\varepsilon}\big[\,\big\|v_\theta(X_t,t \mid s) - \dot{X}_t\big\|^2\,\big], \tag{6}$$

where the expectation is additionally taken over $t \sim \mathcal{U}(0,1)$. Optimizing Equation (6), however, requires samples from the optimal policy, which are not available in online RL. Instead, one typically has access to a critic Q, and in the next section we leverage it to obtain a similar regression objective that can be optimized without samples from $\pi^\star$.

4 Fixed-point target flow matching

Now assume we are given a critic function Q with $\pi^\star(\cdot \mid s) \propto \exp(Q(s,\cdot)/\lambda)$ instead of samples X_1. Furthermore, assume that the critic is differentiable, allowing us to compute the *Q-score*

$$\nabla_x \log \pi^\star(x \mid s) \;=\; \nabla_x Q(s,x)/\lambda, \tag{7}$$

which does not require knowledge of the intractable normalization constant in Equation (2). We arrive at the proposed method in two steps. First, we introduce the *target flow identity*, which expresses the velocity as a conditional expectation, similar to flow matching, but in terms of the Q-score. The corresponding objective, *target flow matching*, still requires samples $X_1 \sim \pi^\star$. This is addressed in the second step, where we introduce *fixed-point flow matching (FFM)*, which uses samples of the current policy instead of the optimal one, resulting in a fixed-point iteration with theoretical guarantees.

The target flow identity. Consider the interpolant in Equation (3) and define

$$\kappa(t) \;:=\; \tfrac{\dot\alpha(t)}{\alpha(t)}\beta^2(t) - \dot\beta(t)\beta(t) \;>\; 0 \quad \text{on } (0,1), \tag{8}$$

which recurs throughout the paper. The following proposition provides an alternative to flow matching (see Equation (6)) for representing the velocity field.

Proposition 1 (Target flow identity). *Let $\pi^\star(\cdot \mid s) \propto \exp(Q(s,\cdot)/\lambda)$ with $\mathcal{Z}(s) < \infty$, let $Q(s,\cdot)$ be continuously differentiable with $\mathbb{E}_{X_1\sim\pi^\star}\big[\|\nabla_x Q(s,X_1)\|\big] < \infty$, and let v be the velocity field Equation (4) of the interpolant Equation (3) with $X_1 \sim \pi^\star(\cdot \mid s)$. Then, for all $t \in (0,1]$,*

$$v(x,t \mid s) \;=\; \tfrac{\dot\alpha(t)}{\alpha(t)}\, x \;+\; \tfrac{\kappa(t)}{\lambda\,\alpha(t)}\, \mathbb{E}_{X_1\sim\pi^\star,\,\varepsilon}\big[\nabla_x Q(s,X_1) \,\big|\, X_t = x\big]. \tag{9}$$

Intuitively, the velocity field that generates the Boltzmann policy is, up to a rescaling of the current noisy action, a conditional average of Q-scores at clean actions. The proof of Proposition 1 is given in Section A and uses the fact that the velocity can be expressed in terms of the score for Gaussian source distributions in combination with the target score identity (De Bortoli et al., 2024). Note that the factor $1/\alpha(t)$ in Equation (9) diverges as $t \to 0$, where X_t is almost pure noise and the Q-score at the clean action X_1 carries little information about it. The conditional expectation itself remains finite, but a regression target built from Equation (9) has large variance near the noise end. In Section B.2 we show that mixing the target with the flow-matching target using a variance-optimal control variate removes this singularity.

Because the L^2-optimal predictor is the conditional expectation, Equation (9) can directly be turned into a regression objective as stated below.

Corollary 2 (Target flow matching). *Draw $t \sim \mathcal{U}(0,1)$, $X_1 \sim \pi^\star(\cdot \mid s)$, $\varepsilon \sim \mathcal{N}(\cdot;\,0,\,I)$ and form X_t by Equation (3), and let w be a time-dependent weighting with $w(t) > 0$ on $(0,1)$. The* target flow matching *objective*

$$\mathcal{L}_{\mathrm{TFM}}(\theta) \;=\; \mathbb{E}_{t,\,X_1\sim\pi^\star,\,\varepsilon}\Big[w(t)\,\big\|v_\theta(X_t,t \mid s) - \tfrac{\dot\alpha(t)}{\alpha(t)}\,X_t - \tfrac{\kappa(t)}{\lambda\,\alpha(t)}\,\nabla_x Q(s,X_1)\big\|^2\Big] \tag{10}$$

shares its population minimizer with flow matching, $\mathcal{L}_{\mathrm{FM}}$ of Equation (6).

In contrast to flow matching, whose target $\dot X_t$ encodes where the data moves, the target of target flow matching encodes where the critic points. However, the main obstacle remains: the expectation in Equation (10) still requires samples $X_1 \sim \pi^\star$.

Fixed-point flow matching. While we cannot sample from $\pi^\star$, we can sample $X_1 \sim \pi_\theta(\cdot \mid s)$, where π_θ denotes the terminal density at $t = 1$ obtained by integrating Equation (5) with the learned velocity field v_θ in place of v. In other words, we can sample from the current policy instead of the optimal one. FFM leverages this fact by building the interpolant in Equation (10) from the policy's own samples. To this end, consider the loss where the interpolant is built from clean actions of an arbitrary flow policy $\pi_{\bar\theta}$ with parameters $\bar\theta$, that is,

$$\mathcal{L}_{\mathrm{FFM}}(\theta,\bar\theta) = \mathbb{E}_{t,\,X_1\sim\pi_{\bar\theta},\,\varepsilon}\Big[w(t)\,\big\|v_\theta(X_t,t \mid s) - \tfrac{\dot\alpha(t)}{\alpha(t)}\,X_t - \tfrac{\kappa(t)}{\lambda\,\alpha(t)}\,\nabla_x Q(s,X_1)\big\|^2\Big], \tag{11}$$

where the expectation is again over the interpolant Equation (3), now with $X_1 \sim \pi_{\bar\theta}(\cdot \mid s)$ instead of $X_1 \sim \pi^\star(\cdot \mid s)$. Given some initial parameters θ_0, FFM computes the updates

$$\theta_{k+1} \;=\; \arg\min_\theta \;\mathcal{L}_{\mathrm{FFM}}(\theta,\theta_k), \tag{12}$$

which has the optimal policy as a fixed point, as stated below.

Theorem 3 (Fixed point). *Consider the iteration Equation (12) in its population form, that is, with the regression solved exactly over all velocity fields. Under regularity conditions on the critic Q that we state in Section A, if the current policy is optimal, $\pi_{\theta_k}(\cdot \mid s) = \pi^\star(\cdot \mid s)$, then the update returns the velocity field Equation (4) of $\pi^\star$, so that $\pi_{\theta_{k+1}}(\cdot \mid s) = \pi^\star(\cdot \mid s)$ and every subsequent update returns the same velocity field. The optimal policy is therefore a fixed point of FFM.*

The proof is given in Section A. In practice, FFM repeats the following steps. First, clean actions $X_1 \sim \pi_{\theta_k}(\cdot \mid s)$ are generated by integrating Equation (5) with the current velocity field (forward passes only, no gradients). Next, the Q-scores $\nabla_x Q(s, X_1)$ are computed, which can be stored in a replay buffer for sample reuse. Then, fresh noise ε and times t are drawn to form the noised actions X_t, on which the regression loss Equation (11) is minimized; see Figure 1 for an illustration.

5 Applications in Reinforcement Learning

Offline-to-online RL. In offline-to-online RL, one commonly pre-trains a flow policy v_{ref} by flow matching Equation (6) and a critic by an offline method on a fixed dataset. The policy is then fine-tuned online, where it must stay close to the pretrained flow while it improves, since offline critics are reliable only near the data. This motivates the behavior-constrained objective of Section 3. Prior work for fine-tuning flow policies typically converts the flow to a diffusion model with a specific (memoryless) noise schedule and leverages stochastic optimal control to compute adjoint states, which yield a stable regression objective for training (Domingo-Enrich et al., 2025; Li & Levine, 2026; Dong et al., 2026b). In contrast, FFM requires neither a conversion to a diffusion model nor the computation of adjoint states.

Keeping the policy close to the pretrained policy is commonly realized by considering the behavior-tilted target policy $\pi^\star \propto p_{\mathrm{ref}}\, e^{Q/\lambda}$, where p_{ref} is the terminal distribution of the pretrained flow policy (Uehara et al., 2024; Domingo-Enrich et al., 2025). Note that FFM only requires the score $\mathsf{s}_{\mathrm{ref}}(x) := \nabla_x \log p_{\mathrm{ref}}(x \mid s)$ of the reference policy, which can be read off the pretrained velocity field via the velocity–score bridge (see Section B.1),

$$\mathsf{s}_{\mathrm{ref}}(x) \;=\; \big(v_{\mathrm{ref}}(x, t \mid s) - \tfrac{\dot\alpha(t)}{\alpha(t)}\, x\big)/\kappa(t) \qquad (t \to 1). \tag{13}$$

Similar to Shin et al. (2026a), FFM can then directly be used for behavior-constrained fine-tuning by considering

$$\mathcal{L}_{\mathrm{BT-FFM}}(\theta, \bar\theta) = \mathbb{E}_{t,\, X_1 \sim \pi_{\bar\theta},\, \varepsilon}\Big[\big\| v_\theta(X_t, t \mid s) - \tfrac{\dot\alpha(t)}{\alpha(t)}\, X_t - \tfrac{\kappa(t)}{\alpha(t)}\, g(X_1)\big\|^2\Big], \tag{14}$$

with $g(x) := \mathsf{s}_{\mathrm{ref}}(x) + \frac{1}{\lambda}\nabla_x Q(s, x)$ the score of the behavior-tilted target, in place of $\mathcal{L}_{\mathrm{FFM}}$ in the fixed-point iteration Equation (12). By the argument of Theorem 3 (Section A), the behavior-tilted policy $\pi^\star \propto p_{\mathrm{ref}}\, e^{Q/\lambda}$ is a fixed point of the resulting iteration.

Maximum entropy reinforcement learning. The optimal policy in maximum entropy RL, as outlined in Section 3, is given by $\pi^\star(x \mid s) \propto \exp\big(Q(s, x)/\lambda\big)$. Hence, we can directly use FFM to realize the policy improvement step. However, computing the soft Q-function requires the entropy of the current flow policy, $\mathcal{H}(\pi_\theta(\cdot \mid s)) = -\mathbb{E}_{X_1 \sim \pi_\theta}[\log \pi_\theta(X_1 \mid s)]$. To that end, we use the *instantaneous change of variables* (Chen et al., 2018) to compute the log-density of a generated action along its own sampling trajectory as

$$\log \pi_\theta(X_1 \mid s) \;=\; \log \mathcal{N}\big(X_0;\, 0,\, \sigma_0^2 I\big) \;-\; \int_0^1 \nabla_x \cdot\, v_\theta(X_t, t \mid s)\, \mathrm{d}t, \tag{15}$$

and estimate the entropy by averaging over generated actions. However, the integral in Equation (15) requires the divergence of the velocity field in each integration step, which can be computationally expensive. Therefore, we follow common practice (Grathwohl et al., 2019) and rely on Hutchinson's trace estimator (Hutchinson, 1989; Skilling, 1989). More specifically, the estimator computes the divergence as the expectation $\nabla_x \cdot v = \mathbb{E}_z\big[z^\top (\partial_x v)\, z\big]$ over random probes z with $\mathbb{E}[z z^\top] = I$. The cost is one vector–Jacobian product per integration step, and no auxiliary density network is needed. For details, including an algorithmic description, we refer to Section D.

Trust-region constraints. Recently, on-policy maximum entropy RL has gained popularity, in particular when using massively parallel simulators. For on-policy RL it is well known that trust-region methods stabilize policy optimization and prevent premature convergence (Peters et al., 2010; Schulman et al., 2015; 2017; Abdolmaleki et al., 2015; Otto et al., 2021; Akrour et al., 2019; Voelcker et al., 2026; Le et al., 2026; Thilges et al., 2026). Similar to Thilges et al. (2026), we employ a trust-region loss that keeps the velocity close to the previous one via

$$\mathcal{L}_{\mathrm{TR}}(\theta, \bar\theta) \;=\; \int_0^1 \mathbb{E}_{X_1 \sim \pi_{\bar\theta},\, \varepsilon}\Big[\big\| v_\theta(X_t, t \mid s) - v_{\bar\theta}(X_t, t \mid s)\big\|^2\Big]\, \mathrm{d}t. \tag{16}$$

Equation (16) is then used to constrain the fixed-point update Equation (12) to stay close to the previous flow policy with parameters θ_k, i.e.,

$$\theta_{k+1} = \arg\min_{\theta} \; \mathcal{L}_{\mathrm{FFM}}(\theta, \theta_k) \quad \text{s.t.} \quad \mathcal{L}_{\mathrm{TR}}(\theta, \theta_k) \le \varepsilon_{\mathrm{TR}}. \tag{17}$$

The constrained problem in Equation (17) can be approached by the method of Lagrange multipliers (Boyd & Vandenberghe, 2004). Introducing the multiplier $\nu \ge 0$, the Lagrangian reads

$$\mathcal{L}_{\mathrm{Lag}}(\theta, \nu; \theta_k) \;=\; \mathcal{L}_{\mathrm{FFM}}(\theta, \theta_k) \;+\; \nu \left(\mathcal{L}_{\mathrm{TR}}(\theta, \theta_k) - \varepsilon_{\mathrm{TR}} \right), \tag{18}$$

and Equation (17) is replaced by the saddle-point problem

$$\theta_{k+1} \;=\; \arg\min_{\theta} \max_{\nu \ge 0} \; \mathcal{L}_{\mathrm{Lag}}(\theta, \nu; \theta_k). \tag{19}$$

We solve Equation (19) by alternating stochastic gradient descent on θ with gradient ascent on ν, such that the multiplier grows while the constraint is violated and decays once it is satisfied. We refer to Sections D and E for the instantiations with the previous policy and with the pretrained flow, respectively, including the dual update.

Correlated exploration. A flow policy is a deterministic function of its prior draw, and hence, integrating Equation (5) from $X_0 = \sigma_0 \varepsilon$ yields the action $x = \Psi_\theta(\varepsilon \mid s)$, whose only source of randomness is $\varepsilon \sim \mathcal{N}(\cdot; 0, I)$. Here, $\Psi_\theta(\cdot \mid s)$ is simply the sampler of the flow policy: the map that takes a prior draw ε and returns the clean action obtained by integrating the ODE with the learned velocity field. Hence, exploration can also be shaped by correlating the prior draws across environment steps. Similar to colored-noise exploration for Gaussian policies (Eberhard et al., 2023), we draw the priors within an episode from a stationary Ornstein–Uhlenbeck chain with correlation time τ_{ou}, i.e.,

$$\varepsilon_{k+1} \;=\; e^{-1/\tau_{\mathrm{ou}}} \, \varepsilon_k \;+\; \sqrt{1 - e^{-2/\tau_{\mathrm{ou}}}} \, \xi_k, \qquad \xi_k \sim \mathcal{N}(\cdot; 0, I), \qquad \varepsilon_0 \sim \mathcal{N}(\cdot; 0, I), \tag{20}$$

and execute $x_k = \Psi_\theta(\varepsilon_k \mid s_k)$. Since the chain in Equation (20) is stationary, every ε_k is marginally distributed according to $\mathcal{N}(\cdot; 0, I)$, and every executed action remains a sample of $\pi_\theta(\cdot \mid s_k)$ (see Section B.3). Therefore, the FFM regression Equation (11), the entropy estimate Equation (15), and the soft Bellman backup Equation (42) are unchanged, and so is the fixed point $\pi^\star$ of Theorem 3; only the states the agent visits change. Instead of resampling the prior at every step, the agent commits to a direction for roughly τ_{ou} steps, which produces the sustained, directed behavior that sparse-reward tasks require before any reward is observed. We use it only for the hard-exploration tasks of Section D.4. Other colored noise processes, such as the pink noise of Eberhard et al. (2023), can be used as well, which we leave to future work.

6 Experiments

We evaluate FFM in two settings: offline-to-online fine-tuning, and online on-policy maximum entropy RL, where we train a flow policy from scratch. For the offline-to-online setting, we focus on the ten OGBench (Park et al., 2025a) locomotion and manipulation domains totaling 50 tasks. For the online RL setting, we consider the GPU-accelerated MJX (Zakka et al., 2025) Playground and humanoid tasks, which result in 24 tasks in total. We restrict ourselves to presenting the main results in this section, and refer the reader to Section E and to Section D for additional results and ablations, such as varying the number of integration steps, the interpolation schedules, or different trust region constraints.

Offline-to-online RL. We pretrain a behavior-cloning flow policy and a critic offline (to 1M gradient steps) and then fine-tune online (to 1.5M), against TRQAM (Dong et al., 2026b), QAM (+QAM-E) (Li & Levine, 2026), FQL (+IFQL) (Park et al., 2025b), DSRL (Wagenmaker et al., 2025), and CGQL-L (Dong et al., 2026b) under an identical protocol (Figure 2). FFM's offline phase is deliberately conservative, keeping the policy inside the pretrained flow's support instead of chasing an offline critic that is unreliable off-support, so its offline average (0.63) trails the strongest offline baseline (TRQAM, 0.68; Table 5). However, once the online phase starts and the interaction makes the critic trustworthy, the budget is relaxed and FFM improves rapidly and attains the best aggregate success at 1.5M steps (0.95 mean vs. 0.93 for the next-best method; Table 6), with the widest margins on the hard, long-horizon tasks (e.g. HumanoidMaze-Large, 0.99 vs. 0.76) and without a single divergence across seeds.

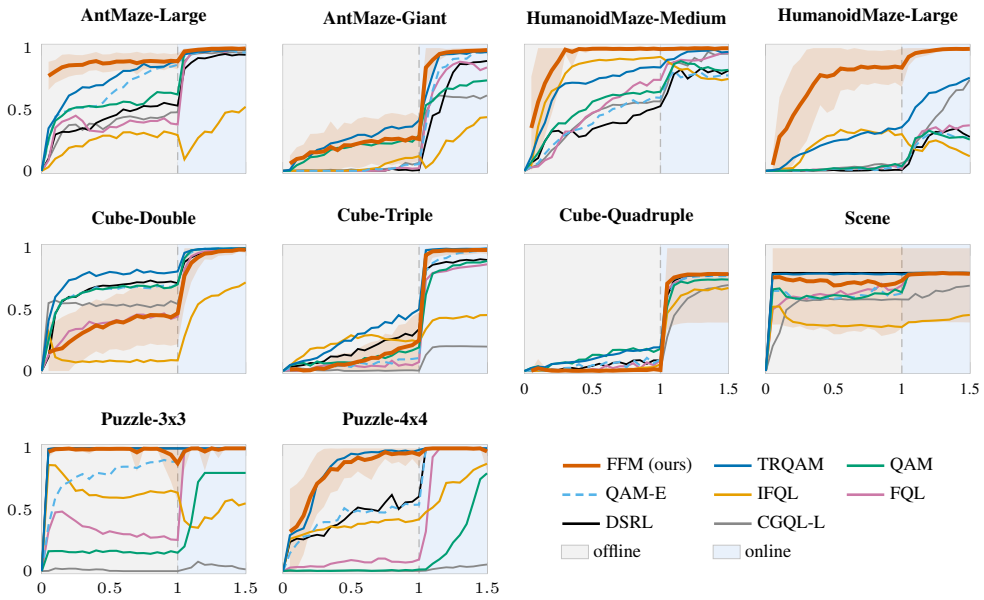


Figure 2: **Per-domain aggregated offline-to-online success rate** (mean over the five tasks and all seeds). Offline ($\leq$ 1M) and online ($>$ 1M) phases are shaded; FFM (ours) is drawn in bold with a ± 1 standard-deviation band. Per-task curves and full tables are in Section E.3.

Online RL. We additionally train FFM in the online on-policy RL setting where the policy is trained from scratch. Concretely, we train FFM on the 20 MuJoCo Playground DMC (Zakka et al., 2025) tasks and on the four MuJoCo Playground humanoid tasks that require precise high-dimensional control. We compare FFM's performance against state-of-the-art Gaussian policy representations (REPPO (Voelcker et al., 2026), PPO (Schulman et al., 2017), SPO (Xie et al., 2025)), diffusion policy representations (TRUDI (Le et al., 2026), AMDP (Thilges et al., 2026), DIME (Celik et al., 2025), DPPO (Ren et al., 2025)) and flow policy representations (FPO (McAllister et al., 2026)). Following Agarwal et al. (2021), every learning curve represents the aggregated interquartile mean (IQM) with 95% confidence intervals from a stratified bootstrap for every task in Figure 4 for the Playground tasks and in Figure 5 for the humanoid tasks, where every task is run on 10 seeds. While FFM performs comparably on the lower-dimensional Playground tasks (Figure 3 left two panels and Figure 4), FFM converges faster to the highest performance alongside TRUDI on the more sophisticated, high-dimensional humanoid tasks. This indicates that FFM's benefits are most visible in high-dimensional tasks. We further integrated correlated exploration into FFM and ran experiments on the Playground tasks. The resulting learning curves are shown in Figure 3 left two panels and Figure 4 for the aggregated and per-task results, respectively. Please note that we have run these experiments only on a subset of the tasks (see Figure 4). The aggregated FFM + OU curve (dashed in Figure 3) uses the FFM + OU learning curve on each task where we ran correlated exploration and the FFM learning curve on all other tasks. FFM + OU shows a faster convergence to a similar return value as the highest-performing baseline DIME, indicating that if computational resources permit, FFM's performance can be boosted even more by tuning τ_{ou}.

7 Conclusion

We introduced fixed-point flow matching, a data-free procedure for training flow policies using a flow-matching-style regression objective. Its basis is the target flow identity, which expresses the velocity field as a conditional average of Q-scores at clean actions and thereby replaces the data-derived targets of flow matching by critic gradients. Evaluating the resulting objective on samples from the current policy instead of the optimal policy yields a fixed-point iteration that has the optimal policy as a fixed point. The construction is conceptually simple, easy to implement, and needs no gradients through the sampling chain, no conversion of the flow into a stochastic differential equation, and no adjoint simulation. We consider two instantiations of the proposed framework: an

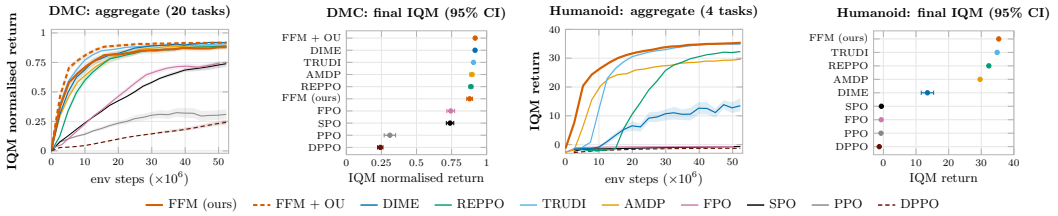


Figure 3: **Maximum Entropy Online RL.** We have run FFM on all 20 MJX Playground tasks and on the four MJX humanoid tasks and present the aggregated results over all Playground tasks (left two panels) and the aggregated results over the humanoid tasks (right two panels). We have evaluated all methods on 10 seeds for every single environment (see Figure 4 and Figure 5 for results) and report the IQM with 95% stratified bootstrapped confidence intervals following Agarwal et al. (2021). FFM performs on par with the baselines on the Playground tasks (left two panels) but converges to the highest return much faster than TRUDI on the high-dimensional humanoid tasks. This result indicates that FFM's benefits are most pronounced on high-dimensional tasks. Furthermore, we can boost FFM's performance by leveraging its deterministic mapping properties and incorporating correlated exploration (FFM + OU), which leads to a significantly faster convergence to a similar return value as the best-performing baseline DIME (left two panels) (see Figure 15).

online maximum entropy actor–critic algorithm, and offline-to-online fine-tuning against a behavior-tilted target. Across 50 offline-to-online tasks and 24 online tasks, both are competitive with or better than the strongest available baselines.

Limitations. FFM requires a *differentiable* target. The regression target is built from the Q-score $\nabla_x Q(s, x)$, so the critic must be differentiable in the action and the policy update is only as accurate as the critic's gradient. This restricts FFM to continuous action spaces. Extensions to non-differentiable targets and discrete action spaces are left for future work.

THE USE OF LARGE LANGUAGE MODELS (LLMS)

LLMs were used to draft parts of the paper from the authors' notes (introduction, related work, conclusion, several appendix sections), to help restructure and check the proofs, to implement parts of the experiment code and the figure and table scripts, to help with literature search, and to review the draft for errors before submission. The research question, method, and experimental design originate from the authors, who reviewed all LLM output and take full responsibility for the paper's content.

REFERENCES

Abbas Abdolmaleki, Rudolf Lioutikov, Jan R Peters, Nuno Lau, Luis Pualo Reis, and Gerhard Neumann. Model-based relative entropy stochastic search. *Advances in Neural Information Processing Systems*, 28, 2015.

Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=S1ANxQW0b.

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in neural information processing systems*, 34:29304–29320, 2021.

Tara Akhound-Sadegh, Jarrid Rector-Brooks, Joey Bose, Sarthak Mittal, Pablo Lemos, Cheng-Hao Liu, Marcin Sendera, Siamak Ravanbakhsh, Gauthier Gidel, Yoshua Bengio, Nikolay Malkin, and Alexander Tong. Iterated denoising energy matching for sampling from boltzmann densities. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp.

760–786. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/akhound-sadegh24a.html.

Riad Akrour, Joni Pajarinen, Jan Peters, and Gerhard Neumann. Projections for approximate policy iteration algorithms. In *International Conference on Machine Learning*, pp. 181–190. PMLR, 2019.

Michael Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 26(209):1–80, 2025.

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.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=YCWjhGrJFD.

Denis Blessing, Lorenz Richter, Julius Berner, Egor Malitskiy, and Gerhard Neumann. Bridge matching sampler: Scalable sampling via generalized fixed-point diffusion matching. *arXiv preprint arXiv:2603.00530*, 2026.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*, volume 1. Cambridge university press Cambridge, 2004.

Onur Celik, Zechu Li, Denis Blessing, Ge Li, Daniel Palenicek, Jan Peters, Georgia Chalvatzaki, and Gerhard Neumann. DIME: Diffusion-based maximum entropy reinforcement learning. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=Aw6dBR7Vxj.

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.

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

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Tianyi Chen, Haitong Ma, Na Li, Kai Wang, and Bo Dai. One-step flow policy mirror descent. *arXiv preprint arXiv:2507.23675*, 2025.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin CM Burchfiel, and Shuran Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.026.

Valentin De Bortoli, Michael Hutchinson, Peter Wirnsberger, and Arnaud Doucet. Target score matching. *arXiv preprint arXiv:2402.08667*, 2024.

Niclas Dern, Lennart Redl, Sebastian Pfister, Marcel Kollovieh, David Lüdke, and Stephan Günnemann. Energy-weighted flow matching: Unlocking continuous normalizing flows for efficient and scalable Boltzmann sampling. *arXiv preprint arXiv:2509.03726*, 2025.

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 *Advances in Neural Information Processing Systems*, 2024.

Shutong Ding, Ke Hu, Shan Zhong, Haoyang Luo, Weinan Zhang, Jingya Wang, Jun Wang, and Ye Shi. Genpo: Generative diffusion models meet on-policy reinforcement learning. *Advances in neural information processing systems*, 38:130443–130474, 2025.

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.

Carles Domingo-Enrich and Jiequn Han. Adjoint matching through the lens of the stochastic maximum principle in optimal control. *arXiv preprint arXiv:2604.08580*, 2026.

Carles Domingo-Enrich, Michal Drozdzal, Brian Karrer, and Ricky T. Q. Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=xQBRrtQM8u.

Carles Domingo-Enrich, Yuanqi Du, and Michael S. Albergo. A unified perspective on fine-tuning and sampling with diffusion and flow models. *arXiv preprint arXiv:2605.00229*, 2026.

Xiaoyi Dong, Jian Cheng, and Xi Sheryl Zhang. Maximum entropy reinforcement learning with diffusion policy. In *International Conference on Machine Learning*, 2025.

Xiaoyi Dong, Xi Sheryl Zhang, and Jian Cheng. Mean flow policy optimization. In *Forty-third International Conference on Machine Learning*, 2026a. URL https://openreview.net/forum?id=88H3V90Wzo.

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

JaeHyeok Doo, Byeongguk Jeon, Seonghyeon Ye, Kimin Lee, and Minjoon Seo. Q-Flow: Stable and expressive reinforcement learning with flow-based policy. *arXiv preprint arXiv:2605.13435*, 2026.

Onno Eberhard, Jakob Hollenstein, Cristina Pinneri, and Georg Martius. Pink noise is all you need: Colored noise exploration in deep reinforcement learning. In *International Conference on Learning Representations*, 2023.

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

Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. DPOK: Reinforcement learning for fine-tuning text-to-image diffusion models. In *Advances in Neural Information Processing Systems*, 2023.

Jesse Farebrother, Jordi Orbay, Quan Vuong, Adrien Ali Taiga, Yevgen Chebotar, Ted Xiao, Alex Irpan, Sergey Levine, Pablo Samuel Castro, Aleksandra Faust, Aviral Kumar, and Rishabh Agarwal. Stop regressing: Training value functions via classification for scalable deep RL. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 13049–13071. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/farebrother24a.html.

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

Ting Gao, Stavros Orfanoudakis, Nan Lin, Winnie Daamen, Serge Hoogendoorn, and Elvin Isufi. Flow matching policy optimization with mirror descent and entropy constraints. *arXiv preprint arXiv:2603.17685*, 2026.

Abdelghani Ghanem and Mounir Ghogho. Entropy-regularized adjoint matching for offline reinforcement learning. *arXiv preprint arXiv:2605.06156*, 2026.

Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. FFJORD: Free-form continuous dynamics for scalable reversible generative models. In *International Conference on Learning Representations*, 2019.

Zhengyi Guo, Jiayuan Sheng, David D. Yao, and Wenpin Tang. Improved techniques for fine-tuning flow models via adjoint matching: A deterministic control pipeline. *arXiv preprint arXiv:2605.06583*, 2026.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *International conference on machine learning*, pp. 1352–1361. PMLR, 2017.

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

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

Aaron Havens, Benjamin Kurt Miller, Bing Yan, Carles Domingo-Enrich, Anuroop Sriram, Brandon Wood, Daniel Levine, Bin Hu, Brandon Amos, Brian Karrer, Xiang Fu, Guan-Horng Liu, and Ricky T. Q. Chen. Adjoint sampling: Highly scalable diffusion samplers via adjoint matching. In *International Conference on Machine Learning*, 2025.

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

Michael F Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076, 1989.

Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, 2022.

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

Sungha Kim, Gawon Lee, Jusuk Lee, Jonghae Park, H. Jin Kim, and Daesol Cho. FLAG: Flow policy MaxEnt-RL by latent augmented guidance. *arXiv preprint arXiv:2605.30749*, 2026.

Huy Le, Onur Celik, Denis Blessing, Tai Hoang, Claas A Voelcker, Axel Brunnbauer, Felix Richter, Michael Volpp, and Gerhard Neumann. Trust-region diffusion policies for massively parallel on-policy RL. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=mGu2fs7kJt.

Sergey Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

Qiyang Li and Sergey Levine. Q-learning with adjoint matching. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=vd4eNAdtO6.

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

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.

Guan-Horng Liu, Jaemoo Choi, Yongxin Chen, Benjamin Kurt Miller, and Ricky T. Q. Chen. Adjoint Schrödinger bridge sampler. In *Advances in Neural Information Processing Systems*, 2025a.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-GRPO: Training flow matching models via online RL. *arXiv preprint arXiv:2505.05470*, 2025b.

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 *International Conference on Machine Learning*, pp. 22825–22855. PMLR, 2023.

Lei Lv, Yunfei Li, Yu Luo, Fuchun Sun, Tao Kong, Jiafeng Xu, and Xiao Ma. Flow-based policy for online reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=CANUXhPoyn.

Lei Lv, Yunfei Li, Yu Luo, Fuchun Sun, and Xiao Ma. FLAC: Maximum entropy RL via kinetic energy regularized bridge matching. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=v0qBWYewKR.

Haitong Ma, Tianyi Chen, Kai Wang, Na Li, and Bo Dai. Efficient online reinforcement learning for diffusion policy. In *International Conference on Machine Learning*, 2025.

David McAllister, Songwei Ge, Brent Yi, Chung Min Kim, Ethan Weber, Hongsuk Choi, Haiwen Feng, and Angjoo Kanazawa. Flow matching policy gradients. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=eoEmoKoQpJ.

Mattia Mosso, Jaemoo Choi, and Heng Yang. Hard-constrained sampling on embedded Riemannian manifolds via adjoint Schrödinger bridges. *arXiv preprint arXiv:2608.25838*, 2026.

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

Fabian Otto, Philipp Becker, Vien Anh Ngo, Hanna Carolin Maria Ziesche, and Gerhard Neumann. Differentiable trust region layers for deep reinforcement learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=qYZD-AO1Vn.

RuiKang OuYang, Bo Qiang, and José Miguel Hernández-Lobato. BNEM: A Boltzmann sampler based on bootstrapped noised energy matching. *Transactions on Machine Learning Research*, 2026.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In *International Conference on Learning Representations (ICLR)*, 2025a.

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. *arXiv preprint arXiv:1910.00177*, 2019.

Jan Peters, Katharina Mulling, and Yasemin Altun. Relative entropy policy search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 24, pp. 1607–1612, 2010.

Peter Potaptchik, Cheuk-Kit Lee, and Michael S. Albergo. Tilt matching for scalable sampling and fine-tuning. *arXiv preprint arXiv:2512.21829*, 2025.

Michael Psenka, Alejandro Escontrela, Pieter Abbeel, and Yi Ma. Learning a diffusion model policy from rewards via q-score matching. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=35ahHydjXo.

Antonin Raffin, Jens Kober, and Freek Stulp. Smooth exploration for robotic reinforcement learning. In *Conference on Robot Learning*, pp. 1634–1644. PMLR, 2022.

Allen Z. Ren, Justin Lidard, Lars Lien Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=mEpqHvbD2h.

Moritz Reuss, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov. Goal-conditioned imitation learning using score-based diffusion policies. *arXiv preprint arXiv:2304.02532*, 2023.

Herbert Robbins. An empirical Bayes approach to statistics. In *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pp. 157–163, 1956.

Yifan Ruan, Chenyang Cao, Andreas Burger, Ali Pesaranghader, Kaveh Kamali, Jaehong Kim, Nandita Vijaykumar, Alán Aspuru-Guzik, Igor Gilitschenski, and Nicholas Rhinehart. QPILOTS: Efficient test-time Q-steering for flow policies. *arXiv preprint arXiv:2606.14801*, 2026.

Sebastian Sanokowski and Kaustubh Patil. Diffusion-augmented Markov decision processes for maximum entropy reinforcement learning. *arXiv preprint arXiv:2512.02019*, 2025.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. Pmlr, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Kexin Shi, Junyao Shi, Poorvi Hebbar, Zhuolun Zhao, Tarun Amarnath, Yifan Su, Shikhar Bahl, and Deepak Pathak. FlowDPG: Deterministic policy gradient on flow matching policies for real-world manipulation. *arXiv preprint arXiv:2606.22303*, 2026.

Jeongwoo Shin, Dongsoo Shin, Yuchen Zhu, Wei Guo, Yongxin Chen, Joonseok Lee, Jaewoong Choi, and Jaemoo Choi. Efficient adjoint matching for fine-tuning diffusion models. *arXiv preprint arXiv:2605.11480*, 2026a.

Yongjae Shin, Jongseong Chae, Jongeui Park, and Youngchul Sung. Flow matching with injected noise for offline-to-online reinforcement learning. In *International Conference on Learning Representations*, 2026b.

John Skilling. The eigenvalues of mega-dimensional matrices. In *Maximum Entropy and Bayesian Methods: Cambridge, England, 1988*, pp. 455–466. Springer, 1989.

Oswin So, Brian Karrer, Chuchu Fan, Ricky T. Q. Chen, and Guan-Horng Liu. Discrete adjoint matching. In *International Conference on Learning Representations*, 2026.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

Serge Thilges, Onur Celik, Denis Blessing, Emiliyan Gospodinov, and Gerhard Neumann. Scalable maximum entropy reinforcement learning for diffusion policies via adjoint matching. *arXiv preprint arXiv:2606.22630*, 2026.

Masatoshi Uehara, Yulai Zhao, Kevin Black, Ehsan Hajiramezanali, Gabriele Scalia, Nathaniel Lee Diamant, Alex M. Tseng, Tommaso Biancalani, and Sergey Levine. Fine-tuning of continuous-time diffusion models as entropy-regularized control. *arXiv preprint arXiv:2402.15194*, 2024.

Masatoshi Uehara, Yulai Zhao, Chenyu Wang, Xiner Li, Aviv Regev, Sergey Levine, and Tommaso Biancalani. Inference-time alignment in diffusion models with reward-guided generation: Tutorial and review. *arXiv preprint arXiv:2501.09685*, 2025.

Francisco Vargas, Will Sussman Grathwohl, and Arnaud Doucet. Denoising diffusion samplers. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=8pvnfTAbu1f.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

Claas A Voelcker, Axel Brunnbauer, Marcel Hussing, Michal Nauman, Pieter Abbeel, Radu Grosu, Eric Eaton, Amir massoud Farahmand, and Igor Gilitschenski. Relative entropy pathwise policy optimization. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=4vmm8mlHkS.

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

Yinuo Wang, Likun Wang, Yuxuan Jiang, Wenjun Zou, Tong Liu, Xujie Song, Wenxuan Wang, Liming Xiao, Jiang Wu, Jingliang Duan, and Shengbo Eben Li. Diffusion actor-critic with entropy regulator. In *Advances in Neural Information Processing Systems*, 2024.

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.

Ziqian Wang, Rui Zhang, Yitian Liu, Xingjian Mao, Minqian Wang, and Yao Mu. Q-VGM: Q-guided value-gradient matching for offline-to-online RL of flow-matching VLA policies. *arXiv preprint arXiv:2606.08015*, 2026.

Dongyeop Woo and Sungsoo Ahn. Iterated energy-based flow matching for sampling from Boltzmann densities. *arXiv preprint arXiv:2408.16249*, 2024.

Zhengpeng Xie, Qiang Zhang, Fan Yang, Marco Hutter, and Renjing Xu. Simple policy optimization. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=SG8Yx1FyeU.

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

Kevin Zakka, Baruch Tabanpour, Qiayuan Liao, Mustafa Haiderbhai, Samuel Holt, Jing Yuan Luo, Arthur Allshire, Erik Frey, Koushil Sreenath, Lueder A. Kahrs, Carlo Sferrazza, Yuval Tassa, and Pieter Abbeel. Mujoco playground: An open-source framework for gpu-accelerated robot learning and sim-to-real transfer., 2025. URL https://github.com/google-deepmind/mujoco_playground.

Qinsheng Zhang and Yongxin Chen. Path integral sampler: A stochastic control approach for sampling. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=_uCb2ynRu7Y.

Tonghe Zhang, Chao Yu, Sichang Su, and Yu Wang. ReinFlow: Fine-tuning flow matching policy with online reinforcement learning. In *Advances in Neural Information Processing Systems*, 2025a.

Yixian Zhang, Shu'ang Yu, Tonghe Zhang, Mo Guang, Haojia Hui, Kaiwen Long, Yu Wang, Chao Yu, and Wenbo Ding. SAC Flow: Sample-efficient reinforcement learning of flow-based policies via velocity-reparameterized sequential modeling. *arXiv preprint arXiv:2509.25756*, 2025b.

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

Christos Ziakas, Alessandra Russo, and Avishek Joey Bose. Aligning flow map policies with optimal Q-guidance. *arXiv preprint arXiv:2605.12416*, 2026.

Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, AAAI'08, pp. 1433–1438. AAAI Press, 2008. ISBN 9781577353683.

Appendix contents

A Proofs **17**

B Further method details **18**

B.1 One-sided stochastic interpolants 18

B.2 Generalized regression targets and control variates 19

B.3 Correlated exploration 20

C Extended related work **21**

D Maximum-entropy reinforcement learning **22**

D.1 Actor-Critic based Maximum-entropy reinforcement learning 22

D.2 Pseudocode 25

D.3 Experimental details 25

D.4 Further experiments and design ablations 26

E Offline-to-online fine-tuning **32**

E.1 Further details 32

E.2 Pseudocode 33

E.3 Experimental details 33

E.4 Full results 36

E.5 Design ablations 36

A PROOFS

This appendix proves the three results of Section 4: the target flow identity (Proposition 1), the target flow matching objective (Corollary 2), and the fixed-point property of FFM (Theorem 3). We use the notation of Section B.1 and suppress the state s.

Proof of the target flow identity (Proposition 1) The proof combines the velocity–score bridge Equation (28) with the *target score identity* of De Bortoli et al. (2024): for a continuously differentiable density μ with $\mathbb{E}_\mu\big[\|\nabla \log \mu(X_1)\|\big] < \infty$ and all $t \in (0,1]$,

$$\mathsf{s}^\mu(x,t) \;=\; \frac{1}{\alpha(t)}\,\mathbb{E}_\mu\big[\nabla \log \mu(X_1)\,\big|\,X_t = x\big], \tag{21}$$

where s^μ denotes the score Equation (26) of the interpolant of μ. Whereas the noise identity Equation (27) averages the noise, Equation (21) averages the score of the *target* at clean samples; it follows from an integration by parts that moves the gradient from the Gaussian kernel of ρ_t onto μ, and the factor $1/\alpha(t)$ accounts for the scaling of X_1 in Equation (25).

Proof of Proposition 1. Apply Equation (21) to $\mu = \pi^\star \propto e^{Q(s,\cdot)/\lambda}$. By Equation (7), $\nabla_x \log \pi^\star = \nabla_x Q(s,\cdot)/\lambda$, and the integrability assumption of Proposition 1 is exactly that of the identity, so the score of the noised Boltzmann policy is

$$\mathsf{s}^{\pi^\star}(x,t) \;=\; \frac{1}{\lambda\,\alpha(t)}\,\mathbb{E}_{\pi^\star}\big[\nabla_x Q(s,X_1)\,\big|\,X_t = x\big]. \tag{22}$$

Inserting Equation (22) into the bridge Equation (28) gives Equation (9). □

Proof of target flow matching (Corollary 2)

Proof of Corollary 2. Both objectives are weighted least-squares regressions of $v_\theta(X_t,t)$ onto a random target under the interpolant of $\pi^\star$ and $t \sim \mathcal{U}(0,1)$: flow matching Equation (6) onto $\dot{X}_t$, and target flow matching Equation (10) onto

$$Y \;:=\; \tfrac{\dot\alpha(t)}{\alpha(t)}\,X_t + \tfrac{\kappa(t)}{\lambda\,\alpha(t)}\,\nabla_x Q(s,X_1). \tag{23}$$

When the minimization ranges over all velocity fields, the minimizer of a least-squares regression is the conditional expectation of the target given the regressor, here (X_t,t), for almost every (x,t) with respect to the interpolant marginals; since the weighting $w(t) > 0$ depends on t alone, it does not change this pointwise minimizer. For flow matching the conditional expectation is $\mathbb{E}_{\pi^\star}[\dot{X}_t \mid X_t = x] = v(x,t)$ by the definition Equation (4) of the velocity. For target flow matching, the first term of Y is a function of X_t and therefore

$$\mathbb{E}_{\pi^\star}[Y \mid X_t = x] \;=\; \tfrac{\dot\alpha(t)}{\alpha(t)}\,x + \tfrac{\kappa(t)}{\lambda\,\alpha(t)}\,\mathbb{E}_{\pi^\star}\big[\nabla_x Q(s,X_1)\,\big|\,X_t = x\big] \;=\; v(x,t) \tag{24}$$

by the target flow identity Equation (9). Both objectives hence share the population minimizer v, for every positive weighting w. □

Proof of the fixed point (Theorem 3) We first state Theorem 3 in full. The regularity conditions the main text defers concern only the critic.

Theorem (Theorem 3, full statement)**.** *Let $Q(s,\cdot)$ be continuously differentiable, let $\pi^\star \propto e^{Q/\lambda}$ with $\mathcal{Z}(s) = \int e^{Q/\lambda}\,\mathrm{d}x < \infty$ have finite second moments and $\mathbb{E}_{\pi^\star}\big[\|\nabla_x Q(s,X_1)\|^2\big] < \infty$, and let $w(t) > 0$ on $(0,1)$. Consider the iteration Equation (12) in its population form, in which the regression is solved exactly over all velocity fields at every $t \in (0,1)$. If $\pi_{\theta_k}(\cdot \mid s) = \pi^\star(\cdot \mid s)$, then $v_{\theta_{k+1}}$ is the velocity field v of Equation (4), $\pi_{\theta_{k+1}}(\cdot \mid s) = \pi^\star(\cdot \mid s)$, and $v_{\theta_{k+m}} = v_{\theta_{k+1}}$ for all $m \geq 1$.*

The two moment conditions make the regression target of Equation (11) square integrable at every $t \in (0,1)$, so that the regression at each time is well posed, and they imply the integrability assumption of Proposition 1.

Proof of Theorem 3. The loss Equation (11) depends on the current parameters θ_k only through the distribution $\pi_{\theta_k}(\cdot \mid s)$ of the clean actions X_1. For $\pi_{\theta_k} = \pi^\star$ it therefore coincides with the target flow matching objective Equation (10), whose minimizer over all velocity fields is the velocity field v of the interpolant of $\pi^\star$ by Corollary 2; hence $v_{\theta_{k+1}} = v$. Integrating Equation (5) with v transports the prior to $\pi^\star$ (Section B.1), so $\pi_{\theta_{k+1}} = \pi^\star$. The next update therefore again minimizes Equation (10) and returns v, and by induction does every later one. □

The behavior-tilted target The argument applies verbatim to the offline-to-online objective Equation (14). There the target is $\pi^\star \propto p_{\mathrm{ref}}\, e^{Q/\lambda}$, whose terminal score is $g = \mathsf{s}_{\mathrm{ref}} + \nabla_x Q/\lambda$, and the regression target of Equation (14) is $\frac{\dot\alpha(t)}{\alpha(t)} X_t + \frac{\kappa(t)}{\alpha(t)}\, g(X_1)$. Applying the target score identity Equation (21) to $\mu = \pi^\star$, with $\nabla \log \pi^\star = g$ since the normalizer of $\pi^\star$ is constant in x, and inserting it into the bridge Equation (28) shows that under sampling from $\pi^\star$ the conditional expectation of this target is the velocity field of $\pi^\star$. The proof above then goes through with $\nabla_x Q/\lambda$ replaced by g, under the conditions of the theorem with Q replaced by $Q + \lambda \log p_{\mathrm{ref}}$.

B Further method details

This appendix collects details of the method that the main text defers: the background on one-sided stochastic interpolants that the main text and the proofs of Section A rely on (Section B.1), the family of regression targets that mixes the FFM target with the flow-matching target and its variance-optimal member (Section B.2), and the correlated exploration scheme of Section 5 (Section B.3).

B.1 One-sided stochastic interpolants

This subsection collects the background on one-sided stochastic interpolants (Albergo et al., 2025) that the main text and the proofs of Section A rely on: the score of the noised marginals, the denoising identities that express it as a conditional expectation, the velocity–score bridge that eliminates the score from all main-text formulas, and the deterministic and stochastic samplers that share the marginals of the flow. All identities are standard and we state them without proof. Throughout, μ is a probability density on $\mathbb{R}^d$ (an action distribution; the conditioning state s is suppressed), $X_1 \sim \mu$ and $\varepsilon \sim \mathcal{N}(\cdot;\, 0,\, I)$ are independent, and

$$X_t = \alpha(t) X_1 + \beta(t)\varepsilon, \qquad \alpha(0) = 0,\;\; \alpha(1) = 1,\;\; \beta(0) = \sigma_0,\;\; \beta(1) = 0, \tag{25}$$

with α, β continuously differentiable, $\alpha(t) > 0$ on $(0, 1]$ and $\beta(t) > 0$ on $[0, 1)$. Expectations $\mathbb{E}_\mu[\cdot]$ are over the pair (X_1, ε), that is, over the interpolant of μ; they abbreviate the main-text notation $\mathbb{E}_{X_1 \sim \mu,\, \varepsilon}$ of Equation (4). We write ρ_t for the density of X_t, which is the Gaussian convolution $\rho_t(x) = \int \mu(x_1)\, \varphi_{\beta^2(t)}(x - \alpha(t)x_1)\, \mathrm{d}x_1$ with $\varphi_{\beta^2(t)}$ the density of $\mathcal{N}\big(\cdot;\, 0,\, \beta^2(t) I\big)$, and we call

$$\mathsf{s}(x, t) \;:=\; \nabla_x \log \rho_t(x) \tag{26}$$

the *score* of the noised marginals. It is the second field, next to the velocity Equation (4), around which the constructions of diffusion and flow models revolve.

Denoising identities Since $X_t \mid X_1 \sim \mathcal{N}\big(\cdot;\, \alpha(t)X_1,\, \beta^2(t) I\big)$, the score of the Gaussian-smoothed marginal is a conditional average of the noise, and the posterior mean of the clean sample is an affine function of the score. Written for the interpolant Equation (25), Tweedie's formula (Robbins, 1956; Efron, 2011) reads

$$\mathsf{s}(x, t) = -\frac{1}{\beta(t)}\, \mathbb{E}_\mu[\varepsilon \mid X_t = x], \qquad \hat{x}_1(x, t) := \mathbb{E}_\mu[X_1 \mid X_t = x] = \frac{x + \beta^2(t)\, \mathsf{s}(x, t)}{\alpha(t)}, \tag{27}$$

where the second identity follows from the first by taking the conditional expectation of $x = \alpha(t)X_1 + \beta(t)\varepsilon$ given $X_t = x$. We refer to the first identity as the *noise identity* and to $\hat{x}_1$ as the *denoiser*; the noise identity is also what turns score matching into the denoising regression of Vincent (2011).

The velocity–score bridge and sampling The velocity Equation (4) is the conditional mean of $\dot{X}_t = \dot{\alpha}(t)X_1 + \dot{\beta}(t)\varepsilon$, so inserting the two identities of Equation (27) expresses it through the score,

$$v(x,t) = \dot{\alpha}(t)\,\mathbb{E}_\mu[X_1 \mid X_t = x] + \dot{\beta}(t)\,\mathbb{E}_\mu[\varepsilon \mid X_t = x] = \frac{\dot{\alpha}(t)}{\alpha(t)}\,x + \kappa(t)\,\mathsf{s}(x,t), \tag{28}$$

with the coefficient $\kappa(t) = \frac{\dot{\alpha}(t)}{\alpha(t)}\beta^2(t) - \dot{\beta}(t)\beta(t) > 0$ of Equation (8), and conversely

$$\mathsf{s}(x,t) \;=\; \frac{1}{\kappa(t)}\Big(v(x,t) - \tfrac{\dot{\alpha}(t)}{\alpha(t)}\,x\Big). \tag{29}$$

For the one-sided interpolant either field therefore determines the other pointwise (Albergo et al., 2025): the *velocity–score bridge*. Equation (29) is how the main text reads the reference's terminal score off a pretrained velocity field in Equation (13); since $\kappa(t) \to 0$ as $t \to 1$, the score is evaluated at a time close to, but not at, the clean endpoint.

The marginals ρ_t solve the continuity equation $\partial_t \rho_t + \nabla\cdot(\rho_t\, v) = 0$ (Albergo et al., 2025), so integrating the ODE Equation (5) from the prior $\rho_0 = \mathcal{N}\big(\cdot;\, 0,\, \sigma_0^2 I\big)$ transports it to $\rho_1 = \mu$: this is the flow policy of the main text. The same marginals are shared by the family of SDEs

$$\mathrm{d}X_t \;=\; \Big(v + \tfrac{\sigma^2(t)}{2}\,\mathsf{s}\Big)(X_t,t)\,\mathrm{d}t + \sigma(t)\,\mathrm{d}W_t, \qquad X_0 \sim \mathcal{N}\big(\cdot;\, 0,\, \sigma_0^2 I\big)\,, \tag{30}$$

for any diffusion coefficient $\sigma(t) \geq 0$, because the additional drift $\frac{\sigma^2(t)}{2}\mathsf{s}$ cancels the diffusion term in the Fokker–Planck equation and leaves the continuity equation of the flow (Song et al., 2021; Albergo et al., 2025). A trained velocity field can hence be sampled either deterministically by Equation (5) or stochastically by Equation (30), with the score supplied by the bridge Equation (29), and the time reversal of Equation (30), with drift $v - \frac{\sigma^2(t)}{2}\mathsf{s}$ and run from $t = 1$ to 0, transports μ back to the prior (Song et al., 2021). We use the ODE throughout.

B.2 Generalized regression targets and control variates

The main text regresses the velocity head onto the target of Equation (10), in which the critic gradient is, besides X_t, the only random quantity. This subsection places that target in a one-parameter family that mixes it with the flow-matching target, and states the member of the family with the smallest variance. The family is most transparent in score space, where it is the generalized target score identity of the bridge matching sampler (Blessing et al., 2026), and transfers to velocity space through the bridge Equation (28); the velocity form is what our offline-to-online implementation uses (Section E.3). We state the results without proofs, which are given in Blessing et al. (2026) and follow the pattern of Section A.

The generalized target score identity The noise identity Equation (27) and the target score identity Equation (21) express the same score $\mathsf{s}^\mu(x,t)$ as conditional expectations of two different random variables, the scaled noise $-\varepsilon/\beta(t)$ and the scaled target score $\nabla \log \mu(X_1)/\alpha(t)$. Any convex combination of the two is therefore again conditionally unbiased: for a mixing schedule $c(t) \in [0,1]$,

$$\mathsf{s}^\mu(x,t) \;=\; \mathbb{E}_\mu\big[\zeta(X_1,\varepsilon,t)\,\big|\,X_t = x\big], \qquad \zeta(X_1,\varepsilon,t) \;:=\; \frac{c(t)}{\alpha(t)}\,\nabla \log \mu(X_1) - \frac{1-c(t)}{\beta(t)}\,\varepsilon, \tag{31}$$

which is the *generalized target score identity* of Blessing et al. (2026); $c \equiv 1$ recovers Equation (21) and $c \equiv 0$ the noise identity. The two random variables have complementary variances: the target-score term carries the factor $1/\alpha(t)$, which diverges at the noise end $t \to 0$, and the noise term the factor $1/\beta(t)$, which diverges at the clean end $t \to 1$. Since the conditional mean of the noise term is known, it acts as a *control variate* for the target-score term, and the mixing schedule can be chosen to minimize the variance of the combined estimator. For a schedule that depends on t alone, the relevant quantity is the variance averaged over X_t, $\mathbb{E}_\mu\big[\|\zeta - \mathsf{s}^\mu(X_t,t)\|^2\big] = \mathbb{E}_\mu\big[\|\zeta\|^2\big] - \mathbb{E}_\mu\big[\|\mathsf{s}^\mu(X_t,t)\|^2\big]$, whose second term does not depend on c. Because X_1 and ε are independent, the cross term of the first vanishes and

$$\mathbb{E}_\mu\big[\|\zeta(X_1,\varepsilon,t)\|^2\big] \;=\; d\,\Big(\frac{c^2(t)}{\alpha^2(t)}\,\bar{\mathcal{I}} + \frac{(1-c(t))^2}{\beta^2(t)}\Big), \qquad \bar{\mathcal{I}} := \tfrac{1}{d}\,\mathbb{E}_\mu\big[\|\nabla \log \mu(X_1)\|^2\big], \tag{32}$$

where $\bar{\mathcal{I}}$ is the per-coordinate Fisher information of μ. Minimizing Equation (32) over $c(t)$ yields (Blessing et al., 2026)

$$c^\star(t) \;=\; \frac{\alpha^2(t)}{\alpha^2(t) + \beta^2(t)\,\bar{\mathcal{I}}}, \tag{33}$$

for any target μ with finite Fisher information; in practice, $\bar{\mathcal{I}}$ is estimated on the current batch. Its shape is that of an inverse signal-to-noise ratio: the target-score term dominates near $t = 1$, where samples are clean, and the noise anchor near $t = 0$.

The generalized target flow identity The velocity image of the family Equation (31) has a direct reading. Two velocity targets are available for the interpolant Equation (25), the flow-matching target $\dot{X}_t$ of Equation (6) and the target-flow-identity target of Equation (10), and their convex combination is

$$Y_c(X_1, \varepsilon, t) \;:=\; \big(1 - c(t)\big)\,\dot{X}_t \;+\; c(t)\left(\tfrac{\dot\alpha(t)}{\alpha(t)}\,X_t + \tfrac{\kappa(t)}{\lambda\,\alpha(t)}\,\nabla_x Q(s, X_1)\right), \qquad c(t) \in [0, 1]. \tag{34}$$

Under sampling from the Boltzmann policy, both endpoints have the same conditional mean, the velocity field generating $\pi^\star$ (Equation (4) for $c \equiv 0$ and Proposition 1 for $c \equiv 1$), which gives the *generalized target flow identity*

$$v(x, t \mid s) \;=\; \mathbb{E}_{X_1 \sim \pi^\star,\, \varepsilon}\big[\,Y_c(X_1, \varepsilon, t) \,\big|\, X_t = x\big] \qquad \text{for every } c(t) \in [0, 1]. \tag{35}$$

Substituting $\dot{X}_t = \dot\alpha(t) X_1 + \dot\beta(t)\varepsilon$, $X_t = \alpha(t)X_1 + \beta(t)\varepsilon$, and the schedule identity $\frac{\dot\alpha(t)}{\alpha(t)}\beta(t) = \dot\beta(t) + \frac{\kappa(t)}{\beta(t)}$ into Equation (34) shows that the family is exactly the velocity image of the score-space target ζ of Equation (31),

$$Y_c \;=\; \tfrac{\dot\alpha(t)}{\alpha(t)}\,X_t \;+\; \kappa(t)\,\zeta(X_1, \varepsilon, t), \tag{36}$$

so Equation (35) is the bridge applied to Equation (31). By Equation (35), the proof of Theorem 3 carries over to the mixed target: under sampling from $\pi^\star$ the regression onto Y_c returns the velocity field of $\pi^\star$, so $\pi^\star$ remains a fixed point for every mixing schedule $c(t) \in [0, 1]$. The map $\zeta \mapsto Y_c$ is affine with a shift that depends only on X_t and a scale $\kappa(t)$ that does not depend on c, so $\mathrm{Var}(Y_c \mid X_t = x) = \kappa^2(t)\,\mathrm{Var}(\zeta \mid X_t = x)$ and the two families share their variance-minimizing schedule:

$$c^\star(t) \;=\; \frac{\alpha^2(t)}{\alpha^2(t) + \beta^2(t)\,\bar{\mathcal{I}}}, \qquad \bar{\mathcal{I}} = \tfrac{1}{d}\,\mathbb{E}_q\big[\|\nabla_x Q/\lambda\|^2\big], \tag{37}$$

is the variance-optimal control variate also in velocity space. It also removes the endpoint singularity noted in Section 4: in Equation (34) the critic gradient enters with the coefficient $c(t)\,\kappa(t)/(\lambda\,\alpha(t))$, and since $c^\star(t) = \Theta(\alpha^2(t))$ as $t \to 0$ while $\alpha(t)\,\kappa(t) \to \dot\alpha(0)\,\sigma_0^2 < \infty$, this coefficient stays bounded at the noise end, whereas $c^\star(t) \to 1$ as $t \to 1$ hands the target back to the pure FFM form where actions are clean. No time truncation $t \geq \epsilon_0$ is needed; our offline-to-online implementation uses this schedule (Section E.3).

B.3 Correlated exploration

This subsection expands on the correlated exploration scheme of Section 5.

The flow policy as a deterministic map Integrating the ODE Equation (5) with the learned velocity field from the prior draw $X_0 = \sigma_0\varepsilon$ up to $t = 1$ is a deterministic computation once ε and the state s are fixed. We write $\Psi_\theta(\varepsilon \mid s)$ for the resulting clean action, so that $\Psi_\theta(\cdot \mid s)$ is the sampler of the flow policy: it maps a prior draw to an action, and $\pi_\theta(\cdot \mid s)$ is the law of $\Psi_\theta(\varepsilon \mid s)$ for $\varepsilon \sim \mathcal{N}(\cdot;\, 0,\, I)$. In particular, an action is a sample of $\pi_\theta(\cdot \mid s)$ whenever the prior draw it is computed from is distributed according to $\mathcal{N}(\cdot;\, 0,\, I)$, regardless of how that draw was generated.

The Ornstein–Uhlenbeck chain is stationary The chain Equation (20) is a first-order autoregressive process with coefficient $a := e^{-1/\tau_{\mathrm{ou}}} \in (0, 1)$, that is, $\varepsilon_{k+1} = a\,\varepsilon_k + \sqrt{1 - a^2}\,\xi_k$. If $\varepsilon_k \sim \mathcal{N}(\cdot;\, 0,\, I)$ and $\xi_k \sim \mathcal{N}(\cdot;\, 0,\, I)$ are independent, then ε_{k+1} is Gaussian with mean zero and covariance

$$\mathbb{E}\big[\varepsilon_{k+1}\varepsilon_{k+1}^\top\big] \;=\; a^2\,I + (1 - a^2)\,I \;=\; I, \tag{38}$$

so $\varepsilon_{k+1} \sim \mathcal{N}(\cdot; 0, I)$ as well. Since $\varepsilon_0 \sim \mathcal{N}(\cdot; 0, I)$, induction over k shows that every ε_k is marginally standard normal, and by the previous paragraph every executed action $x_k = \Psi_\theta(\varepsilon_k \mid s_k)$ is a sample of $\pi_\theta(\cdot \mid s_k)$. What changes is the dependence between draws: unrolling the recursion gives

$$\mathbb{E}\big[\varepsilon_{k+j}\varepsilon_k^\top\big] \;=\; a^j\, I \;=\; e^{-j/\tau_{\mathrm{ou}}}\, I, \qquad j \geq 0, \tag{39}$$

so the correlation between two prior draws decays exponentially with their distance in time, with τ_{ou} steps as the time constant. The two limits recover familiar schemes: $\tau_{\mathrm{ou}} \to 0$ gives $a \to 0$ and independent draws at every step, which is the usual way of sampling the policy, and $\tau_{\mathrm{ou}} \to \infty$ gives $a \to 1$ and a single draw ε_0 that is kept for the whole episode, similar to the state-dependent exploration of Raffin et al. (2022).

C Extended related work

This appendix complements Section 2. The first paragraph places FFM in the family of matching objectives that train diffusion and flow models from an unnormalized target rather than from samples, which is where its fixed-point structure comes from. The second collects work on critic-driven updates of diffusion and flow policies that Section 2 groups but has no room to discuss individually.

Fixed-point methods for diffusion and flow models. Training a diffusion or flow model from an unnormalized density instead of samples is the problem of neural samplers. Early approaches minimize a path-space divergence by simulating the generative process (Zhang & Chen, 2022; Vargas et al., 2023). Matching-based samplers instead regress onto a target computed from the energy and its gradient: iDEM (Akhound-Sadegh et al., 2024), iEFM (Woo & Ahn, 2024), BNEM (OuYang et al., 2026), and EWFM (Dern et al., 2025) estimate the noised score, velocity, or energy by importance-weighted Monte Carlo, target score matching (De Bortoli et al., 2024), on which FFM builds, writes the noised score as a conditional expectation of the clean-data score but requires samples of the target, and tilt matching (Potaptchik et al., 2025) matches the velocity of a reward-tilted interpolant without reward gradients. These methods are iterative: the target is evaluated with samples of the current model and the regression is repeated, which is the fixed-point structure that Theorem 3 makes explicit for critic-driven targets. Adjoint matching (Domingo-Enrich et al., 2025) forms a second family of such regressions. Proposed for reward fine-tuning, it has since been used for sampling from unnormalized densities (Havens et al., 2025; Liu et al., 2025a; Mosso et al., 2026), for discrete generative models (So et al., 2026), for faster fine-tuning pipelines (Guo et al., 2026; Shin et al., 2026a), and in RL for policy extraction (Li & Levine, 2026; Dong et al., 2026b; Ghanem & Ghogho, 2026) and maximum entropy RL (Thilges et al., 2026); its theory has been connected to the stochastic maximum principle (Domingo-Enrich & Han, 2026) and unified with sampling (Domingo-Enrich et al., 2026). The bridge matching sampler (Blessing et al., 2026) shows that these matching objectives, including adjoint sampling and ASBS, are fixed-point iterations rooted in Nelson's relation and stabilizes them by damping. FFM belongs to this family with two differences: the target flow identity provides the regression target from critic gradients at clean actions, so no adjoint ODE is needed, and the target itself moves as the critic is learned.

Further critic-driven policy updates. Sorted by how the critic enters the actor update, the methods of Section 2 fall into a few families, and we record here the entries the main text omits. Methods based on backpropagation through time differentiate the critic through the entire sampling chain: besides those cited in Section 2, Lv et al. (2025) constrain the update relative to a behavior policy. GenPO (Ding et al., 2025) instead applies a PPO-style objective (Schulman et al., 2017) to a diffusion policy whose likelihood is made tractable by a coupling construction. A second family avoids the chain by regressing onto improved *actions*: DIPO (Yang et al., 2023) and Li et al. (2024) improve the actions stored in the replay buffer by gradient ascent on Q and then clone them with the standard denoising loss. This is the closest relative of FFM among the action-target methods, and the difference is that its regression target is an action rather than a velocity, so the noise level of the interpolant is not accounted for. A third family leaves the denoising loss in place and weights it by the exponentiated critic value (Ding et al., 2024; Ma et al., 2025; Dong et al., 2025), of which one-step flow policy mirror descent (Chen et al., 2025) is the one-step variant, and flow map policies trained by mirror descent (Dong et al., 2026a; Gao et al., 2026) are the multimodal ones. None of these families obtains its target from the velocity field of the Boltzmann policy itself, which is what the target flow identity provides.

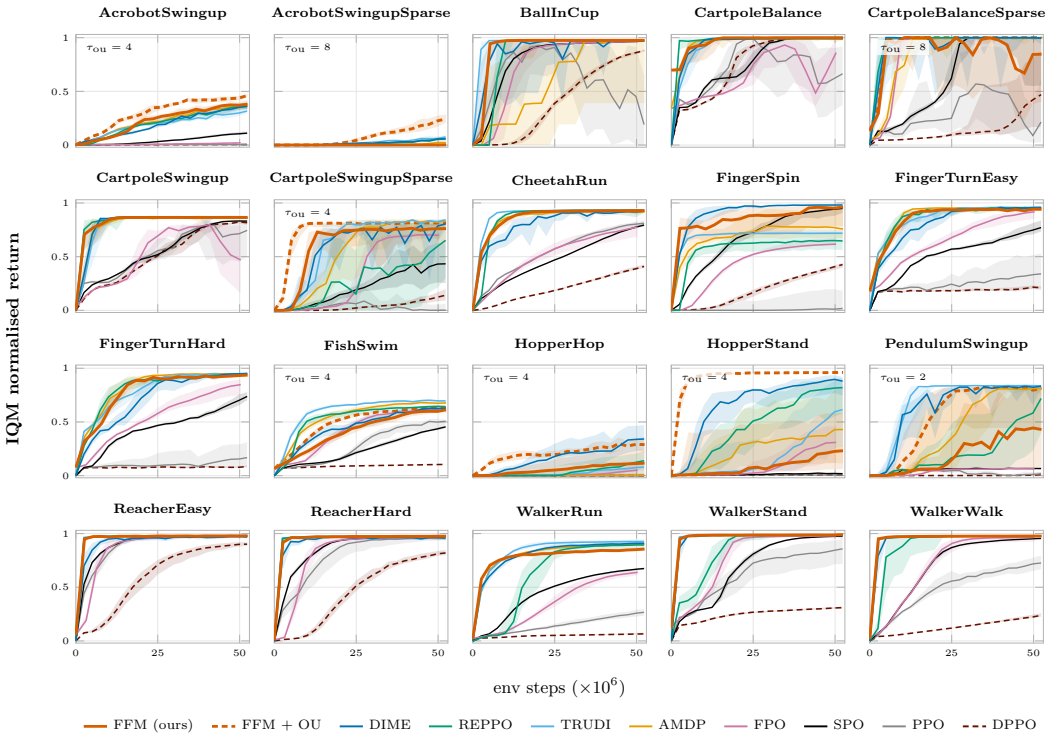


Figure 4: **Maximum Entropy Online RL: MJX Playground tasks.** We plot the IQM per-task learning curves with 95% stratified bootstrapped confidence intervals (Agarwal et al., 2021) over 10 seeds. On most of the tasks, FFM performs similarly to or on par with baselines. However, we observe that there are some tasks such as *AcrobotSwingupSparse, HopperHop, HopperStand*, where FFM (and most of the baselines) perform badly, indicating that a more sophisticated exploration strategy is required. On a subset of the tasks, we have therefore run FFM with correlated exploration (FFM + OU), where we show the chosen τ_{ou} hyperparameter on the respective plot. Please note that correlated exploration is only possible due to FFM's deterministic mapping (Section 5) Correlated exploration can help improve performance, such as in *HopperHop*, or even more significantly in the *HopperStand* task. The aggregated results when using the correlated exploration curves whenever applicable are given in Figure 3 (left two panels).

D Maximum-entropy reinforcement learning

This section provides information on FFM's implementation in the online on-policy maximum-entropy RL setting as proposed in Section 5. We review the maximum-entropy actor-critic variant similar to that of Haarnoja et al. (2018), where we optimize the state-action value function and the velocity field's parameters. Moreover, we give an overview of how FFM resolves bounding final actions that are executed in the environment, provide an algorithm box, report ablations on design choices, and provide detailed per-task results.

D.1 Actor-Critic based Maximum-entropy reinforcement learning

In maximum-entropy RL (Ziebart et al., 2008; Levine, 2018; Haarnoja et al., 2017; 2018), the rewards are augmented with the current policy's entropy, leading to the objective

$$J(\pi) \;=\; \mathbb{E}_{\pi}\Big[\sum_{k\geq 0}\gamma^{k}\big(r(s_k,x_k)+\lambda\,\mathcal{H}(\pi(\cdot\mid s_k))\big)\Big], \tag{40}$$

where $\lambda > 0$ trades off maximizing the expected task return against maximizing the policy's stochasticity by maximizing the entropy $\mathcal{H}(\pi(\cdot \mid s_k))$. The soft state-action value function is given as

$$Q^{\pi}(s,x) = \mathbb{E}_{\pi}\big[\sum_{k\geq 0}\gamma^{k}\big(r(s_k,x_k)+\lambda\,\mathcal{H}(\pi(\cdot\mid s_k))\big)\mid s_0=s,\, x_0=x\big], \tag{41}$$

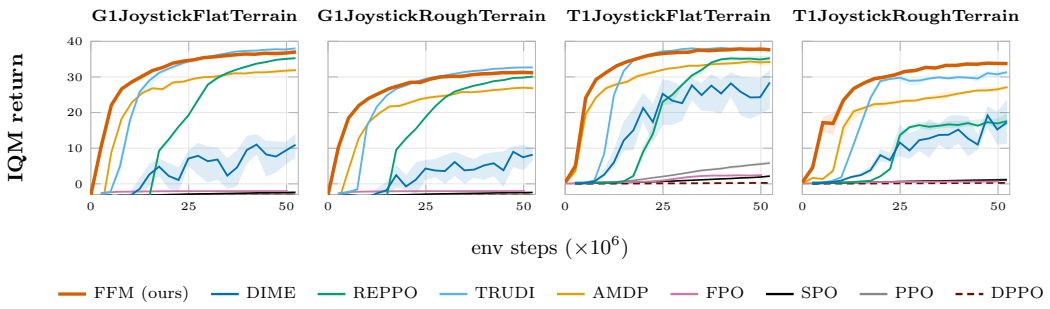


Figure 5: **Maximum Entropy Online RL: MJX Humanoid tasks.** We plot the IQM per-task learning curves with 95% stratified bootstrapped confidence intervals (Agarwal et al., 2021) over 10 seeds. FFM converges faster to a similar performance level as TRUDI and performs significantly better than TRUDI on the *T1JoystickRoughTerrain* task. The high-dimensional humanoid tasks showcase FFM's benefits more clearly.

which can also be expressed using the recursive Bellman equation as

$$Q^\pi(s,x) = r(s,x) + \gamma\, \mathbb{E}_{s'\sim P, x'\sim\pi}\big[Q^\pi(s',x') - \lambda \log \pi(x' \mid s')\big]. \tag{42}$$

Based on this equation, we can then update the parameters of Q^π using the Bellman residual,

$$J_Q(\phi) = \mathbb{E}\left[\frac{1}{2}\left(Q^\pi_\phi(s,x) - y\right)^2\right] \tag{43}$$

where ϕ are the parameters of the critic network that we have omitted so far to avoid cluttered notation, and y are the critic targets, that are calculated by

$$y = r(s,x) + \gamma \mathbb{E}_{s'\sim p, x'\sim\pi}\left[Q^\pi_{\bar\phi}(s',x') - \lambda \log \pi(x' \mid s')\right] \tag{44}$$

for the one-step lookahead, mainly used in the off-policy RL setting (Haarnoja et al., 2018; Fujimoto et al., 2018). Here, $\bar\phi$ denotes the target network parameters, a Polyak averaging of the actual parameters ϕ.

In the on-policy setting, commonly no target network is used, and the target is calculated using N-step returns based on a Monte Carlo estimate with samples from the current rollout and the current target as

$$y = \sum_{k=t}^{N} \gamma^{k-t}\left(r(s_k,x_k) - \lambda \log \pi(x_k \mid s_k)\right) + \gamma^{N+1} Q_\phi(s_N, x_N), \tag{45}$$

as suggested, for example, by Voelcker et al. (2026) and used in our online on-policy methods. Note that the gradients for the critic's parameters ϕ are stopped for calculating the targets in Equation (45). For calculating the entropy and the likelihoods of the current policy, we follow Equation (15). Once the parameters of the Q-function are updated in the policy evaluation step, the parameters of the policy π are updated in the policy improvement step such that the expected soft Q-function is maximized. In the standard case, e.g., with Gaussian policies, the objective for the policy improvement step follows from

$$L(\pi) = D_{\mathrm{KL}}\left(\pi(x \mid s) \,\|\, \frac{\exp\big(Q(s,x)/\lambda\big)}{\mathcal{Z}(s)}\right), \tag{46}$$

where $\mathcal{Z}(s) = \int \exp\big(Q(s,x)/\lambda\big)\,\mathrm{d}x$ is the normalizing constant and independent of the action x. It can be shown that alternating both steps converges to the optimal maximum-entropy policy in the tabular setting (Haarnoja et al., 2017). For FFM to optimize the flow policy, we rely on Equation (17).

Enforcing action bounds In RL benchmarks and potentially in real-world use cases, the applied actions are bounded due to physical constraints such as bounded motor torques. FFM squashes the generated actions to $[-1,1]^d$, to avoid querying the Q-function at unreasonable action regions.

However, please note that we keep the flow unconstrained on $\mathbb{R}^d$, such that the theory does not change, but we map the flow's clean output X_1 to the environment action through a coordinate-wise squashing function g

$$x_{\text{env}} = g(X_1), \tag{47}$$

where, in the literature (Haarnoja et al., 2018), the $\tanh$ function is used as

$$g(z) = \tanh(z), \tag{48}$$

with the log derivative

$$\log g'_{\tanh}(z) = \log\left(1 - \tanh^2 z\right). \tag{49}$$

Recently, Thilges et al. (2026) proposed an alternative squashing function and reported that using the error function

$$g(z) = \operatorname{erf}(z/\sqrt{2}) \tag{50}$$

yields improved numerical behavior. Its log derivative is given by

$$\log g'_{\text{erf}}(z) = \tfrac{1}{2}\log\tfrac{2}{\pi} - \tfrac{z^2}{2}. \tag{51}$$

However, please note that squashing the actions changes the likelihood calculations by the change-of-variables rule, such that the likelihood of the new distribution is

$$\pi_{\text{env}}(\cdot \mid s) = \pi(\cdot \mid s)\left|\det\left(\frac{dx_{\text{env}}}{dx}\right)\right|^{-1}, \tag{52}$$

which leads to the log-likelihoods needed for calculating the entropy values as

$$\log \pi_{\text{env}}(\cdot \mid s) = \log \pi(\cdot \mid s) - \sum_i \log g'(X_{1,i}), \tag{53}$$

where, depending on which squashing function is used, either Equation (49) or Equation (51) is used for the Jacobian in Equation (52). This log-likelihood is required to estimate the entropy (Equation (15)) over a batch of samples during the rollout to calculate the soft Q-function in maximum-entropy RL.

Because the FFM target from Equation (10) is w.r.t. the clean action X_1 rather than the squashed action x_{env}, a consequence of squashing the clean actions is that, by the chain rule, the critic's gradient w.r.t. X_1 is

$$\nabla_{X_1} Q(s, g(X_1)) = g'(X_1) \odot \nabla_{x_{\text{env}}} Q(s, x_{\text{env}}), \tag{54}$$

where $\odot$ denotes coordinate-wise multiplication. In our experiments, we have used the error function as the squashing function, although, as shown in our ablation studies, the tanh squashing only performs slightly worse (Section D.4).

D.2 PSEUDOCODE

Algorithm 1 Online on-policy maximum-entropy RL with an FFM actor

input: Initialized parameters for velocity field v_θ, critic Q_ϕ, temperature λ with entropy target $\mathcal{H}^\star$, trust-region budget $\varepsilon_{\mathrm{TR}}$ with multiplier ν, schedules α, β (Equation (57)), time weighting w
for each iteration **do**
  rollout (on-policy, no gradients):
    • act $x_k \sim \pi_\theta(\cdot \mid s_k)$ by integrating Equation (5)
    • optionally, correlate the prior draws via the OU chain of Equation (20)
    • collect $\{(s_k, x_k, r_k, s_{k+1})\}_{k=t}^{N}$
  likelihoods: $\log \pi_\theta(x_k \mid s_k)$ by Equation (15), divergences by Hutchinson's estimator
  policy evaluation: calculate N-step soft returns of Equation (45) (target gradients stopped)

$$y_t = \sum_{k=t}^{N} \gamma^{k-t}\big(r(s_k, x_k) - \lambda \log \pi_\theta(x_k \mid s_k)\big) + \gamma^{N+1} Q_\phi(s_N, x_N)$$

  for each critic epoch over minibatches of the rollout **do**
    update critic parameters ϕ using $\big(Q_\phi(s_t, x_t) - y_t\big)^2$, or if distributional using HL-Gauss (Farebrother et al., 2024; Voelcker et al., 2026)
  Calculate Q-scores: $\nabla_x Q_\phi(s, X_1)$ at the rollout actions $X_1 \sim \pi_{\bar\theta}(\cdot \mid s)$
  for each actor epoch over minibatches of the rollout **do**
    $t \sim \mathcal{U}(0,1)$, $\varepsilon \sim \mathcal{N}(\cdot;\, 0,\, I)$, $X_t \leftarrow \alpha(t) X_1 + \beta(t)\varepsilon$
    minimize over θ using Equation (18):

$$\mathbb{E}_{t, X_1, \varepsilon}\left[w(t)\left\| v_\theta(X_t, t \mid s) - \tfrac{\dot\alpha(t)}{\alpha(t)} X_t - \tfrac{\kappa(t)}{\lambda\, \alpha(t)} \nabla_x Q_\phi(s, X_1)\right\|^2\right] + \nu\big(\mathcal{L}_{\mathrm{TR}}(\theta, \bar\theta) - \varepsilon_{\mathrm{TR}}\big)$$

    with $\mathcal{L}_{\mathrm{TR}}$ of Equation (16) evaluated at the same points (X_t, t)
    duals:
    • ascend ν (Equation (19))
    • descend λ on $\lambda\big(\mathbb{E}[\widehat{\mathcal{H}}] - \mathcal{H}^\star\big)$, so λ grows while the entropy is below $\mathcal{H}^\star$
  update the trust-region reference: $\bar\theta \leftarrow \theta$
evaluation: integrate Equation (5) from $X_0 = 0$ (Section D.4)

D.3 EXPERIMENTAL DETAILS

Environments and tasks. In our online on-policy RL experiments from Section 6 we have evaluated FFM on the GPU-accelerated MuJoCo Playground (Zakka et al., 2025) tasks. More concretely, we distinguish between the 20 DeepMind Control (DMC) tasks of the suite, whose episode returns are bounded between $[0, 1000]$, and the four MJX humanoid joystick tasks, which require precise and high-dimensional whole-body control and whose returns are not normalized.

For every learning curve, we follow Agarwal et al. (2021) and report the interquartile mean (IQM) of the return with stratified bootstrapped 95% confidence intervals over 10 seeds per task. Furthermore, in Figure 3 the aggregated results over all tasks for the Playground and the Humanoid tasks are reported, where the per-task results are shown in Section D.4.

Baselines. We compare against Gaussian-policy methods, REPPO (Voelcker et al., 2026), PPO (Schulman et al., 2017), and SPO (Xie et al., 2025), diffusion-policy methods, TRUDI (Le et al., 2026), AMDP (Thilges et al., 2026), DIME (Celik et al., 2025), and DPPO (Ren et al., 2025), and the flow-policy method FPO (McAllister et al., 2026).

Hyperparameters. Table 1 lists the hyperparameters of FFM and the baselines. Notably, for FFM the flow starts from the prior $X_0 \sim \mathcal{N}\big(\cdot;\, 0,\, \sigma_0^2 I\big)$ with $\sigma_0 = 1$ under the geometric schedule of Equation (57) and is integrated with 16 Euler steps. Furthermore, FFM employs action squashing using the error function map from Section D.1 and the entropy estimate Equation (15) from Section 5 with four Rademacher probes per integration node.

Implementation details. *Distributional critic.* As done in most baselines and related works recently, we parameterize the critic distributionally following HL-Gauss (Farebrother et al., 2024), trained by cross-entropy on the same targets as Equation (45).

Actor regression. The implemented policy-improvement step is the FFM regression Equation (11) with the regression target replaced by the variance-optimal mixture of Equation (34) and the time weighting fixed to $w(t) = 1/\kappa^2(t)$,

$$\mathcal{L}(\theta, \bar{\theta}) \;=\; \mathbb{E}_{t,\, X_1 \sim \pi_{\bar{\theta}},\, \varepsilon}\Big[w(t)\, \big\| v_\theta(X_t, t \mid s) - Y_{c^\star}(X_1, \varepsilon, t) \big\|^2\Big], \qquad w(t) \;=\; \frac{1}{\kappa^2(t)}, \tag{55}$$

where $Y_{c^\star}$ is the velocity target Equation (34) at the variance-optimal mixing schedule of Equation (37). The variance-optimal mixing schedule is given as $c^\star(t) = \alpha^2(t)/\big(\alpha^2(t) + \beta^2(t)\, \bar{\mathcal{I}}\big)$ (Equation (37)), but we fixed the Fisher scale to $\bar{\mathcal{I}} = 1$, such that we obtain $c^\star(t) = \alpha^2(t)/\big(\alpha^2(t) + \beta^2(t)\big)$. Since the generated actions are squashed (Section D.1), the Q-score $\nabla_x Q(s, X_1)/\lambda$ in $Y_{c^\star}$ becomes $\nabla_x \big[Q(s, g(x))/\lambda + \log|\det \partial g(x)|\big]\big|_{x=X_1}$, which is the score of the squashed targets. For $c^\star = 1$, Equation (55) is Equation (11). Please note that multiplying the residual by $\sqrt{w(t)} = 1/\kappa(t)$ converts the velocity difference into a score difference through the bridge Equation (28), so Equation (55) is the score-field regression of Section B.2. The regression Equation (55) in its score-field form is

$$\mathcal{L}(\theta, \bar{\theta}) \;=\; \mathbb{E}_{t,\, X_1 \sim \pi_{\bar{\theta}},\, \varepsilon}\Bigg[\bigg\| \frac{\alpha(t)\,\big(v_\theta(X_t, t \mid s) - \dot{\alpha}(t)\, X_1\big) + a(t)\, \varepsilon}{\alpha(t)\, \kappa(t)} - \frac{c^\star(t)}{\alpha(t)}\, \nabla_x \Big[\tfrac{1}{\lambda} Q\big(s, g(x)\big) + \log|\det \partial g(x)|\Big]\Big|_{x=X_1} \bigg\|^2\Bigg], \tag{56}$$

with the coefficient $a(t) := -\big(c^\star \dot{\alpha}\beta + (1 - c^\star)\, \alpha\dot{\beta}\big)$. However, since $1/\kappa(t)$ diverges as $t \to 1$ for the linear schedule and $1/\alpha(t)$ diverges as $t \to 0$, the implementation softens the two denominators of Equation (56), $\alpha\kappa \to \alpha\kappa + \delta\sigma_0^2$ and $\alpha \to \alpha + \delta$, by the constant $\delta = 0.05$.

Deterministic evaluation. During evaluation, we found that starting the flow's integration from the central point $X_0 = 0$ instead of a random prior draw yields improved performance (Section D.4); training rollouts keep stochastic prior sampling.

Interpolant schedule. We instantiate the schedules α, β with a *geometric* schedule. Given interpolation scales $\sigma_{\max} > \sigma_{\min} > 0$ with ratio $r = \sigma_{\max}/\sigma_{\min}$, we write $\tau(t) = \sigma_{\max}^2\big(1 - r^{-2t}\big)$ and $\tau_T = \sigma_{\max}^2 - \sigma_{\min}^2$, and set

$$\alpha(t) \;=\; \frac{\tau(t)}{\tau_T} \;=\; \frac{1 - r^{-2t}}{1 - r^{-2}}, \qquad \beta(t) \;=\; \sqrt{\tau_T - \tau(t)} \;=\; \sigma_{\max}\sqrt{r^{-2t} - r^{-2}}. \tag{57}$$

The boundary conditions of the one-sided interpolant hold exactly since $\tau(0) = 0$ and $\tau(1) = \sigma_{\max}^2(1 - r^{-2}) = \tau_T$, which gives $\alpha(0) = 0$, $\alpha(1) = 1$, $\beta(1) = 0$, and a prior scale of $\sigma_0 = \beta(0) = \sqrt{\tau_T}$. In our experiments the geometric schedule led to slightly better results (Figures 6 to 8) than the linear schedule $\alpha(t) = t$, $\beta(t) = \sigma_0(1 - t)$. All runs use $\sigma_0 = 1$ and $r = 2.5$, corresponding to $\sigma_{\max} = \sigma_0/\sqrt{1 - r^{-2}} \approx 1.09$ and $\sigma_{\min} = \sigma_{\max}/r \approx 0.44$, identical across all task suites.

D.4 Further experiments and design ablations

Figure 4 and Figure 5 show the per-task learning curves on the 20 DMC tasks and the four humanoid tasks, respectively. An aggregation over all environments can be found in Figure 3.

Interpolant schedule. As detailed in Section D.3, we use a geometric interpolant schedule for the on-policy RL setting. We refer to Section D.3 for the exact parameter setting. A valid and intuitive interpolant is the linear schedule as described in Section 3. However, the schedule type's effect on performance is unclear, which is why we have conducted thorough experiments and report the aggregated results over the 20 Playground tasks together with their final performances and the Humanoid tasks together with their final performances in Figure 6. In Figure 7, we report the per-task results of the Playground suite, and the per-task results on the humanoid environments are

Table 1: Hyperparameters of FFM and of the online baselines.

Parameter	FFM (ours)	AMDP	REPPO	DIME	SPO	PPO	FPO	DPPO
Actor Network								
Hidden dim.	512	512	512	512	256	256	32	256
Hidden layers	3	3	3	3	3	3	5	3
Activation	GeLU	GeLU	GeLU	GeLU	ELU	ELU	SiLU	Mish
Flow/diff. steps	16	16	N/A	16	N/A	N/A	10	8
Critic Network								
Critic type	Dist. (HL-Gauss)	Dist. (HL-Gauss)	Dist. (HL-Gauss)	N/A	MSE	MSE	MSE	MSE
Hidden dim.	512	512	512	512	256	256	256	256
Hidden layers	3	3	3	3	3	3	5	3
Ensemble/bins	151 bins	151 bins	151 bins	100	1	1	1	1
Activation	GeLU	GeLU	GeLU	GeLU	ELU	ELU	SiLU	Mish
Optimization								
Optimizer	Adam	Adam	Adam	Adam	Adam	Adam	Adam	Adam
Actor LR	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}	1×10^{-3}	1×10^{-3}	3×10^{-4}	3×10^{-4}
Critic LR	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}	1×10^{-3}	1×10^{-3}	3×10^{-4}	3×10^{-4}
Max grad. norm	0.5	0.5	0.5	N/A	1.0	1.0	1.0	1.0
Constraint	$\epsilon = 0.1$ KL	$\epsilon = 0.1$ KL	$\epsilon = 0.1$ KL	N/A	$\epsilon = 0.2$ Penalty	$\epsilon = 0.2$ Clip	0.2 Clip	0.2 Clip
Other Settings								
Discount†	0.99 / 0.97	0.99 / 0.97	0.99 / 0.97	0.99 / 0.97	0.99 / 0.97	0.99 / 0.97	0.99 / 0.97	0.99 / 0.97
Batch size	1024	1024	1024	1024	1024	N/A	N/A	N/A
Buffer size	131072	131072	131072	131072	N/A	N/A	N/A	N/A
Target entropy	$-0.1\dim(\mathcal{A})$	$-1.0\dim(\mathcal{A})$	$-0.5\dim(\mathcal{A})$	$-4\dim(\mathcal{A})$	N/A	N/A	N/A	N/A

† All methods use discount 0.99 on the MuJoCo Playground DMC tasks and 0.97 on the humanoid locomotion tasks (G1/T1 Joystick); all other FFM hyperparameters are identical across the two suites.

shown in Figure 8. While there seems to be only a minor, negligible difference in the aggregated results for the Playground tasks (Figure 6 (left)), a clear difference can be seen in favor of the geometric schedule in the aggregated results for the humanoid tasks (Figure 6 (right)). However, our experiments show that both schedules lead to a reasonably well-performing policy on average.

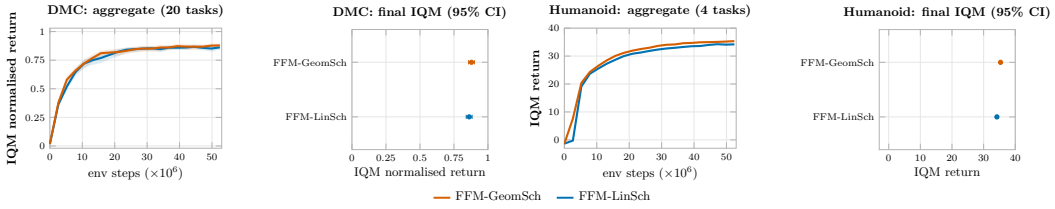


Figure 6: **Ablation. Geometric versus linear interpolant schedule.** As discussed in Section D.4, FFM is not restricted to a specific interpolation schedule type such as the linear schedule. Motivated by diffusion-based methods, we have compared the performance of a linear schedule to the geometric schedule as described in Section D.4 and report the aggregated results over all 20 MJX Playground tasks (left two panels) and the four humanoid tasks (right two panels), where every single task was run on 10 seeds, and we report the IQM and the stratified bootstrapped 95% confidence intervals (Agarwal et al., 2021). While the results indicate that there is barely a performance difference in the Playground tasks, a slight improvement on the humanoid tasks can be seen, which is why we have chosen the geometric schedule as default for the online RL tasks. See Figure 7 and Figure 8 for the per-task results.

Number of integration steps. We further analyze the effect of the number of integration steps on the humanoid tasks' performance. Figure 9 (left) shows the results aggregated over the four MJX humanoid tasks for different numbers of Euler integration steps K. Please note that we only vary K and keep everything else as detailed in Table 1. The corresponding per-task curves to the aggregated results are shown in Figure 10. As expected, the performance increases with the number of integration steps, since the integration error of the flow decreases. However, we observe that performance plateaus at $K = 16$, where increasing to $K = 32$ does not improve performance and yields similar results. Based on this observation and the baseline's hyperparameters, we use

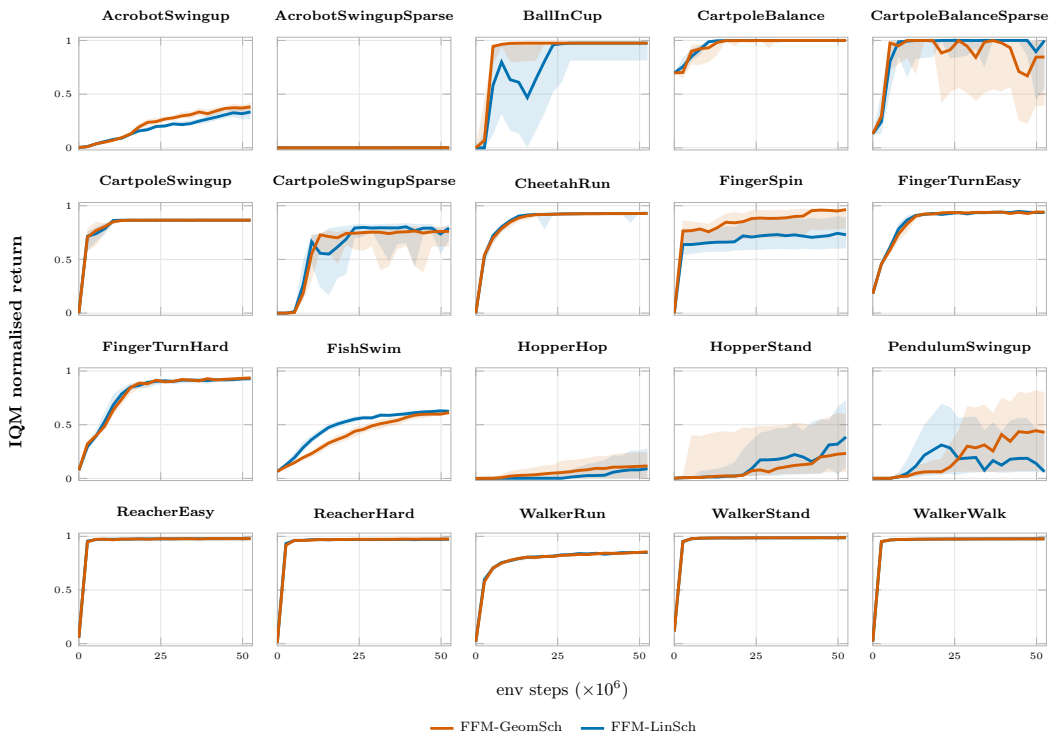


Figure 7: **Ablation. Geometric versus linear interpolant schedule (MJX Playground).** We plot the IQM per-task learning curves with 95% stratified bootstrapped confidence intervals (Agarwal et al., 2021) over 10 seeds on the 20 MJX Playground tasks. The performance between the schedule types is the same for most tasks and differs only for some tasks such as *FingerSpin* or *Pendulum-Swingup*, which, however, lie within the confidence intervals. Therefore, there is no statistically clear result on which interpolant should be used for the Playground tasks.

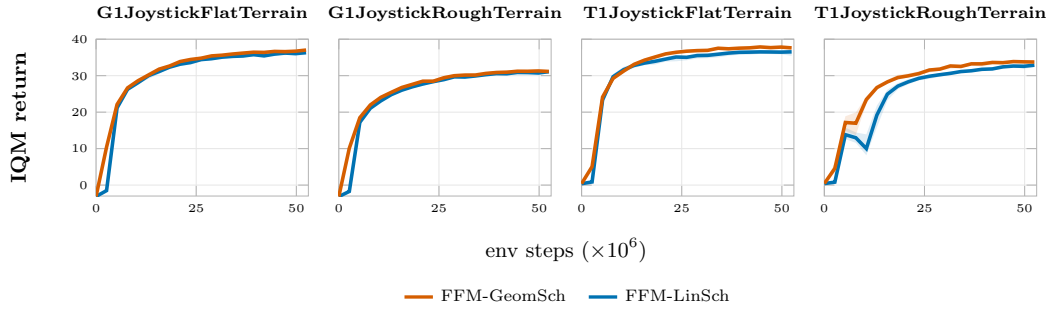


Figure 8: **Ablation. Geometric versus linear interpolant schedule (MJX Humanoid).** We plot the IQM per-task learning curves with 95% stratified bootstrapped confidence intervals (Agarwal et al., 2021) over 10 seeds on the four MJX humanoid tasks. While for the *G1JoystickFlatTerrain* and for the *G1JoystickRoughTerrain* tasks there is no difference in performance, it seems that the geometric schedule performs slightly better on the *T1* tasks, which is why we have chosen the geometric schedule as default for the online RL experiments.

$K = 16$ throughout the paper for a fair comparison against the other diffusion- and flow-based methods (Table 1).

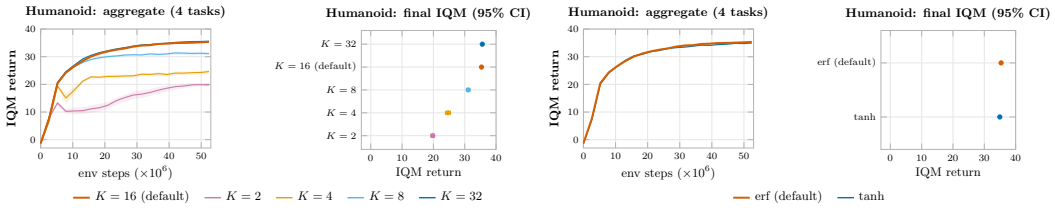


Figure 9: **Ablation. Number of Integration Steps (left two panels) and Action Squashing mapping (right two panels).** *Number of integration steps (left two panels):* We analyze the effect of varying integration steps and report the aggregated results over all four humanoid tasks (left two panels). Every single task was run on 10 seeds, and we report the IQM and the 95% stratified bootstrapped confidence intervals (Agarwal et al., 2021). The results indicate that with an increasing number of integration steps, the performance increases until it plateaus from $K = 16$ onwards, which is why we have chosen $K = 16$ as default in our experiments. Please see Figure 10 for the per-task learning curves. *Action Squashing Mapping (right two panels):* We analyze the effect of utilizing the $\tanh$ and the erf as squashing functions for mapping the generated actions to a defined action bound. We have run experiments on the four MJX humanoid tasks and report the aggregated IQM with 95% stratified bootstrapped confidence intervals (10 seeds per task) (Agarwal et al., 2021) in the two right panels. The results show that FFM is not sensitive to the choice of the squashing function. Please see Figure 11 for the per-task learning curves.

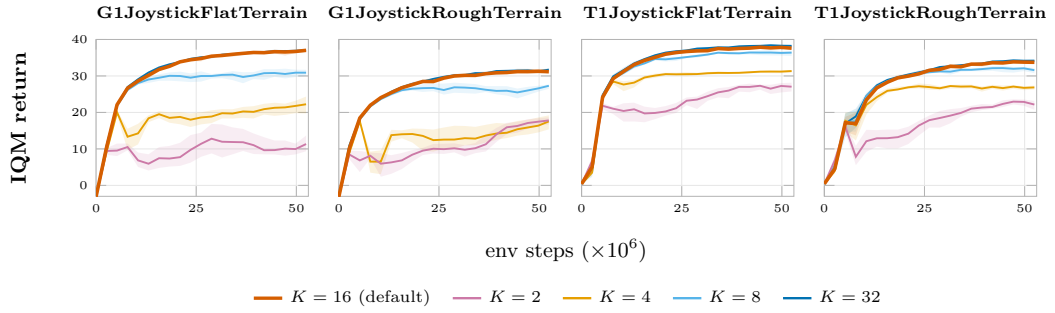


Figure 10: **Ablation. Number of Integration Steps.** We plot the IQM per-task learning curves with 95% stratified bootstrapped confidence intervals (Agarwal et al., 2021) over 10 seeds on the four MJX humanoid tasks for varying numbers of integration steps K. With increasing number of integration steps, the performance improves. However, the performance plateaus at $K = 16$ steps.

Action squashing. A key factor for stable training in actor-critic-based RL is that the generated actions need to be squashed to a valid region to avoid querying the critic at unreasonable action regions during the policy update. As detailed in Section D, there are two distinctive choices for the squashing function, namely, the $\tanh$ and the erf functions. To assess the effect on the performance of the design choices, we have run FFM with a $\tanh$ squashing function while keeping all other design choices the same and report the aggregated learning curves on the humanoid tasks in Figure 9(right), together with the final returns. The per-task results are shown in Figure 11. From the results, it becomes clear that both variants perform on par, indicating that FFM is not sensitive to this choice and that either function is a valid default.

Deterministic evaluation. During our experiments, we observed that starting the flow's integration always at the central point $x_0 = 0$ during evaluation only, yields improved performance against the standard case, where $x_0 \sim \mathcal{N}(0, I)$ is drawn from a standard normal distribution as done during training. This observation is depicted in Figure 12, where we plot the aggregated learning curves on the Playground tasks together with their final returns (left) and the aggregated learning curves on

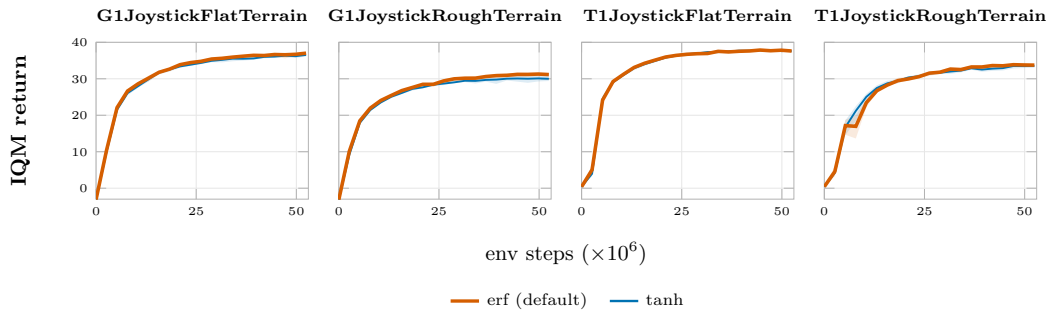


Figure 11: **Ablation. Action squashing mapping.** We plot the IQM per-task learning curves with 95% stratified bootstrapped confidence intervals (Agarwal et al., 2021) over 10 seeds on the four MJX humanoid tasks for the tanh and erf squashing functions. The results indicate that FFM is insensitive to the choice of the squashing function.

the humanoid tasks together with their final returns (right). The per-task learning curves are shown in Figure 13 and Figure 14 respectively. While the improvement is not very obvious for the Playground tasks (Figure 12 (left)), a clear and significant performance boost in the humanoid tasks can be observed (Figure 12 (right)). This result indicates that, for high-dimensional tasks in particular, removing the stochasticity from the action generation process yields improved performance. While this observation is very interesting and worth a more detailed analysis, in particular from a theoretical point of view, we leave this analysis for future work and apply the deterministic evaluation throughout our experiments.

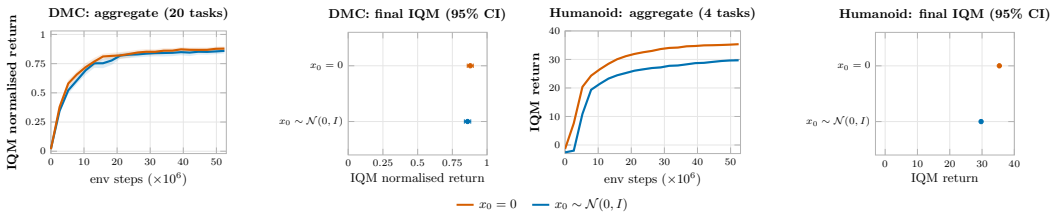


Figure 12: **Ablation. Deterministic evaluation.** We compare the effect of evaluating FFM using $x_0 = 0$ (deterministic) and $x_0 \sim \mathcal{N}(0, I)$ (stochastic) on the 20 MJX Playground (left two panels) and four humanoid tasks (right two panels), where every task was run on 10 seeds, and we plot the aggregated IQM with stratified bootstrapped 95% confidence intervals (Agarwal et al., 2021). While there is almost no performance gap between both variants for the Playground tasks, a significant performance boost follows from the deterministic evaluation on the humanoid tasks. This indicates that deterministic evaluation matters most for high-dimensional tasks. Please see Figure 13 and Figure 14 for the per-task performances of each task suite.

Correlated exploration. A flow policy's key benefit is that the action generation process is a deterministic mapping up to the sampling of the starting sample. As discussed in Section 5, this allows us to apply a correlated exploration scheme, where the exploration is shaped by correlating the prior draws across the environment steps. To analyze the additional hyperparameter τ_{ou} that is introduced by Equation (20), we have conducted experiments on single Playground tasks, where the initial instantiation of FFM, i.e., without correlated exploration, performs unsatisfactorily (see Figure 4). Figure 15 shows the learning curves on a subset of tasks that we have picked from the task suite to analyze the effect on performance for varying τ_{ou} values. We can observe that for each task, there exists at least one τ_{ou} value that outperforms the initial case without correlated exploration, and only for the *FingerSpin* and *WalkerRun* tasks, we observe that a too-high value yields decreased performance. Especially for the *AcrobotSwingupSparse, HopperHop* and *HopperStand* tasks, correlated exploration yields a significant improvement over the case without correlated exploration. However, the results also indicate that there is no global value for τ_{ou} that performs best across every task, which makes tuning this hyperparameter difficult and is only possible if enough compute resources

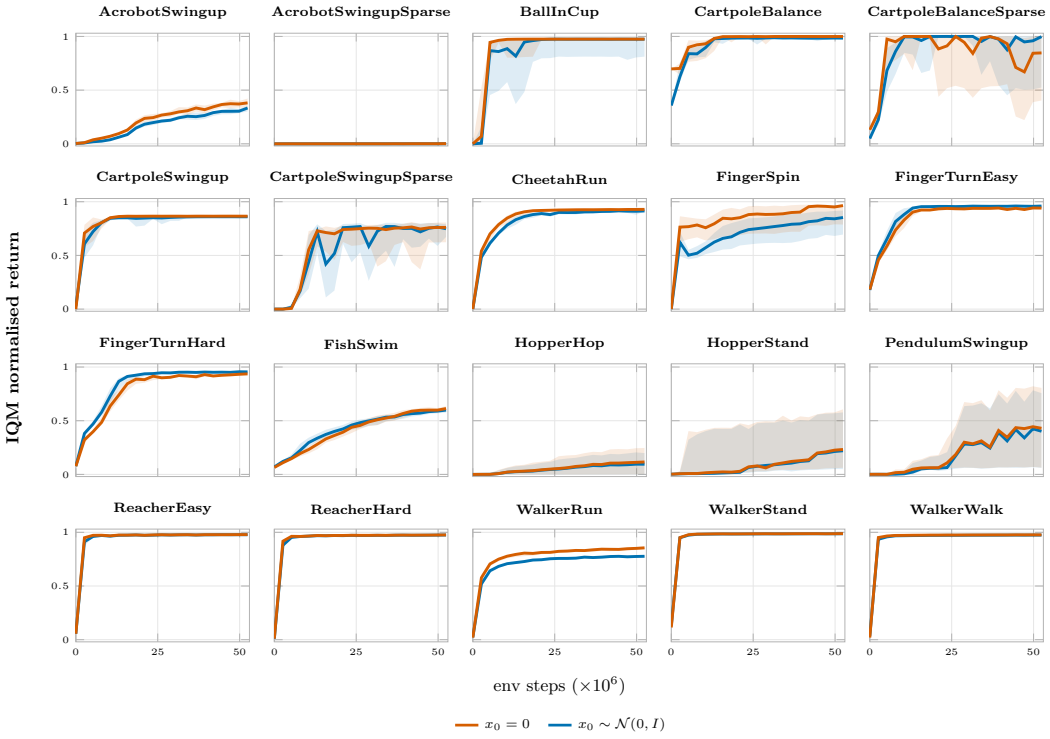


Figure 13: **Ablation. Deterministic evaluation (MJX Playground).** We plot the IQM per-task learning curves with 95% stratified bootstrapped confidence intervals (Agarwal et al., 2021) over 10 seeds on each of the 20 MJX Playground tasks for $x_0 = 0$ (deterministic) and $x_0 \sim \mathcal{N}(0, I)$ (stochastic) during evaluation. While most of the results indicate that there are not many differences, for *FingerSpin*, or *WalkerRun*, deterministic evaluation seems to improve performance.

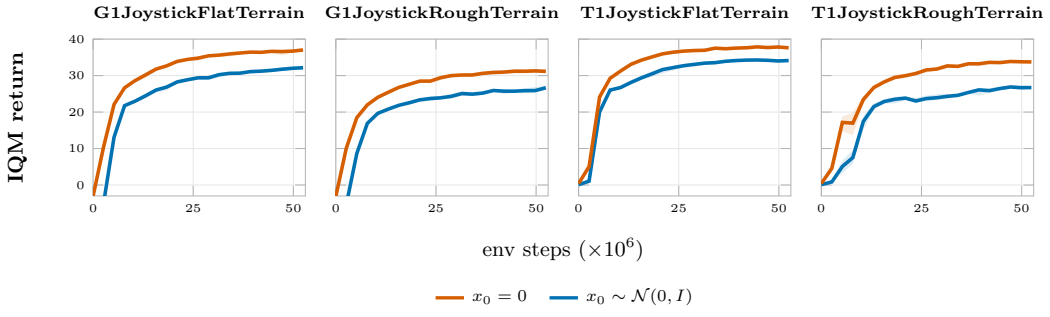


Figure 14: **Ablation. Deterministic evaluation (MJX Humanoids).** We plot the IQM per-task learning curves with 95% stratified bootstrapped confidence intervals (Agarwal et al., 2021) over 10 seeds on each of the four MJX humanoid tasks for $x_0 = 0$ (deterministic) and $x_0 \sim \mathcal{N}(0, I)$ (stochastic) during evaluation. For all four tasks, deterministic evaluation shows significantly improved performance, indicating that stochasticity can harm performance, in particular for high-dimensional action spaces. We use deterministic evaluation throughout all of our experiments.

are available. Nevertheless, we believe that correlated exploration is a useful feature that FFM can leverage.

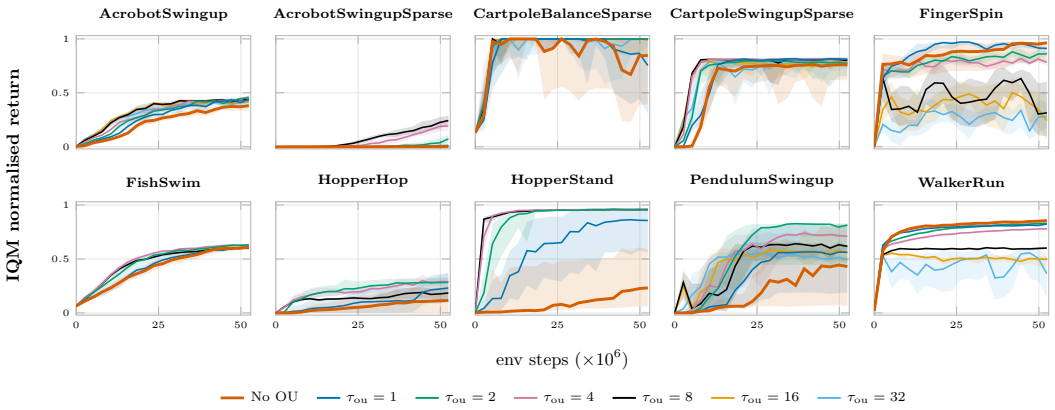


Figure 15: **Ablation. Correlated Exploration.** On a subset of the MJX Playground tasks, we analyze the effect of correlated exploration by varying $\tau_{\text{ou}} \in \{1, 2, 4, 8, 16, 32\}$, where we have run every curve on 10 seeds and report the IQM with 95% stratified bootstrapped confidence intervals (Agarwal et al., 2021). Correlated exploration can significantly boost FFM's performance for the right choice of τ_{ou}. For example, the *HopperStand* task is solved, while FFM without correlated exploration struggles on this task.

E Offline-to-online fine-tuning

This appendix collects the offline-to-online instantiation of FFM (Section 5): the behavior-constrained setting and the behavior-tilted target (Section E.1), the fine-tuning procedure (Section E.2), and the protocol, full results, and design ablations of the offline-to-online experiments of Section 6 (Sections E.3 to E.5). Throughout, v_{ref} and s_{ref} denote the velocity and score fields of the pretrained (behavior) policy p_{ref}, related by the bridge Equation (28), and $\kappa(t) = \frac{\dot\alpha(t)}{\alpha(t)}\beta^2(t) - \dot\beta(t)\beta(t)$ is the bridge coefficient Equation (8), assumed positive on $(0, 1)$.

E.1 Further details

In offline-to-online RL a fixed dataset $\mathcal{D}$ of transitions is available before any interaction. Two objects are pretrained on it: a *behavior flow* p_{ref}, fit by flow matching Equation (6) to the dataset actions (its velocity field v_{ref} is the reference of Section 5), and a critic Q, fit by an offline method (here pessimistic TD, Algorithm 2). Fine-tuning then continues both while the dataset grows with online interaction. The difficulty is that an offline critic is trustworthy only near the data: its extrapolation errors off the support of p_{ref} are large and systematically optimistic, so an unconstrained improvement step chases them (Peng et al., 2019; Nair et al., 2020; Park et al., 2025b). The standard remedy penalizes deviation from the behavior policy: at each state, maximize $\mathbb{E}_{x\sim\pi}[Q(s,x)] - \lambda\, D_{\text{KL}}(\pi(\cdot \mid s) \,\|\, p_{\text{ref}}(\cdot \mid s))$, whose maximizer is the *behavior-tilted* Boltzmann policy $\pi^\star(x \mid s) \propto p_{\text{ref}}(x \mid s)\, e^{Q(s,x)/\lambda}$ of Section 3 (Abdolmaleki et al., 2018; Peng et al., 2019; Nair et al., 2020). Its terminal score is

$$\nabla_x \log \pi^\star(x \mid s) \;=\; \mathsf{s}_{\text{ref}}(x) + \nabla_x Q(s,x)/\lambda, \qquad \mathsf{s}_{\text{ref}} \;:=\; \nabla_x \log p_{\text{ref}}(\cdot \mid s), \tag{58}$$

so relative to the pure Boltzmann target of Section 4 exactly one new ingredient enters: the reference's terminal score. For a flow-parameterized reference it is available without a density and without the dataset, by reading the score off the pretrained velocity field through the bridge Equation (29) near $t = 1$, which is Equation (13). Two mechanisms then keep the fine-tuned policy close to p_{ref}, and Algorithm 2 combines them: the *target* itself, which by Section A has the tilt as a fixed point, and the trust region of Section 5, here taken with respect to the pretrained flow, that is, $\mathcal{L}_{\text{TR}}(\theta, \text{ref}) \le \varepsilon_{\text{TR}}$ with v_{ref} in place of $v_{\bar\theta}$ in Equation (16), which bounds every step.

E.2 Pseudocode

Algorithm 2 summarizes the fine-tuning procedure. It alternates a pessimistic TD update of the critic ensemble with the behavior-tilted FFM step of Section 5: clean actions are generated by the current flow, labeled with the score g of the behavior-tilted target, noised, and regressed onto by Equation (14), while the trust region Equation (16), taken with respect to the pretrained flow v_{ref} instead of the previous policy, is enforced through the Lagrangian Equation (18) and the dual update of Equation (19). We write $\mathcal{L}_{\mathrm{TR}}(\theta, \mathrm{ref})$ for Equation (16) with v_{ref} in place of $v_{\bar{\theta}}$, $\bar{\theta}$ for a frozen copy of the current actor parameters, and $\bar{\phi}$ for the EMA target parameters of the critic. The implementation differs from the pseudocode in the form of the actor regression, which uses the variance-reduced target of Section B.2; this and the remaining implementation details are given in Section E.3.

Algorithm 2 Offline-to-online fine-tuning with an FFM actor

input: Pretrained velocity field v_{ref} with terminal law p_{ref}, critic ensemble $Q_{\phi_{1:N}}$ with EMA targets $Q_{\bar{\phi}_{1:N}}$, dataset $\mathcal{D}$, temperature λ, pessimism ϱ, trust-region budget $\varepsilon_{\mathrm{TR}}$ with multiplier ν and dual step size η_ν, schedules α, β
initialization: warm start $v_\theta \leftarrow v_{\mathrm{ref}}$, set $\nu \leftarrow \nu_0$
for each step, first offline on $\mathcal{D}$, then online **do**
  interaction (online phase only, no gradients):
    • act $x \sim \pi_\theta(\cdot \mid s)$ by integrating Equation (5)
    • add (s, x, r, s') to $\mathcal{D}$
  policy evaluation: pessimistic TD targets on a minibatch of $\mathcal{D}$ with $x' \sim \pi_\theta(\cdot \mid s')$ (target gradients stopped)
    $y = r + \gamma \left(\mathrm{mean}_j\, Q_{\bar{\phi}_j}(s', x') - \varrho\, \mathrm{std}_j\, Q_{\bar{\phi}_j}(s', x')\right)$
  update critic parameters $\phi_{1:N}$ using $\sum_j \big(Q_{\phi_j}(s, x) - y\big)^2$, then the EMA targets $\bar{\phi}_{1:N}$
  calculate scores (no gradients):
    • clean actions $X_1 \sim \pi_{\bar{\theta}}(\cdot \mid s)$ by integrating Equation (5)
    • reference score $\mathsf{s}_{\mathrm{ref}}(X_1)$ read off v_{ref} by Equation (13)
    • behavior-tilted score $g(X_1) = \mathsf{s}_{\mathrm{ref}}(X_1) + \frac{1}{\lambda}\nabla_x Q(s, X_1)$
  $t \sim \mathcal{U}(0,1)$, $\varepsilon \sim \mathcal{N}(\cdot;\, 0,\, I)$, $X_t \leftarrow \alpha(t) X_1 + \beta(t)\varepsilon$
  minimize over θ using Equation (18) with $\mathcal{L}_{\mathrm{BT-FFM}}$ of Equation (14):
    $\mathbb{E}_{t, X_1, \varepsilon}\left[\left\| v_\theta(X_t, t \mid s) - \frac{\dot{\alpha}(t)}{\alpha(t)} X_t - \frac{\kappa(t)}{\alpha(t)}\, g(X_1) \right\|^2\right] + \nu\big(\mathcal{L}_{\mathrm{TR}}(\theta, \mathrm{ref}) - \varepsilon_{\mathrm{TR}}\big)$
    with $\mathcal{L}_{\mathrm{TR}}(\theta, \mathrm{ref})$ of Equation (16) evaluated at the same points (X_t, t)
  dual: ascend $\nu \leftarrow \max\big(0,\, \nu + \eta_\nu\, (\mathcal{L}_{\mathrm{TR}}(\theta, \mathrm{ref}) - \varepsilon_{\mathrm{TR}})\big)$ (Equation (19))
  update the frozen actor copy: $\bar{\theta} \leftarrow \theta$
return fine-tuned velocity field v_θ

E.3 Experimental details

E.3.1 Environments and tasks

We evaluate on the offline-to-online benchmark of OGBench (Park et al., 2025a), comprising ten domains that split into *locomotion* (goal-reaching navigation) and *manipulation* (goal-conditioned object rearrangement); Figure 16 shows a representative frame of each. Every domain defines five *singletask* goals (Task 1–5): the offline dataset is shared across a domain's tasks (goal-free `navigate`/`play` data) and each task fixes a goal used for the sparse reward and for evaluation. Table 2 lists the state and action dimensionality, episode length, dataset size, and network width for each domain.

E.3.2 Offline-to-online protocol

We follow the protocol of TRQAM. For every (environment, task, seed) we first pretrain a behavior-cloning flow policy by flow matching for 300K gradient steps, then run 1M offline gradient steps followed by 500K online environment steps (1.5M total); the online phase continues from the offline

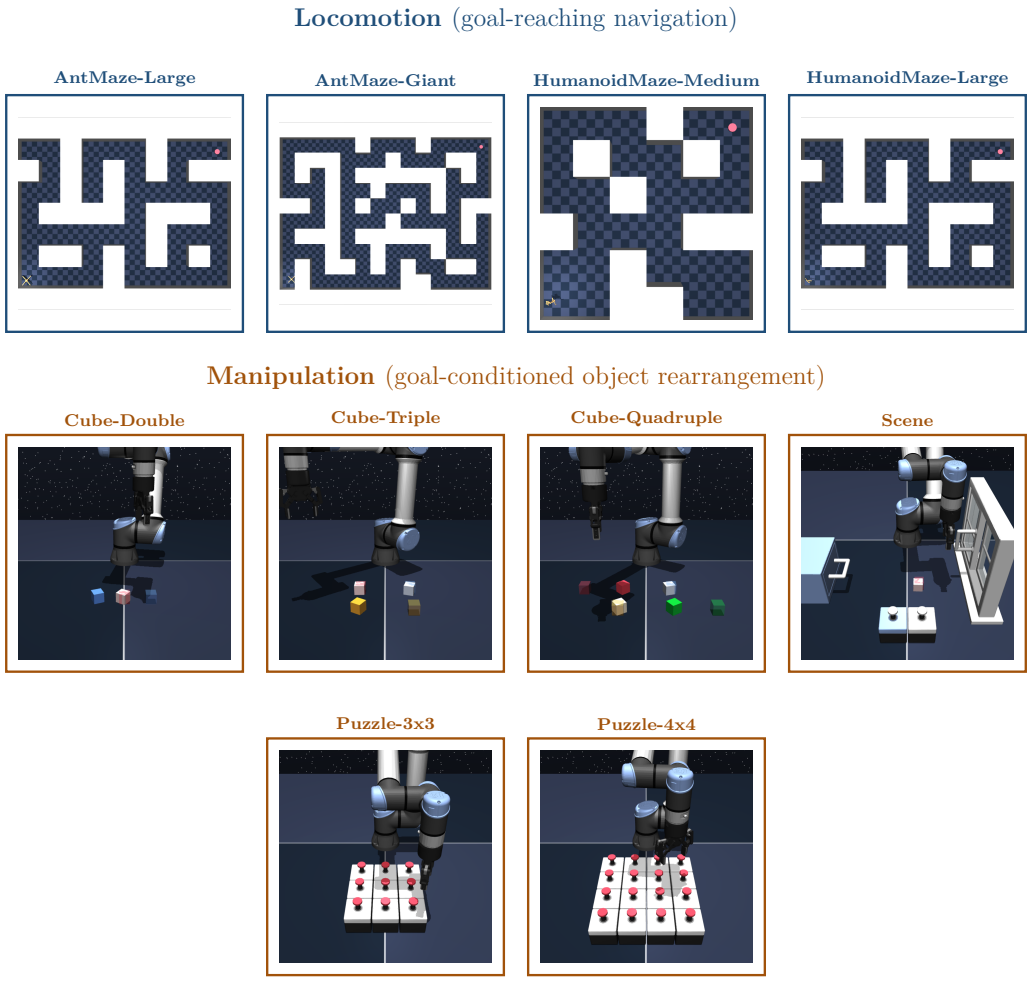


Figure 16: The ten evaluation environments (five tasks each). Top: locomotion; bottom: manipulation.

Table 2: Environment specifications. Network width and layer norm follow the dataset-size convention (1M: width 512; 10M/100M: width 1024 with actor layer norm), applied identically to the BC-pretraining and fine-tuning networks.

Environment	Type	State	Act	Ep. len	Dataset	Width
AntMaze-Large	locomotion	29	8	1000	1M	512
AntMaze-Giant	locomotion	29	8	1000	10M	1024
HumanoidMaze-Medium	locomotion	69	21	2000	1M	512
HumanoidMaze-Large	locomotion	69	21	2000	1M	512
Cube-Double	manipulation	37	5	500	1M	512
Cube-Triple	manipulation	46	5	1000	10M	1024
Cube-Quadruple	manipulation	55	5	1000	100M	1024
Scene	manipulation	40	5	750	1M	512
Puzzle-3x3	manipulation	55	5	500	1M	512
Puzzle-4x4	manipulation	83	5	500	10M	1024

actor, critic, optimizer, and dual variables. Policies are evaluated every 50K steps over 50 episodes and we report the binary task-success rate. We use the eight-seed set $\{10001, 20002, \ldots, 80008\}$.

Table 3: Shared hyperparameters (all methods).

Optimizer	Adam, learning rate 3×10^{-4}
Batch size	256
MLP	4 hidden layers, width 512 (1M) / 1024 (10M–100M)
Layer norm	critic: yes; actor: 1024-width domains only
Discount γ	0.995 (HumanoidMaze: 0.999)
Pessimism ρ	0.5 (HumanoidMaze: 0.0)
Target EMA rate	0.005
Flow integration steps	10
Action chunk length (manipulation)	5

Table 4: FFM two-regime recipe: the only per-domain choice.

	Regime A (locomotion)	Regime B (manipulation)
Critic scale τ_Q	30	1
Fisher normalizer	fixed ($i_{\mathrm{F}} = 1$)	adaptive (EMA)
Behavior anchor	trust region ($\lambda_{\mathrm{scale}} = 1$)	off ($\lambda_{\mathrm{scale}} = 0$, pure product)
Action chunking	no	yes ($h = 5$)

E.3.3 Shared architecture and hyperparameters

All methods use four-hidden-layer MLP actors and critics; the critic is an ensemble with pessimistic backup. The discount and pessimism are $(\gamma, \rho) = (0.995, 0.5)$ except on the long-horizon HumanoidMaze domains, where $(\gamma, \rho) = (0.999, 0.0)$. Table 3 lists the shared optimization hyperparameters.

E.3.4 FFM hyperparameters

FFM exposes a single per-domain choice, the temperature/normalizer *regime*, selected by a simple rule: chunked manipulation uses Regime B, sparse-reward locomotion uses Regime A. This choice transferred across all ten domains with no per-task or per-seed tuning. Shared across regimes: the terminal reference score is used, bridge offset $\delta_{\mathrm{ref}} = 0.1$, endpoint softening $t_\epsilon = 0.05$, four interpolant samples per state, and a phase-dependent behavior anchor with budget $\varepsilon_{\mathrm{TR}} = 1.0$ offline and 8.0 online. Table 4 contrasts the two regimes.

Implementation of the actor update. The implementation differs from the pseudocode of Algorithm 2 in the form of the actor regression. Instead of regressing onto the pure FFM target of Equation (14), we regress onto the mixed target Y_c of Equation (34), which combines it with the flow-matching target under the variance-optimal schedule $c^\star(t)$ of Equation (37), with the score $g = \mathsf{s}_{\mathrm{ref}} + \nabla_x Q/\lambda$ of the behavior-tilted target in place of $\nabla_x Q/\lambda$; the Fisher scale $\bar{\mathcal{I}}$ is either held fixed or tracked by an EMA (the normalizer row of Table 4). The residual is divided by $\kappa(t)$, so that the regression is carried out in score space (Section B.2). All offline-to-online experiments use the linear interpolant $\alpha(t) = t$, $\beta(t) = 1 - t$, for which $\dot\alpha(t)/\alpha(t) = 1/t$ and $\kappa(t) = (1-t)/t$, and the regression line then reads

$$\frac{v_\theta(X_t, t) - Y_c}{\kappa(t)} = \frac{t\big(v_\theta(X_t, t) - X_1\big) + (t - c)\,\varepsilon}{1 - t} - \frac{c}{t}\, g(X_1). \tag{59}$$

The denominators $1 - t$ and t are softened by the endpoint constant t_ϵ listed above, which replaces a time truncation (Section B.2), and the trust region Equation (16) is evaluated at the same points (X_t, t) with the weight $1/\kappa(t)$ under the same softening. Further details: the reference score Equation (13) is read off the pretrained flow at $t = 1 - \delta_{\mathrm{ref}}$; the clean actions X_1 are clipped to the action box before the critic gradient is taken; four interpolant samples (t, ε) are drawn per clean action; the critic target uses the best of n sampled next actions; and the pretrained flow v_{ref} continues to be trained by flow matching Equation (6) on the growing dataset during fine-tuning, so that the reference tracks the data.

Why two regimes. The four rows of Table 4 are not four independent knobs but two coordinated operating points, set by how informative the critic's action gradient is in each domain family.

Critic scale and Fisher normalizer are a matched pair. FFM pulls the flow toward the product tilt $p_{\text{ref}}(a)\, e^{\tau_Q Q(o,a)}$, whose score $s_{\text{ref}} + \tau_Q \nabla_a Q$ drives the flow through the inverse-SNR schedule $c(t) = t^2/\big(t^2 + (1-t)^2 I\big)$. In the sparse-reward mazes the reward is rarely seen and Q is smooth over a large state space, so $\nabla_a Q$ is small. We amplify it with $\tau_Q = 30$ which survives only if the normalizer I is held *fixed* ($i_{\text{F}} = 1$): the adaptive normalizer estimates $\|\tau_Q \nabla_a Q\|^2$ and divides it back out, cancelling exactly the gain we introduced. In manipulation the chunked critic gradient is already informative, so $\tau_Q = 1$ suffices and there is no gain to preserve; there the *adaptive* (EMA) normalizer is preferable, because a multi-object action chunk has gradient magnitudes that vary by orders of magnitude across coordinates and over training, and the EMA renders the update scale-invariant. Hence (τ_Q=30, fixed) and (τ_Q=1, adaptive) are the only two sensible combinations: mixing them either normalizes away the amplification or leaves the step uncalibrated.

The behavior anchor differs because off-support risk does. The terminal reference score is active in both regimes, so the target of the iteration is anchored to the pretrained flow by construction. Locomotion additionally retains the explicit path-space trust region ($\lambda_{\text{scale}} = 1$): with sparse rewards the critic is unreliable off the visited data and an untethered navigation policy can drift into unseen regions of the maze, so the dual-controlled trust region actively pulls it back. In manipulation the product tilt together with chunking already keep the actor on the data manifold, and the external trust region then over-constrains it and slows online improvement, so we drop it ($\lambda_{\text{scale}} = 0$, pure product).

Action chunking is inherited, not tuned. Q-chunking with horizon $h = 5$ is part of the benchmark protocol we adopt from TRQAM (which in turn follows QAM): the chunked critic backup and full-chunk execution live in the shared training loop and are applied identically to every method (Table 3). It is enabled on the manipulation domains and disabled on the locomotion domains for *all* methods, exactly as in TRQAM; we do not select it per method. It appears as a row of Table 4 only because it is the defining feature of the manipulation family and so coincides with the regime split, not because FFM tunes it. The genuinely FFM-specific choices are the critic scale, the Fisher normalizer, and the behavior anchor above.

E.3.5 Baselines

We compare against TRQAM, QAM (Li & Levine, 2026), QAM-E, IFQL, FQL (Park et al., 2025b), DSRL, and CGQL-L, each run in the identical protocol above.

E.4 Full results

Tables 5 and 6 report the offline-phase (at 1M steps) and offline-to-online (at 1.5M steps) success rates, aggregated per domain over the five tasks and all seeds, with the best method per row in bold. The per-domain aggregated learning curves are shown in Figure 2 (main text), and Figure 17 gives the full per-(environment, task) grid; in both, the offline ($\leq$1M) and online ($>$1M) phases are shaded. The tables corroborate the discussion of Section 6: FFM's offline average trails the strongest offline baseline yet its offline-to-online average is the best overall, with the largest per-environment gains on the hard, long-horizon tasks and no divergence across seeds.

E.5 Design ablations

The two-regime recipe of Table 4 and the integration budget were fixed once and transferred across all ten domains without per-task tuning. This section reports the offline ablations that justify those choices. Each ablation isolates a single axis on one representative task per family (manipulation on `cube-double` task 3 and locomotion on `antmaze-large` task 3), trained offline only (1M steps, reusing the pretrained flow) with every other setting held at its regime value from Table 4. We report offline success averaged over the final 150K steps (evaluations at $\geq$ 850K) as mean $\pm$ standard deviation over three seeds.

The critic scale and Fisher normalizer form a two-regime crossover. The two *critic-scale* panels of Figure 18 sweep $\tau_Q \in \{1, 2, 4, 8, 30\}$ against the two normalizer settings (fixed i_{F}=1 vs.

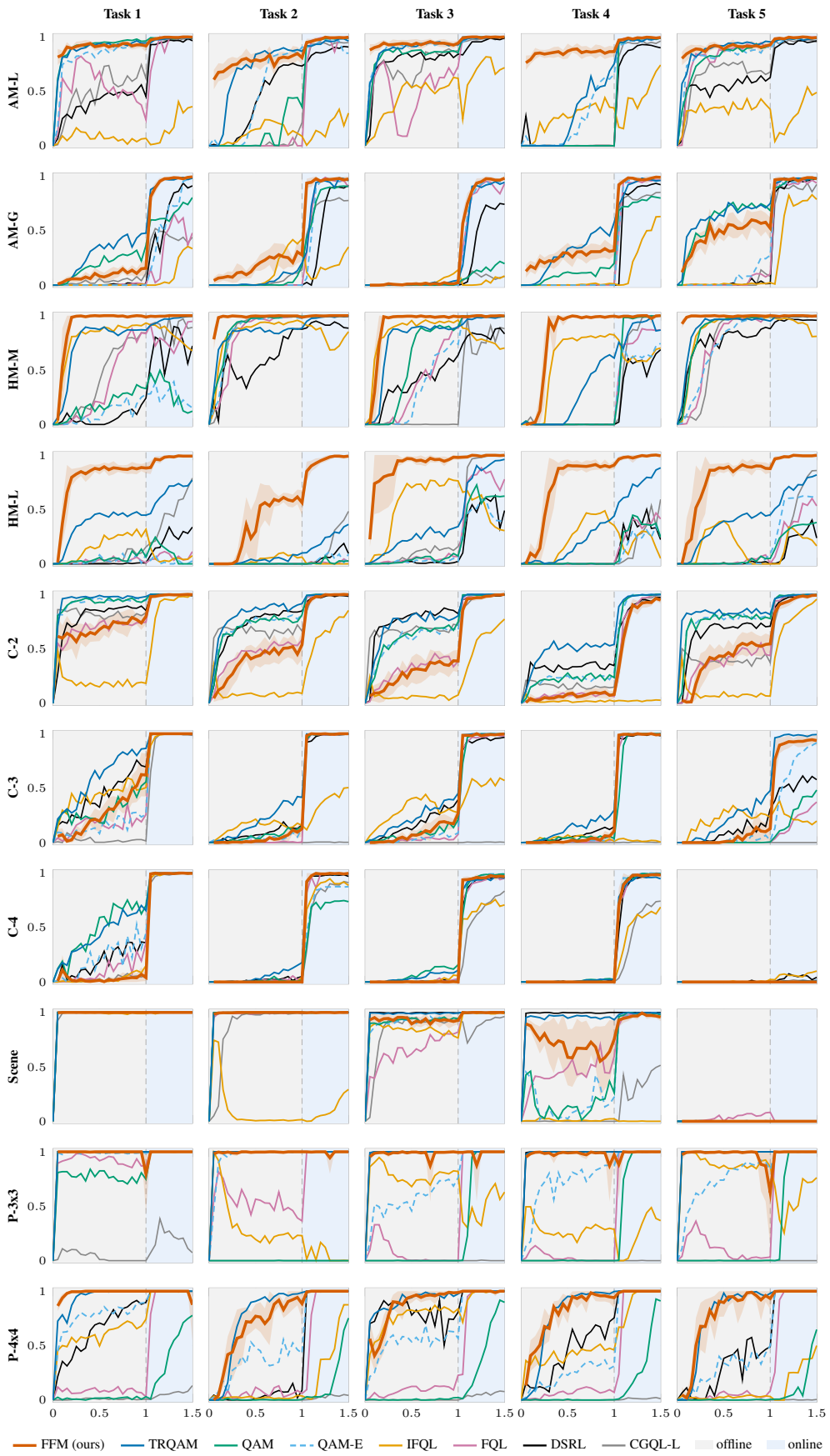


Figure 17: Offline-to-online success rate for every environment (rows) and task (columns), mean over seeds. Offline and online phases are shaded; FFM (ours) is drawn in bold.

Table 5: Offline-phase success rate at the end of offline training (1M steps), mean over 5 tasks and seeds per environment. Best per row in bold.

Environment	FFM	TRQAM	QAM	QAM-E	IFQL	FQL	DSRL	CGQL-L
AntMaze-Large	**0.89**	0.89	0.62	0.86	0.29	0.38	0.53	0.48
AntMaze-Giant	0.26	**0.41**	0.29	0.06	0.12	0.02	0.01	0.07
HumanoidMaze-Medium	**0.99**	0.84	0.64	0.59	0.93	0.74	0.53	0.57
HumanoidMaze-Large	**0.84**	0.36	0.04	0.04	0.30	0.02	0.01	0.06
Cube-Double	0.47	**0.81**	0.71	0.71	0.09	0.44	0.72	0.55
Cube-Triple	0.25	**0.50**	0.19	0.11	0.24	0.07	0.34	0.00
Cube-Quadruple	0.01	**0.19**	0.18	0.09	0.06	0.09	0.09	0.01
Scene	0.74	0.79	0.64	0.63	0.36	0.70	**0.80**	0.58
Puzzle-3x3	0.88	1.00	0.15	0.89	0.64	0.25	**1.00**	0.00
Puzzle-4x4	0.95	**0.99**	0.01	0.54	0.42	0.09	0.61	0.00
Average	0.63	**0.68**	0.35	0.45	0.34	0.28	0.46	0.23

Table 6: Offline-to-online success rate at the end of online fine-tuning (1.5M total steps), mean over 5 tasks and seeds per environment. Best per row in bold.

Environment	FFM	TRQAM	QAM	QAM-E	IFQL	FQL	DSRL	CGQL-L
AntMaze-Large	1.00	0.98	0.99	0.96	0.52	**1.00**	0.95	0.97
AntMaze-Giant	**0.98**	0.97	0.74	0.97	0.44	0.84	0.90	0.61
HumanoidMaze-Medium	**1.00**	0.97	0.82	0.78	0.75	0.96	0.82	0.95
HumanoidMaze-Large	**0.99**	0.76	0.26	0.27	0.12	0.37	0.28	0.74
Cube-Double	0.99	1.00	**1.00**	1.00	0.72	1.00	0.99	0.99
Cube-Triple	0.99	**1.00**	0.90	0.98	0.46	0.87	0.90	0.20
Cube-Quadruple	0.79	0.78	0.74	0.77	0.68	0.79	**0.79**	0.70
Scene	0.79	0.80	0.80	**0.80**	0.46	0.80	**0.80**	0.70
Puzzle-3x3	**1.00**	**1.00**	0.80	**1.00**	0.55	**1.00**	**1.00**	0.01
Puzzle-4x4	0.97	**1.00**	0.80	**1.00**	0.88	1.00	**1.00**	0.05
Average	**0.95**	0.93	0.78	0.85	0.56	0.86	0.84	0.59

adaptive EMA), turning the matched-pair argument above into an empirical statement. On manipulation success falls monotonically as τ_Q grows (from 0.40 at τ_Q=1 to 0.00 at τ_Q=30), and the two normalizers are indistinguishable, because at τ_Q=1 there is no amplification for the normalizer to cancel. On locomotion the two curves *cross*: with the fixed normalizer success *rises* with the critic scale ($0.83 \to 0.95$), while with the adaptive normalizer it *falls* ($0.72 \to 0.14$), as the EMA estimate divides out exactly the gain that τ_Q introduces. The amplify pair (τ_Q=30, fixed) is thus necessary for the sparse-reward mazes and fatal on manipulation, while the calm pair (τ_Q=1, adaptive) is what manipulation requires: the two operating points of Table 4, with no mixed combination improving on either.

One normalizer suffices: unit-mean adaptivity restores τ_Q as a signal knob. The fixed/adaptive split exists only because the (scalar) adaptive normalizer estimates $\|\tau_Q \nabla_a Q\|^2$ and divides the amplification back out, so raising τ_Q cancels itself; this is the adaptive curve that collapses on locomotion in Figure 18. A cheaper fix keeps the adaptivity but not the cancellation: track the Fisher normalizer *per coordinate* and rescale it to unit mean, retaining its per-coordinate shape (the scale-invariance across action dimensions) while discarding the overall τ_Q^2 level. The *adaptive-shape* curve, the third line in the critic-scale panels of Figure 18, shows the effect: on locomotion it rises with τ_Q and tracks the fixed normalizer ($0.82 \to 0.93$ vs. $0.83 \to 0.95$) instead of collapsing like the scalar adaptive one ($0.72 \to 0.14$); on manipulation all three coincide. A single adaptive-shape normalizer thus spans both regimes, which would collapse the Fisher-normalizer row of Table 4 and leave τ_Q (and chunking) as the only per-domain choices. Since it *matches* rather than improves on the per-domain normalizer here (within seed noise on both families), we keep the explicit fixed/adaptive split in the shipped recipe for transparency, and report adaptive-shape as an equivalent single-knob alternative.

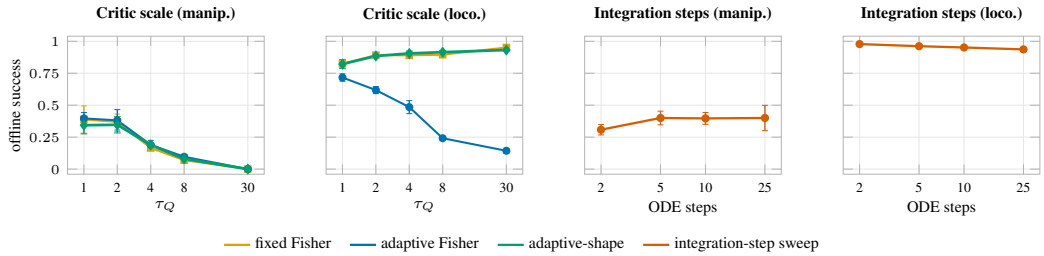


Figure 18: **Per-domain recipe knobs** (offline success, mean ± std over three seeds; a manipulation and a locomotion task). *Critic scale* (two left panels): the fixed, adaptive, and adaptive-shape (per-coordinate, unit-mean) Fisher normalizers vs. τ_Q. Manipulation needs a small τ_Q (normalizer immaterial); locomotion needs a large τ_Q with a normalizer that preserves the amplification: fixed and adaptive-shape both rise, while the scalar adaptive one cancels it and collapses (the matched-pair basis for Table 4; a single adaptive-shape normalizer spans both regimes). *Integration steps* (two right panels): manipulation needs at least five Euler steps, locomotion is insensitive, and the inherited default of ten lies on the plateau for both.

Ten integration steps is a safe, robust default. FFM integrates the sampling ODE with a fixed number of Euler steps, inherited as 10 from the TRQAM/QAM protocol. The two *integration-step* panels of Figure 18 vary this budget over $\{2, 5, 10, 25\}$. On manipulation two steps is visibly too coarse (0.31) but performance saturates by five steps and is flat thereafter (≈ 0.40); on locomotion the metric is nearly insensitive, with a slight preference for *fewer* steps (0.98 at two vs. 0.94 at twenty-five). Ten steps sits on the plateau for both families, so the inherited value is robust rather than tuned; five would trade no measurable accuracy for half the sampling cost.

The behavior anchor: reference score versus trust region, offline and online. FFM is kept on the data manifold by either of two anchors: the terminal reference score, which makes the *product* tilt $p_{\text{ref}}(a)\, e^{\tau_Q Q}$ the target of the iteration, or the explicit path-space trust region. Dropping the reference score leaves the *pure* tilt $e^{\tau_Q Q}$. Figure 19 crosses the two anchors (product vs. pure tilt $\times$ trust region off/on) across both phases and both families.

Offline, the trust region is the stronger anchor on manipulation. Removing both anchors collapses the policy (0.00): a pure tilt with no anchor chases the critic off the data support. Restoring either recovers it, and on `cube-double` the trust region *alone* (pure tilt + trust region, 0.58) beats the product-tilt reference score (0.41), while stacking both is redundant, so the two anchors are substitutes and the trust region is at least as strong. On locomotion an anchor is likewise essential (pure, no anchor $\to$ 0.19) and the product tilt with the trust region is best (0.96).

Online, the picture reverses, and the budget the pure tilt needs is task-dependent. At the shipped online budget ($\varepsilon_{\text{TR}} = 8$) the pure tilt *collapses* on `cube-double` (0.05) while the product tilt reaches 0.99; on locomotion every cell converges to ≈ 1. The online budget sweep over four tasks (right two panels of Figure 20) shows that this is *not* a simple "pure tilt needs a tighter budget" story. Having no reference anchor, the pure tilt depends entirely on the trust region, but the budget it requires varies across tasks, and in *opposite* directions: `cube-double` needs a *tight* budget (1.0 at $\varepsilon_{\text{TR}} = 1$, collapsing to 0.05 by 8), whereas `puzzle-3x3` needs a *loose* one (0.23 at 1, rising to 1.0 by 8). The curves cross, so *no single online budget makes the pure tilt work on every task*: at each fixed ε_{TR} the worst task scores ≤ 0.42.

Why the shipped recipe keeps the product tilt. The product tilt, anchored by the reference score, is *budget-robust*: across all four tasks it stays ≥ 0.99 for every $\varepsilon_{\text{TR}} \geq 4$ (the shipped 8 included), and online the trust region neither helps nor hurts it. It is therefore the only anchor that admits a *single, task-independent* online budget. The pure tilt is offline-competitive, indeed stronger on `cube-double` (0.58 vs. 0.41), but that advantage does not translate into a usable recipe, because its online budget cannot be fixed across tasks. This is the concrete reason the recipe uses the product tilt.

Offline, neither anchor is budget-robust. Running the same sweep on the offline phase (left two panels of Figure 20) reproduces the pure tilt's task-dependence: `cube-double` and `scene` peak

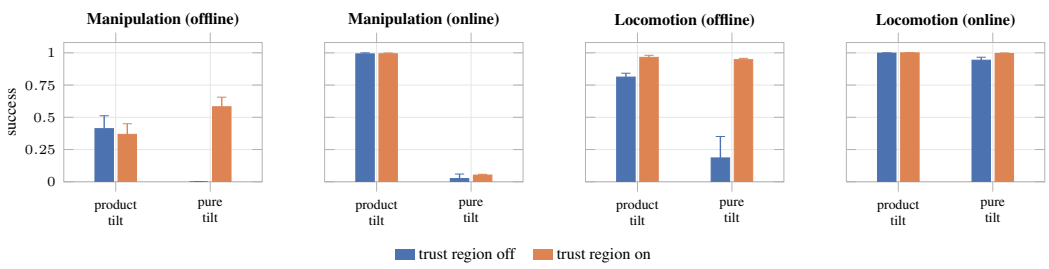


Figure 19: **Behavior anchor across phases** (product vs. pure tilt × trust region; mean ± std over three seeds). Product tilt $= p_{\text{ref}}(a)\, e^{\tau_Q Q}$; pure tilt $= e^{\tau_Q Q}$. Offline success (850K–1M) and online success (last 150K), for a manipulation and a locomotion task. Offline, pure tilt + trust region is the strongest manipulation anchor and the unanchored pure tilt collapses; online (at the shipped $\varepsilon_{\text{TR}} = 8$) the pure tilt collapses on manipulation while the product tilt reaches ≈ 1; locomotion converges regardless.

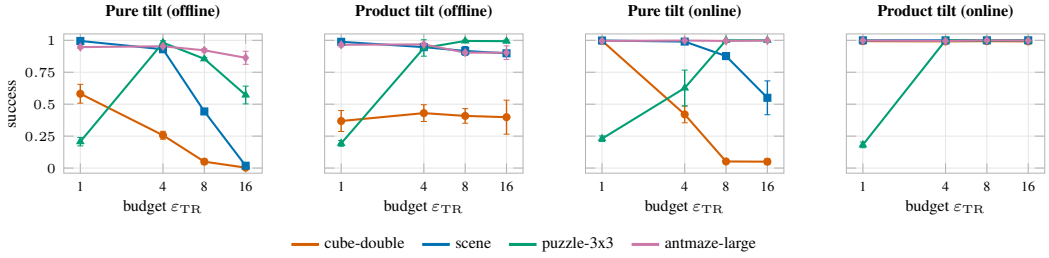


Figure 20: **Trust-region budget sweep across four tasks, both phases.** Success vs. the trust-region budget ε_{TR}; mean ± std over three seeds. *Left two panels:* offline success (850K–1M) against the offline budget. *Right two panels:* online success (last 150K) against the online budget. *Pure tilt (first and third panels):* in *both* phases the budget the pure tilt requires is task-dependent and points in opposite directions, `cube-double` and `scene` needing a tight budget and `puzzle-3x3` a loose one, so the curves cross and no single ε_{TR} works everywhere. *Product tilt (second and fourth panels):* reference-anchored and budget-robust online (≥ 0.99 on all four tasks for every $\varepsilon_{\text{TR}} \geq 4$), but *not* offline, where it is flat on three tasks yet rises from 0.19 to 0.99 on `puzzle-3x3`. With the offline trust region off the product tilt scores 0.41, 0.94, 1.00 and 0.81 on `cube-double`, `scene`, `puzzle-3x3` and `antmaze-large`, so for it a loose offline budget is equivalent to no trust region at all.

at the tightest budget (0.58 and 1.00 at $\varepsilon_{\text{TR}} = 1$, falling to 0.00 and 0.02 by 16), `puzzle-3x3` again runs the other way (0.21 at 1, 0.98 at 4), and `antmaze-large` is insensitive. The product tilt, however, is *not* budget-robust offline the way it is online: it is flat on three tasks but swings from 0.19 at $\varepsilon_{\text{TR}} = 1$ to 0.99 at 8 on `puzzle-3x3`. For the product tilt a loose budget behaves like no trust region at all (0.99 at $\varepsilon_{\text{TR}} = 16$ versus 1.00 with the trust region off), so the shipped manipulation recipe, which disables the offline trust region, never meets this failure; the pathology is specific to imposing a *tight* offline budget. Extending the sweep down to $\varepsilon_{\text{TR}} = 0.25$ on `cube-double` places the pure tilt's offline optimum at $\varepsilon_{\text{TR}} = 1$ (0.35 at 0.25, 0.55 at 0.5, 0.58 at 1), so the offline curve is single-peaked and the tightening cannot usefully be pushed further.