GeMPO: Generalized Measure Matching for Online Diffusion Reinforcement Learning

Anonymous authors
Paper under double-blind review

Abstract

A commonly used family of RL algorithms for diffusion policies conducts softmax reweighting over samples from the behavior policy, which often induces an over-greedy policy and fails to utilize feedback from negative samples. In this work, we introduce GeMPO, a simple and unified framework that generalizes reweighting scheme in diffusion RL from softmax to general monotonic functions. GeMPO revisits diffusion RL via a measure matching perspective: *First*, we construct a virtual target policy measure via solving a regularized policy optimization objective; *Second*, we minimize the divergence between the current policy and this target measure through reweighted flow matching. This formulation offers two key advantages: **i)** It extends weight design beyond traditional exponential reweighting, allowing it to be tailored to diverse reward landscapes; and **ii)** by relaxing the non-negativity constraint on the target measure, our framework provides a principled justification for negative reweighting. We provide interpretations of how negative reweighting actively repels the policy from suboptimal actions and thus facilitates exploration. Extensive empirical evaluations demonstrate that GeMPO achieves competitive or superior performance by leveraging these flexible weighting schemes, and we provide practical guidelines for selecting reweighting methods in practice.

1 Introduction

Diffusion models (Ho et al., 2020; Song & Ermon, 2019) and flow models (Liu et al., 2022; Lipman et al., 2022) have emerged as a dominant class of generative models for synthesizing high-fidelity data across various modalities, from photorealistic imagery to complex structural biology. Beyond traditional density estimation, there is an increasing demand to align these models with specific downstream objectives, such as human preferences (Rafailov et al., 2023), biological evaluations (Avsec et al., 2021), and real-world physical feedback (Lei et al., 2025). Consequently, Reinforcement Learning (RL) for diffusion models has become a critical research direction to achieve the downstream alignment.

The existing RL algorithms for diffusion models generally fall into two categories (Gao et al., 2026). One prominent direction treats the denoising reverse process as a Markov Decision Process (MDP) and optimizes the model using policy gradient (Black et al., 2023; Fan et al., 2023; Ren et al., 2024; Liu et al., 2025; Zhang et al., 2025b) or reparameterization (Wang et al., 2024b; 2025; Celik et al., 2025b), which are computationally intensive. Furthermore, they usually mandate the use of Stochastic Differential Equation (SDE) samplers during training, which may not only require a very different training infrastructure from deployment, but also compromises the deployment performance since the sampler may differ.

An alternative RL paradigm, also known as advantage weighted regression (AWR) (Peng et al., 2019; Nair et al., 2020), reinforces the current policy by regressing toward behavior samples using weights that are exponential to their expected returns. Extending this concept to diffusion RL, Ma et al. (2025a) introduced a suite of efficient algorithms that solve *reweighted flow matching* problems, maintaining high compatibility with existing diffusion model training infrastructures. However, this paradigm tends to assign disproportionately high weights to a small subset of high-advantage samples while assigning negligible weights to the remainder, leading to over-greedy policy updates (Mei et al., 2020) and a failure to leverage information from negative samples (Frans et al., 2025).

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.

To mitigate the inherent limitations of exponential reweighting, we introduce Generalized Measure Matching for Diffusion Policy Optimization (GeMPO), a unified framework that significantly broadens the scope of reweighted flow matching through flexible weighting function design. We reformulate the derivation of reweighted flow matching into two logical stages: *First*, a target policy measure is constructed by solving a regularized RL objective; *Second*, we align the current policy with the target measure through reweighted flow matching. This perspective enables GeMPO to not only subsume existing weighting designs (Ding et al., 2024; Ma et al., 2025a; Tang et al., 2025) as special cases but also to decouple the framework from rigid exponential constraints, allowing the reweighting to be tailored to specific reward landscapes. Furthermore, by relaxing the standard non-negativity constraint on the target measure, GeMPO can accommodate a broad class of monotone reweighting functions, even the ones with negative weights. We provide interpretations on how this mechanism actively repels the policy from suboptimal regions, addressing a critical drawback of conventional designs. Extensive evaluations across multi-armed bandits, MuJoCo locomotion, and DNA sequence generation demonstrate that GeMPO achieves competitive or superior performance by leveraging these flexible reweighting schemes.

Due to space limit, more related works is presented in Appendix A.

2 PRELIMINARIES

KL-regularized Policy Optimization. We consider the Markov Decision Process (MDP) (Puterman, 2014) specified by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, P, \mu_0, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a reward function, $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition operator, $\mu_0 \in \Delta(\mathcal{S})$ is the initial distribution and $\gamma \in (0, 1)$ is the discount factor. We define $Q^\pi(\boldsymbol{s}, \boldsymbol{a}) = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) | \boldsymbol{s}_0 = s, \boldsymbol{a}_0 = a]$ as the state-action value function. Consider optimizing a policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$, via the KL-regularized objective function,

$$\pi = \underset{\pi \in \Pi}{\operatorname{argmax}} \, \mathbb{E}_{\boldsymbol{a} \sim \pi}[Q^{\pi_{\text{old}}}(\boldsymbol{s}, \boldsymbol{a})] - \lambda D_{\text{KL}}(\pi \| \pi_{\text{old}}; \boldsymbol{s}), \tag{1}$$

where $\Pi \doteq \{\pi(\cdot|s) | \int \pi(\cdot|s) = 1, \pi(\cdot|s) \geqslant 0, \forall s \in \mathcal{S}\}$ is the policy family mapping states to probability distributions on the action space. π_{old} is the current policy, $\lambda \in \mathbb{R}^+$ is the regularization strength, and the Kullback–Leibler (KL) divergence $D_{\text{KL}}(p||q) = \mathbb{E}_{x \sim p}[\log \frac{p(x)}{q(x)}]$ serves to constrain the updated policy within a trust region around the old policy π_{old}, similar to proximity-based algorithms like TRPO (Schulman, 2015) and PPO (Schulman et al., 2017). The closed-form optimal solution in policy family Π to this objective is given by an exponential reweighting of the current policy:

$$\pi^*(\boldsymbol{a}|\boldsymbol{s}) \propto \pi_{\text{old}}(\boldsymbol{a}|\boldsymbol{s}) \exp(Q^{\pi_{\text{old}}}(\boldsymbol{s}, \boldsymbol{a})/\lambda). \tag{2}$$

Diffusion and Flow Models. Diffusion models (Ho et al., 2020; Song & Ermon, 2019) or flow models (Lipman et al., 2022; Liu et al., 2022) learn data distributions by gradually perturbing clean data $\boldsymbol{x}_0 \sim p_{\text{data}}$ with Gaussian noise according to a forward process:

$$\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \sigma_t \boldsymbol{\epsilon}, \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}),$$

which emits the conditional distribution $q_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t; \alpha_t \boldsymbol{x}_t, \sigma_t^2)$. It then learns the score or velocity function to reverse this process to sample from the target distribution p_{data}. We employ the $\boldsymbol{v}$-prediction formulation (Lipman et al., 2022; 2024) in this paper, where the velocity $\boldsymbol{v}$ is defined as the time-derivative of $\boldsymbol{x}$, *i.e.*, $\boldsymbol{v} = \dot{\alpha}_t \boldsymbol{x}_0 + \dot{\sigma}_t \boldsymbol{\epsilon}$. They fit a parameterized network $D_\theta(\boldsymbol{x}_t, t)$ to match the *marginal* velocity field:

$$\boldsymbol{v}_t(\boldsymbol{x}_t) = \mathbb{E}_{\boldsymbol{x}_0 \sim p_{0|t}(\cdot|\boldsymbol{x}_t)} \left[\boldsymbol{v}_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0) \right], \tag{3}$$

where $p_{0|t}(\boldsymbol{x}_0|\boldsymbol{x}_t) \doteq \frac{q_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0) p_{\text{data}}(\boldsymbol{x}_0)}{\int q_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0) p_{\text{data}}(\boldsymbol{x}_0) d\boldsymbol{x}_0}$ is the posterior distribution, $\boldsymbol{v}_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)$ is the *conditional* velocity field:

$$\boldsymbol{v}_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0) = \frac{(\sigma_t \dot{\alpha}_t - \dot{\sigma}_t \alpha_t)}{\sigma_t} \boldsymbol{x}_0 + \frac{\dot{\sigma}_t}{\sigma_t} \boldsymbol{x}_t.$$

However, the marginal velocity field is not accessible due to the intractability of the posterior distribution, therefore in practice, we employ the conditional velocity field matching, which produces equivalent gradients as matching the marginal velocity field:

$$\mathcal{L}(\theta) = \mathbb{E}_{\boldsymbol{x}_0 \sim p_{\text{data}}, \boldsymbol{\epsilon} \sim \mathcal{N}} \left[\| D_\theta(\boldsymbol{x}_t, t) - \boldsymbol{v}_{t|0} \|^2 \right].$$

After training is complete, sampling is done by solving a stochastic or ordinary differential equation (SDE/ODE):

$$\mathrm{d}\boldsymbol{x}_t = \left((1+\eta^2)\boldsymbol{v}_t(\boldsymbol{x}_t) - \eta^2 \frac{\dot{\alpha}_t}{\alpha_t}\boldsymbol{x}_t\right)\mathrm{d}t + \eta\sqrt{2\sigma_t\dot{\sigma}_t - 2\frac{\dot{\alpha}_t}{\alpha_t}\sigma_t^2}\,\mathrm{d}\bar{\boldsymbol{w}}_t, \tag{4}$$

where $\eta \geqslant 0$ controls the stochasticity. Different specifications of the noise schedule $\{(\alpha_t, \sigma_t)\}$ lead to different methods. For example, the variance-exploding (VE) diffusion model (Song & Ermon, 2019) has the schedule $\alpha_t = 1, \sigma_t = \sqrt{t}$. Flow models (Lipman et al., 2022; Liu et al., 2022) employ linear interpolation with $\alpha_t = 1 - t, \sigma_t = t$, which simplifies the conditional velocity target to $\boldsymbol{v}_{t|0} = \boldsymbol{\epsilon} - \boldsymbol{x}_0$. Note that diffusion models may also employ $\boldsymbol{x}_0$- or $\boldsymbol{\epsilon}$-prediction, which are equivalent to $\boldsymbol{v}$-prediction used in this paper due to the reparameterization among the three (Li & He, 2025).

f-divergence describes the difference between two probability distributions (Csiszár & Shields, 2004; Liese & Vajda, 2006; Ali & Silvey, 1966). For a convex function $f : \mathbb{R}^+ \to \mathbb{R}$ with $f(1) = 0$, the corresponding f-divergence for two distributions p, q is defined as:

$$D_f(p\|q) = \mathbb{E}_{x\sim q}\left[f\left(\frac{p(x)}{q(x)}\right)\right], \tag{5}$$

where f is called the generator function. Different choices of f-divergence can cover a wide class of popular divergences, which will be explained later in Table 1 in Section 3.2.

Signed Measures and Notations. p, q in (5) are probability distributions, *i.e.*, non-negative measures with total unit mass. Signed measures extend this notion, which may assign negative mass to some sets, provided the total magnitude of its mass is finite. In contrast to $\Delta(\mathcal{A})$ as the space of probability measures over the action space $\mathcal{A}$, we let $\mathcal{M}_{\pm}(\mathcal{A})$ denote the space of signed measures over $\mathcal{A}$. The definition (5) can be extended to cases where p is a finite signed measure (Csiszár et al., 2002; Broniatowski & Keziou, 2006) and q remains a probability measure. This extension is well-defined under mild conditions[1].

3 GEMPO: REVISIT DIFFUSION RL FROM GENERALIZED MEASURE MATCHING

In this section, we revisit diffusion RL from a measure matching perspective. Specifically, diffusion RL can be understood as first constructing the target policy measure which solves the regularized RL objective, and then matching the current policy with the target measure (Section 3.1). This naturally leads to a reweighted flow matching objective, where the weighting function depends on the formulation of the target measure. Instantiated with various f-divergence metrics, this framework subsumes existing algorithms as special cases while also inspiring novel weighting schemes (Section 3.2). Furthermore, by relaxing the standard non-negativity constraint on the target measure, we introduce negative reweighting, which repels the policy from suboptimal actions and facilitates exploration (Section 3.3).

3.1 DIFFUSION RL VIA MEASURE MATCHING

We consider a generalized version of the regularized policy optimization problem (1):

$$\max_{\pi\in\Pi} \mathbb{E}_\pi[Q(\boldsymbol{s},\boldsymbol{a})] - \lambda D_f(\pi\|\pi_{\text{old}}). \tag{6}$$

Here, we assume $f : \mathbb{R}^+ \to \mathbb{R}$ is strictly convex and differentiable and $f(1) = 0$. Recall that the policy family Π here enforces the normalization constraint $\int \pi(\boldsymbol{a}|\boldsymbol{s})\mathrm{d}\boldsymbol{a} = 1$ and the non-negativity constraint $\pi(\boldsymbol{a}|\boldsymbol{s}) \geqslant 0$ for all $\boldsymbol{s} \in \mathcal{S}$ and $\boldsymbol{a} \in \mathcal{A}$.

Proposition 3.1 (Optimal solution to (6) in Π.)**.** *The optimal policy admits the form*

$$\pi^*(\boldsymbol{a}|\boldsymbol{s}) = \pi_{\text{old}}(\boldsymbol{a}|\boldsymbol{s}) g_f\left(\frac{Q(\boldsymbol{s},\boldsymbol{a}) - \nu(\boldsymbol{s})}{\lambda}\right), \tag{7}$$

where $\nu : \mathcal{S} \to \mathbb{R}$ are multipliers whose actual values depend on λ and Q, and g_f is the clipped inverse derivative function *satisfying $g_f(x) = (f')^{-1}(x)$ if $x > f'(0)$ otherwise 0.*

[1] Such as f being a measurable function on $\mathbb{R}$. Detailed derivations are provided in Appendix B.

The detailed derivation is deferred to Appendix C.1.

The next step is to match π^* using diffusion-parameterized policy $\pi_\theta \in \Pi_\theta \subseteq \Pi$, where Π_θ is policy family parameterized by diffusion models D_θ introduced in Section 2. A simple and commonly used technique is reweighted flow matching (Ding et al., 2024; Ma et al., 2025a):

$$\mathcal{L}(\theta) = \mathbb{E}_{\substack{\boldsymbol{s}\sim\mathcal{B} \\ \boldsymbol{a}_0\sim\pi_{\text{old}} \\ \epsilon\sim\mathcal{N}}} \left[w(\boldsymbol{s}, \boldsymbol{a}_0) \| D_\theta(\boldsymbol{s}, \boldsymbol{a}_t, t) - \boldsymbol{v}_{t|0} \|^2 \right], \text{where } w(\boldsymbol{s}, \boldsymbol{a}_0) \doteq g_f \left(\frac{Q(\boldsymbol{s}, \boldsymbol{a}_0) - \nu(\boldsymbol{s})}{\lambda} \right). \tag{8}$$

where $\mathcal{B}$ is the replay buffer, $\boldsymbol{a}_t = \alpha_t \boldsymbol{a}_0 + \sigma_t \epsilon$, and $w(\boldsymbol{s}, \boldsymbol{a}_0)$ is the reweighting factor in (7). In Appendix D, we show that reweighted flow matching recovers the velocity field of the target policy measure, and the optimal diffusion parameterization D_θ^* with respect to (8) is a weighted average of conditional velocities under the posterior distribution $p_{0|t}(\cdot|\boldsymbol{a}_t)$:

$$D_\theta^*(\boldsymbol{s}, \boldsymbol{a}_t, t) = \frac{\mathbb{E}_{\boldsymbol{a}_0\sim p_{0|t}}[w(\boldsymbol{s}, \boldsymbol{a}_0) \boldsymbol{v}_{t|0}(\boldsymbol{a}_t|\boldsymbol{a}_0)]}{\mathbb{E}_{\boldsymbol{a}_0\sim p_{0|t}}[w(\boldsymbol{s}, \boldsymbol{a}_0)]} \tag{9}$$

Compared to the unweighted formulation (3), this solution reweights the conditional velocity according to its value, thereby emphasizing actions with higher advantage.

Beyond reweighted flow matching, there exist other methods to align the current policy with the target measure, such as contrastive learning. We adopt reweighted flow matching for its simplicity and discuss alternative options in Appendix G.

3.2 Unifying Existing Reweighting Schemes via f-divergences

We next instantiate the framework with several common f-divergences, each inducing a corresponding reweighting function.

Table 1: Examples of f-divergence, their generator functions, and resulting reweighting factors.

Divergence D_f	Generator $f(x)$	reweighting factor $g_f(\cdot)$
Forward KL Divergence	$x \log x$	$\exp(x)$
χ^2-Divergence	$x^2 - 1$	x
α-Divergence ($\alpha > 1$)	$\frac{x^\alpha - x}{\alpha(\alpha-1)}$	$x^{\frac{1}{\alpha-1}}$

By choosing the f-divergence as the forward KL divergence, we recover the exponential weighting scheme used in AWR (Peng et al., 2019) and DPMD (Ma et al., 2025a).

Example 3.1. *(DPMD (Ma et al., 2025a) as forward KL-regularization) Let $D_f = D_{\text{KL}}$, therefore $f(x) = x \log x - x + 1$, $\bar{g}_f(x) = \exp(x)$, and $f'(0) = -\infty$. Recognizing $\nu(\boldsymbol{s})$ as the normalization factor $Z(\boldsymbol{s})$, we can recover DPMD with* (8)*:*

$$\mathcal{L}_{\text{DPMD}}(\theta) = \mathbb{E}_{\boldsymbol{s}, \boldsymbol{a}_0, \epsilon} \left[\frac{\exp(Q(\boldsymbol{s}, \boldsymbol{a}))}{Z(\boldsymbol{s})} \left\| D_\theta(\boldsymbol{s}, \boldsymbol{a}_t, t) - \boldsymbol{v}_{t|0} \right\|^2 \right].$$

Moreover, we can see QVPO (Ding et al., 2024), originally derived from a lower bound of policy gradient objective, as a special case of specifying D_f as χ^2 divergence.

Example 3.2. *(QVPO Ding et al. (2024) as χ^2 regularization) Let D_f be the χ^2-divergence by setting $f(x) = x^2 - 1$, $\bar{g}_f(x) = \frac{1}{2}x$, and $f'(0) = 0$. The QVPO objective*

$$\mathcal{L}_{\text{QVPO}}(\theta) = \mathbb{E}_{\boldsymbol{s}, \boldsymbol{a}_0, \epsilon} \left[[Q(\boldsymbol{s}, \boldsymbol{a}) - V(\boldsymbol{s})]_+ \| D_\theta(\boldsymbol{s}, \boldsymbol{a}_t, t) - \boldsymbol{v}_{t|0} \|^2 \right]$$

can be recovered by (8) *through fixing $\lambda = \frac{1}{2}$ and $[\cdot]_+ = \max(\cdot, 0)$.*

By recognizing KL and χ^2 divergences as specific instantiations of the α-divergences, we can extend our framework to a more generalized regularization family.

Example 3.3. *(Power function reweighting as α-divergence regularization) For general α-divergences with $\alpha > 1$, we have $f(x) = \frac{x^\alpha - 1}{(\alpha-1)\alpha}$, which induces $f'(0) = 0$ and $\bar{g}_f(x) = (\alpha - 1)x^{\frac{1}{\alpha-1}}$. The target policy can be therefore defined as*

$$\pi^*(\boldsymbol{a}|\boldsymbol{s}) \propto \pi_{\text{old}}(\boldsymbol{a}|\boldsymbol{s}) \left[\frac{Q(\boldsymbol{s},\boldsymbol{a}) - \nu(\boldsymbol{s})}{\lambda}\right]_+^{\frac{1}{\alpha-1}},$$

where certain constant factors are absorbed into λ. This suggests the following objective:

$$\mathcal{L}_\alpha(\theta) = \mathbb{E}_{\boldsymbol{s},\boldsymbol{a}_0,\epsilon}\left[\left[\frac{Q(\boldsymbol{s},\boldsymbol{a}) - \nu(\boldsymbol{s})}{\lambda}\right]_+^{\frac{1}{\alpha-1}} \left\|D_\theta(\boldsymbol{s},\boldsymbol{a}_t,t) - \boldsymbol{v}_{t|0}\right\|^2\right].$$

By tuning α, we can shape the curvature of the weighting function to better match the characteristics of the underlying value landscape. With an appropriate choice of α, our approach can avoid both the overly flat weighting used in QVPO (Example 3.2) and the statistical instabilities induced by the exponential weighting in DPMD (Example 3.1).

3.3 Extension to Negative Reweighting

Current policy family Π strictly enforces the non-negativity constraint $\pi^*(\boldsymbol{a}|\boldsymbol{s}) \geqslant 0$, which consequently forces the weights $g_f(\cdot)$ or $w(\cdot,\cdot)$ to be non-negative. In practice, this assigns infinitesimal or zero weight to suboptimal samples. As a result, these methods fail to exploit negative samples during training and are prone to getting trapped in local optima.

Our measure matching perspective relaxes the policy family in (6) to mapping from states to normalized signed measures $\widetilde{\Pi} \doteq \{\tilde{\pi} : \mathcal{S} \to \mathcal{M}_\pm(\mathcal{A}) | \int \tilde{\pi}(a|s) = 1, \forall s\} \supseteq \Pi$.

Proposition 3.2 (Optimal solution to (6) in $\widetilde{\Pi}$)**.** *We extend D_f in* (6) *to allow p being signed measures and policy family to be $\widetilde{\Pi}$. Then the optimal solution $\tilde{\pi}^* \in \widetilde{\Pi}$ to* (6) *is:*

$$\tilde{\pi}^*(\boldsymbol{a}|\boldsymbol{s}) \propto \pi_{\text{old}}(\boldsymbol{a}|\boldsymbol{s})\tilde{g}_f\left(\frac{Q(\boldsymbol{s},\boldsymbol{a}) - \nu(\boldsymbol{s})}{\lambda}\right), \tag{10}$$

where $\tilde{g}_f(\cdot)$ is a monotonic increasing function and ***not required to be non-negative****. The derivation is deferred to Appendix C.2.*

Correspondingly, using reweighted flow matching (8) $w(\boldsymbol{s},\boldsymbol{a}) = \tilde{g}_f\left((Q(\boldsymbol{s},\boldsymbol{a}) - \nu(\boldsymbol{s}))/\lambda\right)$ might be negative. This result justifies the use of negative weights and indicates that existing negative-aware reweighting methods fit precisely within our measure matching framework.

Example 3.4 (**WD1** Tang et al. (2025) as an example of negative reweighting)**. WD1** *Tang et al. (2025) also incorporates negative reweighting to enhance the reasoning capability of diffusion language models. Their weighting function is defined as*

$$\tilde{g}_f(\hat{A}(\boldsymbol{s},\boldsymbol{a}_i)) = \text{softmax}(\hat{A}(\boldsymbol{s},\boldsymbol{a}_i)) - \text{softmax}(-\hat{A}(\boldsymbol{s},\boldsymbol{a}_i))$$

where $\text{softmax}(\hat{A}(\boldsymbol{s},\boldsymbol{a}_i)) = \frac{\exp(\hat{A}(\boldsymbol{s},\boldsymbol{a}_i))}{\sum_i \exp(\hat{A}(\boldsymbol{s},\boldsymbol{a}_i))}$ *and* $\hat{A}(\boldsymbol{s},\boldsymbol{a}_i) = \frac{R(\boldsymbol{s},\boldsymbol{a}_i) - \text{mean}(R(\boldsymbol{s},\boldsymbol{a}_{1:G}))}{\text{std}(R(\boldsymbol{s},\boldsymbol{a}_{1:G}))}$ *is the group-relative advantage across a total number of G responses to the same query $\boldsymbol{s}$. Note that, $g(x)$ here is exactly monotonic increasing with respect to the advantage function.*

The repelling effect caused by negative reweighting. In practice, negative reweighting allows the model to leverage low-value samples by explicitly pushing the generation trajectory away from them. Recall the optimal diffusion parameterization in (9). When $\mathbb{E}_{a_0 \sim p_{0|t}}[w(s,a_0)] > 0$, the numerator forms a weighted average of conditional velocity fields. For samples with negative weights, their corresponding conditional velocities are effectively reversed in direction. As a result, low-value samples contribute a repulsive velocity component, steering the learned generation path away from suboptimal regions rather than merely assigning them negligible weight. Cases where $\mathbb{E}_{a_0 \sim p_{0|t}}[w(s,a_0)] \leqslant 0$ can induce an analogous repelling effect, which we discuss in Appendix J.

4 PRACTICAL ALGORITHMS

Algorithm 1 Generalized **M**easure Matching for **P**olicy **O**ptimization (GeMPO)

Require: Q-function, current policy π_{old}, regularization strength λ, number of samples N

1: sample N actions $a_i \sim \pi(\cdot|\boldsymbol{s}), i = 1, \dots N$ for each $\boldsymbol{s}$
2: Evaluate $Q_i \doteq Q(\boldsymbol{s}, \boldsymbol{a}_i), i = 1, 2, \dots N$
3: Compute the empirical normalization factor $\nu(\boldsymbol{s})$
4: Compute reweighting weights $w(\boldsymbol{s}, \boldsymbol{a}_i)$ with ν, Q, λ
5: Train policies by reweighted flow matching (8)

The pseudo-code of the GeMPO framework is presented in Algorithm 1. By instantiating the reweighted flow matching with different weighting functions, we study several variants of GeMPO in this paper. Specifically, we consider: 1) `GeMPO-Exp`, which uses standard/exponential weighting similar with DPMD; 2) `GeMPO-Linear` and `GeMPO-Square`, which uses power functions $w(\boldsymbol{s}, \boldsymbol{a}) = [(Q(\boldsymbol{s}, \boldsymbol{a}) - \nu(\boldsymbol{s})/\lambda]_+$ and $w(\boldsymbol{s}, \boldsymbol{a}) = [(Q(\boldsymbol{s}, \boldsymbol{a}) - \nu(\boldsymbol{s}))/\lambda]_+^2$, respectively. For variants that allow negative weighting, as discussed in Appendix J, negative weights must be carefully controlled to avoid destabilizing learning. To this end, we apply a simple truncation:

$$w(\boldsymbol{s}, \boldsymbol{a}) = \max\left(\frac{Q(\boldsymbol{s}, \boldsymbol{a}) - \nu(\boldsymbol{s})}{\lambda}, \ell\right), \tag{11}$$

where $\ell < 0$ shapes the magnitude of negative weights. We denote this variant as `GeMPO-Neg` (ℓ) .

Normalization constraints. It is not clear how to compute the normalization term $\nu(\boldsymbol{s})$ for general reweighting functions. We propose a population-based solution by estimating the normalizer with a group of samples. For exponential weighting $g(\cdot) = \exp(\cdot)$, this translates to computing the softmax as the final weight; while for power function weighting, we can compute the normalizers by first determining the number of active samples (*i.e.*, samples that are not clipped), and then compute the exact value such that the weights are normalized. Details can be found in Appendix E.

5 EXPERIMENTS

5.1 EXPLORATION-EXPLOITATION TRADE-OFF IN BANDIT

We first illustrate the effect of different weighting schemes on the exploration-exploitation trade-off through a one-dimensional bandit problem, where we optimize a stateless policy $\pi(\cdot)$ to maximize an unknown reward function $R(\cdot)$. We evaluate the performance using the simple regret metrics, given by $\max_{a \in \mathcal{A}} R(a) - \mathbb{E}_{a \sim \pi}[R(a)]$.

Effect of negative weights on exploration. We consider a reward function with two optima (Figure 1, left) and initialize the policy near the suboptimal one. We compare three nonnegative weighting functions in (8) (`Linear`, `Square`, and `Exp`) with negative reweighting in (11) (`Neg`), using the default lower bound $\ell = -0.3$.

The regret curves (Figure 1, right) show that none of the nonnegative methods reaches the global optimum in every trial; `Linear` remains trapped at the local optimum in all trials. In contrast, `Neg` escapes the suboptimal region (Figure 1, middle), suggesting that negative weights promote exploration. Varying the lower bound reveals a tradeoff: decreasing ℓ initially reduces regret, but more negative values increase regret as training becomes unstable. The best results occur at $\ell = -0.3$ and -0.5.

Reweighting and reward landscape. We compare reward functions with two broad or two sharp optima (Figure 2, left) using three nonnegative reweighting functions: `Linear`, `Square`, and `Exp`. The expected simple regret curves (Figure 2, right) show that `Linear` performs best for sharp optima, whereas `Square` performs best for broad optima; both outperform `Exp`. Additional experiments on two DeepMind Control Suite tasks, reported in the Appendix H.1, vary reward curvature and show the same pattern: `Linear` performs better for steep rewards, while `Square` performs better for flatter rewards.

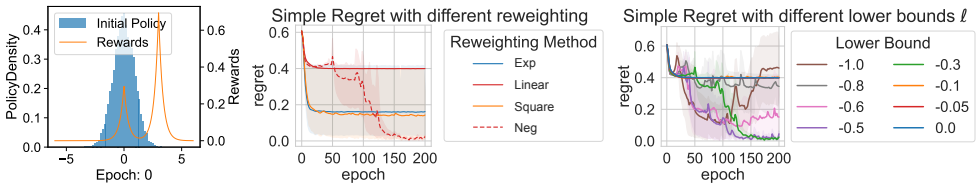


Figure 1: **Effect of negative weights on exploration. Left**: Reward functions and initial policy. **Middle**: Regret curves comparing different reweighting with and without negative reweighting functions over 200 epochs. **Right**: Regret curves with different lower bound ℓ.

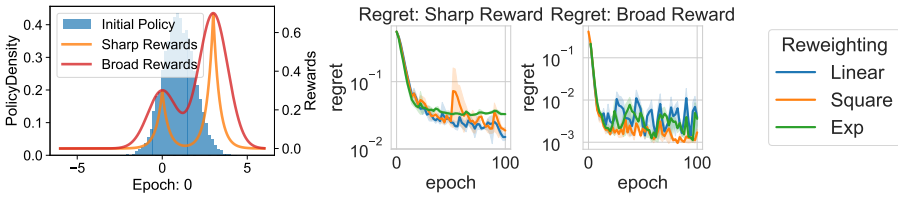


Figure 2: **Reweighting and reward landscape. Left:** Two reward functions and initial policy. **Right:** Regret curves comparing different reweighting schemes over 100 epochs. `Linear` excels in sharp rewards and `Square` excels at broad rewards, both outperform `Exp`.

5.2 Locomotion Tasks

Table 2: **Overall performance on Gym-MuJoCo environments.** We report the average returns of the last 10 checkpoints ± standard error over 5 random seeds, and normalized average score normalized by SAC score. **Bold** numbers indicate best mean returns and underlined numbers indicate mean returns within standard error of the best runs.

Method	HalfCheetah	Humanoid	Ant	Walker2d	Hopper	Swimmer	Norm. Avg.(%)
Classic Model-Free RL							
TD3	9820 ± 1543	5263 ± 47	4400 ± 1196	3732 ± 388	**3387** ± 302	88 ± 25	107.4
SAC	10938 ± 68	5131 ± 97	4945 ± 186	4463 ± 151	2509 ± 276	61 ± 5	100.0
Diffusion RL Baselines							
DIPO	10 136 ± 2525	5184 ± 132	977 ± 20	3809 ± 1112	1191 ± 770	46 ± 3	70.3
QSM	10 444 ± 250	5103 ± 73	3614 ± 360	3635 ± 621	3200 ± 255	86 ± 7	103.0
DACER	7669 ± 1539	**5332** ± 64	**5141** ± 356	4252 ± 176	2151 ± 495	79 ± 28	98.1
QVPO	8655 ± 209	5131 ± 69	3852 ± 67	3577 ± 545	3312 ± 76	**110** ± 5	108.3
Ours							
`GeMPO-Exp`	**10989** ± 329	5293 ± 55	4719 ± 58	**4695** ± 197	3350 ± 58	99 ± 13	116.7
`GeMPO-Square`	10679 ± 500	5179 ± 49	4942 ± 75	4622 ± 83	3325 ± 113	102 ± 20	**117.0**
`GeMPO-Linear`	9812 ± 506	5265 ± 74	4819 ± 73	4683 ± 96	3062 ± 97	99 ± 11	113.2

We also conducted evaluation on standard RL benchmarks, by implementing our algorithm with the JAX package and evaluating the performance on 6 OpenAI Gym MuJoCo v5 tasks. All environments are trained with a total of 1 million environment interactions. Baselines include two families of model-free RL algorithms. The first family is a collection of recent diffusion-policy online RL algorithms, including QSM Psenka et al. (2023), QVPO Ding et al. (2024), DACER Wang et al. (2024b), and DIPO Yang et al. (2023). The second family is classic model-free online RL baselines including TD3 Fujimoto et al. (2018) and SAC Haarnoja et al. (2018). A more detailed explanation to the baselines can be found in Appendix I.1.

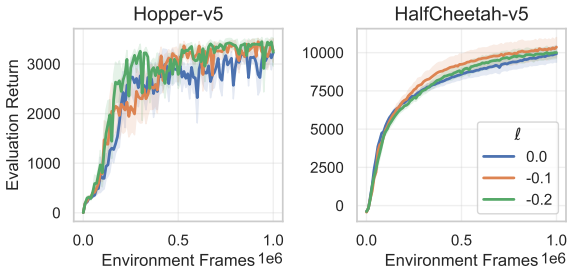


Figure 3: Training curves of `GeMPO-Neg`(ℓ) with different truncation lower bound ℓ. Setting $\ell = 0$ corresponds to `GeMPO-Linear`.

Table 3: Training and evaluation scores for SD3.5-Medium fine-tuning. Aesthetic Score and ImageReward are used only for evaluation. For Flow-GRPO, we evaluated open-sourced checkpoints.

		Training rewards			Evaluation only	
Method	Iter.	PickScore ↑	CLIPScore ↑	HPS v2.1 ↑	Aesthetic ↑	ImageReward ↑
SD3.5-M + CFG	—	22.34	0.285	0.279	5.36	0.85
Flow-GRPO	> 5k	22.51	**0.293**	0.274	5.32	1.06
DiffusionNFT	160	22.829	0.287	0.340	6.55	1.42
AWM	160	22.869	0.289	0.339	6.54	1.40
`GeMPO-Neg`	160	**22.952**	0.288	**0.342**	**6.58**	**1.44**

Overall performance. The results are summarized in Table 2. All GeMPO variants, `Linear`, `Square`, and `Exp`, exhibit consistently strong performance and outperform other diffusion-based RL baselines across most environments. Especially, `GeMPO-Square` achieves the highest normalized average score, highlighting the benefit from flexible general reweighting function in our framework.

Effect of negative weighting on exploration. As shown in Figure 3, setting $\ell = -0.1$ or -0.2 yields modest performance gains over the Linear variant on HalfCheetah-v5 and Hopper-v5. Negative weighting helps by reducing the influence of low-value actions during policy updates and moving the policy away from suboptimal regions to explore alternatives.

5.3 RL Finetuning for Image Generation Models

To further evaluate GeMPO, we use it to fine-tune the SD3.5-Medium (Esser et al., 2024) image-generation model following the experimental setup of DiffusionNFT (Zheng et al., 2025). We optimize a combined reward comprising PickScore (Kirstain et al., 2023), HPS v2.1 (Wu et al., 2023b), and CLIPScore (Hessel et al., 2021), then evaluate the resulting model on Aesthetic Score (Schuhmann & Beaumont, 2022) and ImageReward (Xu et al., 2023b) as additional metrics. We compare three GeMPO variants with DiffusionNFT (Zheng et al., 2025) and AWM (Xue et al., 2025) under the same GPU-hour budget.

As shown in Table 3, `GeMPO-Neg` achieves the highest PickScore, HPS v2.1, Aesthetic Score, and ImageReward after 160 iterations. It also improves CLIPScore over DiffusionNFT (0.288 versus 0.287), although AWM attains a slightly higher CLIPScore. The Aesthetic Score and ImageReward gains suggest that the improvement extends beyond the rewards used for training. Flow-GRPO scores lower on four of the five metrics despite its checkpoint having undergone more than 5,000 iterations, demonstrating the efficiency of GeMPO algorithms.

5.4 DNA Sequence Generation

Finally, we evaluate our method on a real-world problem, learning to generate DNA sequences with a discrete diffusion model[2]. The task is to fine-tune a DNA generation model, learned from a large public enhancer dataset containing roughly 700,000 DNA sequences (Gosai et al., 2023), to optimize the gene expression activity Avsec et al. (2021). We leverage pretrained generation and reward models from Wang et al. (2024a). We tested multiple variants of our algorithm, including `Linear`, `Square`, and `Neg`.

Baselines. We compare against controlled generation approaches including conditional guidance (CG, Nisonoff et al., 2024), SMC and (TDS, Wu et al., 2023a), and classifier-free guidance (CFG, Ho & Salimans, 2022), as well as a strong RL-based baseline, DRAKES (Wang et al., 2024a), which leveraged reparameterized policy gradient using Gumbel-softmax, and RL-D^2 Ma et al. (2025b), which uses weighted regression with KL divergence regularization (the same as `GeMPO-Exp`).

Results. Our experimental results, summarized in Table 4, demonstrate that GeMPO significantly outperforms all considered baselines in terms of the gene expression levels. While the

[2]Despite different definitions, discrete diffusion models share most of the properties with continuous diffusion, such as the optimal solution in Equation (3). Please refer to Lou et al. (2023); Shi et al. (2024) for details.

Table 4: **Main Results on Pred-Activity.** Comparison of GeMPO variants against baselines. Standard GeMPO variants are highlighted in orange, while variants with negative sample awareness are highlighted in blue and named as Neg(ℓ). The inclusion of negative samples consistently provides the highest gains up to 14.3%.

Baselines		
Method	**Pred. ↑**	**Improv.**
Pretrained	0.17 ± 0.04	–
CG	3.30 ± 0.01	–
TDS	4.64 ± 0.21	–
CFG	5.04 ± 0.06	–
DRAKES	6.44 ± 0.04	-1.2%
DRAKES-KL	5.61 ± 0.07	-13.9%
WD1	3.78 ± 0.12	-42.0%
RL-D^2 (best baseline)	$\mathbf{6.52 \pm 0.03}$	–

GeMPO Variants		
Method	**Pred. ↑**	**Improv.**
`Linear`	6.54 ± 0.01	+0.3%
`Square`	$\mathbf{6.77 \pm 0.03}$	**+3.8%**
`Neg(-0.05)`	6.75 ± 0.05	+3.5%
`Neg(-0.1)`	6.95 ± 0.07	+6.6%
`Neg(-0.3)`	$\mathbf{7.45 \pm 0.03}$	**+14.3%**
`Neg(-0.5)`	5.62 ± 1.13	-13.8%

best-performing baseline, RL-D^2, achieves a median score of 6.52, our standard linear and square GeMPO variants already provide competitive or superior performance. However, the most substantial gains are observed when incorporating negative sample awareness. Specifically, `GeMPO-Neg(-0.3)` variant achieves the highest overall performance of 7.45 ± 0.03, representing a $+14.3\%$ improvement over the best baseline. Moreover, the `GeMPO-Neg` variants show that the performance first increases and then decreases as ℓ decreases from 0 to -0.5.

5.5 Sensitivity Analysis

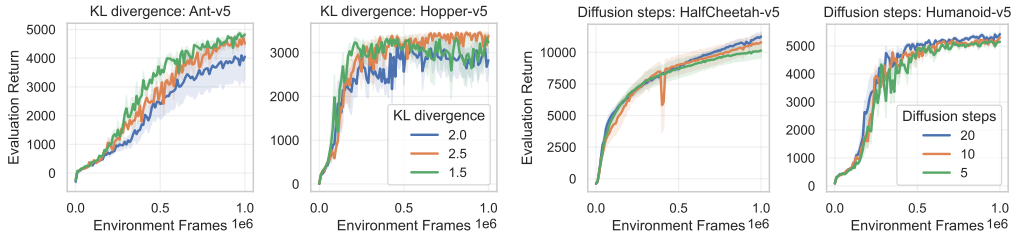


Figure 4: Ablation studies on KL tolerance and diffusion steps.

Ablation on the KL tolerance. GeMPO employs automatic temperature tuning by constraining the KL divergence between the weighting distribution $w(\cdot)$ and the uniform distribution (Appendix F). A higher KL tolerance permits more aggressive temperature adjustments, leading to a greedier reweighting. Our evaluation (Figure 4) indicates that the optimal tolerance is task-dependent, necessitating hyperparameter tuning across different domains.

Ablation on diffusion steps. Results in Figure 4 indicate that performance is largely insensitive to the number of diffusion steps. While we observe a marginal improvement in accuracy as the step count increases, the overall gains remain minimal, suggesting that GeMPO is robust to this hyperparameter.

6 Conclusions

In this work, we established a cohesive framework which uses Generalized Measure matching for diffusion Policy Optimization (GeMPO). By decoupling algorithm design into target measure construction and measure matching, we clarify the broader design space of reweighted flow matching. We demonstrate that by adopting different f-divergences, our framework recovers several existing algorithms, such as QVPO and DPMD, as its special cases. Furthermore, by extending the target to signed measure families, our framework provides a justification for negative weights, thereby resolving a critical exploration dilemma inherent in previous reweighted flow matching methods. One limitation of this negative reweighting is that it might cause an unbounded loss function, while our results showed that proper ℓ could avoid instability. We hope this paper will inspire more efficient and effective post-training algorithms for diffusion models and flow models.

AI USE STATEMENT

In this work, we used generative AI tools to assist in the writing of the paper and assist in translation.

We have not used generative AI tools to generate synthetic datasets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, clean and reformat datasets, support qualitative and thematic data analysis, or interpret results.

Additionally, we used generative AI tools to organize and improve the visualization of the experiment results.

We reviewed all AI-assisted content, including text and figures, to ensure that it accurately reflects the paper's ideas and underlying data. We take full responsibility for the final content of this work, including any text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide experimental details, including the hyperparameters for GeMPO and implementation details for its key components, in Appendix I and Appendix E.4. To facilitate reproducibility, we also include the source code for GeMPO in the supplementary materials.

REFERENCES

Tara Akhound-Sadegh, Jarrid Rector-Brooks, Avishek Joey Bose, Sarthak Mittal, Pablo Lemos, Cheng-Hao Liu, Marcin Sendera, Siamak Ravanbakhsh, Gauthier Gidel, Yoshua Bengio, et al. Iterated denoising energy matching for sampling from boltzmann densities. *arXiv preprint arXiv:2402.06121*, 2024.

Syed Mumtaz Ali and Samuel D Silvey. A general class of coefficients of divergence of one distribution from another. *Journal of the Royal Statistical Society: Series B (Methodological)*, 28(1): 131–142, 1966.

Žiga Avsec, Vikram Agarwal, Daniel Visentin, Joseph R Ledsam, Agnieszka Grabska-Barwinska, Kyle R Taylor, Yannis Assael, John Jumper, Pushmeet Kohli, and David R Kelley. Effective gene expression prediction from sequence by integrating long-range interactions. *Nature methods*, 18 (10):1196–1203, 2021.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. *arXiv preprint arXiv:2305.13301*, 2023.

Michel Broniatowski and Amor Keziou. Minimization of φ-divergences on sets of signed measures. *Studia Scientiarum Mathematicarum Hungarica*, 43(4):403–442, 2006.

Onur Celik, Zechu Li, Denis Blessing, Ge Li, Daniel Palanicek, Jan Peters, Georgia Chalvatzaki, and Gerhard Neumann. Dime: Diffusion-based maximum entropy reinforcement learning. *arXiv preprint arXiv:2502.02316*, 2025a.

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 *International Conference on Machine Learning*, pp. 6958–6977. PMLR, 2025b.

Kevin Clark, Paul Vicol, Kevin Swersky, and David J Fleet. Directly fine-tuning diffusion models on differentiable rewards. *arXiv preprint arXiv:2309.17400*, 2023.

Imre Csiszár and Paul C Shields. Information theory and statistics: A tutorial. 2004.

Imre Csiszár, F Gamgoa, and Elisabeth Gassiat. Mem pixel correlated solutions for generalized moment and interpolation problems. *IEEE Transactions on Information Theory*, 45(7):2253–2270, 2002.

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. *arXiv preprint arXiv:2405.16173*, 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. *arXiv preprint arXiv:2505.18763*, 2025.

Xiaoyi Dong, Jian Cheng, and Xi Sheryl Zhang. Maximum entropy reinforcement learning with diffusion policy. *arXiv preprint arXiv:2502.11612*, 2025.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

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. *Advances in Neural Information Processing Systems*, 36:79858–79885, 2023.

Kevin Frans, Seohong Park, Pieter Abbeel, and Sergey Levine. Diffusion guidance is a controllable policy improvement operator. *arXiv preprint arXiv:2505.23458*, 2025.

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.

Chen-Xiao Gao, Chenyang Wu, Mingjun Cao, Chenjun Xiao, Yang Yu, and Zongzhang Zhang. Behavior-regularized diffusion policy optimization for offline reinforcement learning. In *Forty-second International Conference on Machine Learning*, 2025.

Chenxiao Gao, Edward Chen, Tianyi Chen, and Bo Dai. Flowrl: A taxonomy and modular framework for reinforcement learning with diffusion policies. *arXiv preprint arXiv:2603.27450*, 2026.

Sager J Gosai, Rodrigo I Castro, Natalia Fuentes, John C Butts, Susan Kales, Ramil R Noche, Kousuke Mouri, Pardis C Sabeti, Steven K Reilly, and Ryan Tewhey. Machine-guided design of synthetic cell type-specific cis-regulatory elements. *bioRxiv*, 2023.

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.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 7514–7528, 2021.

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

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020.

Yuval Kirstain, Adam Polyak, Uriel Singer, Shahbuland Matiana, Joe Penna, and Omer Levy. Pick-a-pic: An open dataset of user preferences for text-to-image generation. *Advances in neural information processing systems*, 36:36652–36663, 2023.

Kun Lei, Huanyu Li, Dongjie Yu, Zhenyu Wei, Lingxiao Guo, Zhennan Jiang, Ziyu Wang, Shiyu Liang, and Huazhe Xu. Rl-100: Performant robotic manipulation with real-world reinforcement learning. *arXiv preprint arXiv:2510.14830*, 2025.

Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. *arXiv preprint arXiv:2511.13720*, 2025.

Zeyang Li, Sunbochen Tang, and Navid Azizan. Reverse flow matching: A unified framework for online reinforcement learning with diffusion and flow policies. *arXiv preprint arXiv:2601.08136*, 2026.

Friedrich Liese and Igor Vajda. On divergences and informations in statistics and information theory. *IEEE Transactions on Information Theory*, 52(10):4394–4412, 2006.

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

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *arXiv preprint arXiv:2412.06264*, 2024.

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*, 2025.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*, 2023.

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.

Haitong Ma, Tianyi Chen, Kai Wang, Na Li, and Bo Dai. Efficient online reinforcement learning for diffusion policy. *arXiv preprint arXiv:2502.00361*, 2025a.

Haitong Ma, Ofir Nabati, Aviv Rosenberg, Bo Dai, Oran Lang, Idan Szpektor, Craig Boutilier, Na Li, Shie Mannor, Lior Shani, et al. Reinforcement learning with discrete diffusion policies for combinatorial action spaces. *arXiv preprint arXiv:2509.22963*, 2025b.

Jincheng Mei, Chenjun Xiao, Bo Dai, Lihong Li, Csaba Szepesvári, and Dale Schuurmans. Escaping the gravitational pull of softmax. *Advances in Neural Information Processing Systems*, 33:21130–21140, 2020.

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

Hunter Nisonoff, Junhao Xiong, Stephan Allenspach, and Jennifer Listgarten. Unlocking guidance for discrete state-space diffusion and flow models. *arXiv preprint arXiv:2406.01572*, 2024.

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.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, et al. $\pi^*_{0.6}$: a vla that learns from experience. *arXiv preprint arXiv:2511.14759*, 2025.

Mihir Prabhudesai, Russell Mendonca, Zheyang Qin, Katerina Fragkiadaki, and Deepak Pathak. Video diffusion alignment via reward gradients. *arXiv preprint arXiv:2407.08737*, 2024.

Michael Psenka, Alejandro Escontrela, Pieter Abbeel, and Yi Ma. Learning a diffusion model policy from rewards via q-score matching. *arXiv preprint arXiv:2312.11752*, 2023.

Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Allen Z Ren, Justin Lidard, Lars L Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. *arXiv preprint arXiv:2409.00588*, 2024.

Christoph Schuhmann and Romain Beaumont. Laion-aesthetics. *LAION. AI*, 3, 2022.

John Schulman. Trust region policy optimization. *arXiv preprint arXiv:1502.05477*, 2015.

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

Lior Shani, Yonathan Efroni, and Shie Mannor. Adaptive trust region policy optimization: Global convergence and faster rates for regularized mdps. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 5668–5675, 2020.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. *Advances in neural information processing systems*, 37: 103131–103167, 2024.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Xiaohang Tang, Rares Dolga, Sangwoong Yoon, and Ilija Bogunovic. wd1: Weighted policy optimization for reasoning in diffusion language models. *arXiv preprint arXiv:2507.08838*, 2025.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Manan Tomar, Lior Shani, Yonathan Efroni, and Mohammad Ghavamzadeh. Mirror descent policy optimization, 2021. URL https://arxiv.org/abs/2005.09814.

Nino Vieillard, Tadashi Kozuno, Bruno Scherrer, Olivier Pietquin, Rémi Munos, and Matthieu Geist. Leverage the average: an analysis of kl regularization in rl. *arXiv preprint arXiv:2003.14089*, 2020.

Chenyu Wang, Masatoshi Uehara, Yichun He, Amy Wang, Tommaso Biancalani, Avantika Lal, Tommi Jaakkola, Sergey Levine, Hanchen Wang, and Aviv Regev. Fine-tuning discrete diffusion models via reward optimization with applications to dna and protein design. *arXiv preprint arXiv:2410.13643*, 2024a.

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, December 2024b.

Yinuo Wang, Likun Wang, Mining Tan, Wenjun Zou, Xujie Song, Wenxuan Wang, Tong Liu, Guojian Zhan, Tianze Zhu, Shiqi Liu, et al. Enhanced dacer algorithm with high diffusion efficiency. *arXiv preprint arXiv:2505.23426*, 2025.

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

Luhuan Wu, Brian Trippe, Christian Naesseth, David Blei, and John P Cunningham. Practical and asymptotically exact conditional sampling in diffusion models. *Advances in Neural Information Processing Systems*, 36:31372–31403, 2023a.

Xiaoshi Wu, Keqiang Sun, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score: Better aligning text-to-image models with human preference. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2096–2105. IEEE, 2023b.

Haoran Xu, Li Jiang, Jianxiong Li, Zhuoran Yang, Zhaoran Wang, Victor Wai Kin Chan, and Xianyuan Zhan. Offline rl with no ood actions: In-sample learning via implicit value regularization. *arXiv preprint arXiv:2303.15810*, 2023a.

Haoran Xu, Liyuan Mao, Hui Jin, Weinan Zhang, Xianyuan Zhan, and Amy Zhang. Uni-rl: Unifying online and offline rl via implicit value regularization. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. Imagereward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36:15903–15935, 2023b.

Shuchen Xue, Chongjian Ge, Shilong Zhang, Yichen Li, and Zhi-Ming Ma. Advantage weighted matching: Aligning rl with pretraining in diffusion models. *arXiv preprint arXiv:2509.25050*, 2025.

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.

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

Tonghe Zhang, Chao Yu, Sichang Su, and Yu Wang. Reinflow: Fine-tuning flow matching policy with online reinforcement learning. *arXiv preprint arXiv:2505.22094*, 2025b.

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*, 2025c.

Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. Diffusionnft: Online diffusion reinforcement with forward process. *arXiv preprint arXiv:2509.16117*, 2025.

A Related works

RL for diffusion and flow models. RL for diffusion or flow models can be classified into two categories: **a)** Only working with the reverse process, either by computing *policy gradient* via accumulating the log probabilities over the diffusion timesteps Black et al. (2023); Fan et al. (2023); Ren et al. (2024); Liu et al. (2025); Zhang et al. (2025b), or directly backpropagating gradients through the reverse process by *reparameterization* Clark et al. (2023); Wang et al. (2024b); Prabhudesai et al. (2024); Celik et al. (2025a); Ding et al. (2025); Zhang et al. (2025c); Gao et al. (2025). Such methods closely resemble standard RL algorithms, such as Proximal Policy Optimization (PPO) Schulman et al. (2017) and Soft Actor Critic (SAC) Haarnoja et al. (2018), respectively. However, their computational cost typically scales linearly with the number of reverse diffusion steps (Wang et al., 2022; Gao et al., 2025). Moreover, they require a training pipeline that is substantially different from standard score or flow matching, as we need to generate and differentiate through the denoising trajectory. **b)** Another way is preserving the forward process, and steering the score or flow prediction through *weighted behavior cloning* (Ding et al., 2024; Ma et al., 2025a; Zhang et al., 2025a; Tang et al., 2025; Dong et al., 2025; Li et al., 2026) or *moving towards calibrated targets* (Lu et al., 2023; Psenka et al., 2023; Frans et al., 2025; Akhound-Sadegh et al., 2024). In this way, the optimization is well-formulated as distribution matching. In this paper, we dive into the weighted flow matching formula and explore more reweighting options.

Regularizations in policy extraction. KL divergence is widely adopted to regularize the policy improvement Peng et al. (2019); Vieillard et al. (2020); Nair et al. (2020). However, it is also possible to adopt other types of divergence metrics. Several works also consider reverse KL regularization Xu et al. (2025); Ma et al. (2025b), leading to update rules similar to policy gradient theorems. Xu et al. (2023a; 2025) further study implicit value regularization, which derives closed-form target policies and update rules for general f-divergence regularized problems. In this paper, we deepen this understanding by extending it to both diffusion policy and signed target measures, thus allowing more flexible and effective objective design. Finally, another branch of research about policy mirror descent is motivated by the Bregman divergence Tomar et al. (2021); Shani et al. (2020). Notably, the KL divergence occupies a special position, as it is both a Bregman divergence and an f-divergence.

Leveraging negative samples in RL for diffusion models. Negative weights are typically underutilized in weighted behavior cloning and best-of-N distillation. To better exploit such samples in diffusion models, Frans et al. (2025); Physical Intelligence et al. (2025); Zheng et al. (2025) used reward or advantage functions to classify data into preference pairs, and reinforced the diffusion model with classifier-free guidance. Tang et al. (2025) leveraged a weighted behavior cloning with negative weights for diffusion language model reasoning. However, the negative reweighting function is an ad-hoc design, while we provide a generalized and unified viewpoint with theoretical justification.

B Extension of f-divergence to Signed Measures.

Let Q be a finite nonnegative measure (e.g., a probability measure) and let P be a finite signed measure. Assume the total variation of P is absolutely continuous with respect to Q, i.e., $|P| \ll Q$, so that the Radon–Nikodym derivative $r := dP/dQ$ exists Q-a.e. and may take negative values. If f is extended to a measurable function on $\mathbb{R}$ (or at least on the essential range of r) such that $f(r) \in L^1(Q)$, we define

$$D_f(P\|Q) \;:=\; \int f\left(\frac{dP}{dQ}\right)\, dQ.$$

Moreover, if Q is a probability measure, $P(\Omega) = Q(\Omega) = 1$, and the extension of f is convex on $\mathbb{R}$ with $f(1) = 0$, then Jensen's inequality yields $D_f(P\|Q) \geqslant 0$ despite r potentially being negative; under strict convexity, the usual identity-of-indiscernibles conclusion $D_f(P\|Q) = 0 \Rightarrow P = Q$ follows (up to standard regularity/equality conditions).

C Derivations of the Regularized Policy Optimization

C.1 Derivation of the f-Divergence Regularized Policy Optimization

We omit the state $\boldsymbol{s}$ in the following derivation for brevity. Let $f : \mathbb{R}^+ \to \mathbb{R}$ be strictly convex and differentiable and $f(1) = 0$. We want to find a policy $\pi \in \Pi$ that maximizes the value $Q(\boldsymbol{a})$ while controlling the f-divergence from the behavior policy π_{old}, which can be formulated as the following constrained optimization problem:

$$\begin{aligned} &\max_{\pi\in\Pi} \sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a})Q(\boldsymbol{a}) \\ &\text{s.t.} \begin{cases} D_f(\pi(\cdot)\|\pi_{\text{old}}(\cdot)) \leqslant \epsilon \\ \sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a}) = 1 \\ \forall \boldsymbol{a} \in \mathcal{A},\ \ \pi(\boldsymbol{a}) \geqslant 0 \end{cases} \end{aligned} \tag{12}$$

We introduce Lagrange multipliers λ, ν, and $\eta(\boldsymbol{a}) \geqslant 0$ for the constraints, respectively, leading to the following Lagrange function $\mathcal{L}$,

$$\mathcal{L}(\pi, \lambda, \nu, \eta) = \sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a})Q(\boldsymbol{a}) - \lambda\left(\sum_{\boldsymbol{a}\in\mathcal{A}} \pi_{\text{old}}(\boldsymbol{a}) f\left(\frac{\pi(\boldsymbol{a})}{\pi_{\text{old}}(\boldsymbol{a})}\right) - \epsilon\right) - \nu(1 - \sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a})) + \sum_{\boldsymbol{a}\in\mathcal{A}} \eta(\boldsymbol{a})\pi(\boldsymbol{a}). \tag{13}$$

We take the functional derivative with respect to $\pi(\boldsymbol{a})$ and set it to zero. Recall that the derivative of the term $\mu f(\pi/\mu)$ with respect to π is $f'(\pi/\mu)$,

$$0 = \frac{\partial \mathcal{L}}{\partial \pi(\boldsymbol{a})} = Q(\boldsymbol{a}) - \lambda f'(\frac{\pi(\boldsymbol{a})}{\pi_{\text{old}}(\boldsymbol{a})}) - \nu + \eta(\boldsymbol{a}), \tag{14}$$

Rearranging to isolate the divergence term,

$$f'\left(\frac{\pi(\boldsymbol{a})}{\pi_{\text{old}}(\boldsymbol{a})}\right) = \frac{Q(\boldsymbol{a}) - \nu + \eta(\boldsymbol{a})}{\lambda}. \tag{15}$$

We define $g_f = (f')^{-1}$. Since the complimentary slackness requires $\eta(\boldsymbol{a})\pi(\boldsymbol{a}) = 0$, we consider the following cases:

- $\pi(\boldsymbol{a}) > 0$, in this case $\eta(\boldsymbol{a}) = 0$, therefore we have

$$f'\left(\frac{\pi(\boldsymbol{a})}{\pi_{\text{old}}(\boldsymbol{a})}\right) = \frac{Q(\boldsymbol{a}) - \nu}{\lambda} \Rightarrow \frac{\pi(\boldsymbol{a})}{\pi_{\text{old}}(\boldsymbol{a})} = g_f\left(\frac{Q(\boldsymbol{a}) - \nu}{\lambda}\right)$$

  Besides, since f is strongly convex, both f' and g_f are increasing, therefore

$$\frac{Q(\boldsymbol{a}) - \nu}{\lambda} = f'(\frac{\pi(\boldsymbol{a})}{\pi_{\text{old}}(\boldsymbol{a})}) > f'(0).$$

- $\pi(\boldsymbol{a}) = 0$, in this case $\eta(\boldsymbol{a}) > 0$. From the stationary condition, we know that

$$Q(\boldsymbol{a}) - \nu + \eta(\boldsymbol{a}) = \lambda f'(0) \Rightarrow \frac{Q(\boldsymbol{a}) - \nu}{\lambda} < f'(0).$$

Therefore, for both cases, we can unify the optimal solution as

$$\pi(\boldsymbol{a}) = \pi_{\text{old}}(\boldsymbol{a})\bar{g}_f\left(\frac{Q(\boldsymbol{a}) - \nu}{\lambda}\right), \tag{16}$$

where

$$\bar{g}_f(x) = \begin{cases} g_f(x) & \text{if } x > f'(0) \\ 0 & \text{if } x \leqslant f'(0) \end{cases}. \tag{17}$$

C.2 Derivation of the f-Divergence Regularized Policy Optimization on Signed Measures

Let $f(x) : \mathbb{R} \to \mathbb{R}$ be strictly convex and differentiable and $f(1) = 0$. Since f is strictly convex, f' is monotonically increasing, and therefore $g_f = (f')^{-1}$ is well defined. Consider the solution space $\widetilde{\Pi} = \{\pi : \sum_{\boldsymbol{a}\sim\mathcal{A}} \pi(\boldsymbol{a}) = 1\}$, the optimization problem regularized the generalized f-divergence is defined as

$$\begin{aligned} &\max_{\pi\in\widetilde{\Pi}} \sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a})Q(\boldsymbol{a}) \\ &\text{s.t.} \begin{cases} D_f(\pi(\cdot)\|\pi_{\text{old}}(\cdot)) \leqslant \epsilon \\ \sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a}) = 1 \end{cases} \end{aligned} \tag{18}$$

We introduce Lagrange multipliers λ and ν for the constraints, leading to the following Lagrange function $\mathcal{L}$:

$$\mathcal{L}(\pi, \lambda, \nu) = \sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a})Q(\boldsymbol{a}) - \lambda\left(\sum_{\boldsymbol{a}\in\mathcal{A}} \pi_{\text{old}}(\boldsymbol{a}) f\left(\frac{\pi(\boldsymbol{a})}{\pi_{\text{old}}(\boldsymbol{a})}\right) - \epsilon\right) - \nu(1 - \sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a})). \tag{19}$$

Taking its derivative w.r.t. $\pi(\boldsymbol{a})$ and setting it to zero:

$$0 = \frac{\partial\mathcal{L}}{\partial\pi(\boldsymbol{a})} = Q(\boldsymbol{a}) - \lambda f'\left(\frac{\pi(\boldsymbol{a})}{\pi_{\text{old}}(\boldsymbol{a})}\right) - \nu, \tag{20}$$

i.e.,

$$\pi(\boldsymbol{a}) = \pi_{\text{old}}(\boldsymbol{a})(f')^{-1}\left(\frac{Q(\boldsymbol{a}) - \nu}{\lambda}\right) = \pi_{\text{old}}(\boldsymbol{a})g_f\left(\frac{Q(\boldsymbol{a}) - \nu}{\lambda}\right).$$

Note that ν is the normalization term such that $\sum_{\boldsymbol{a}\in\mathcal{A}} \mu(\boldsymbol{a})g\left((Q(\boldsymbol{a}) - \nu)/\lambda\right) = 1$.

Generally speaking, for any strictly increasing (and typically continuous) function g defined on an interval $I \subseteq \mathbb{R}$, its inverse g^{-1} exists on $g(I)$ and is strictly increasing. Define $f'_g(x) = g^{-1}(x)$ for $x \in g(I)$ and

$$f_g(x) = \int_{\boldsymbol{a}_0}^{x} g^{-1}(t)\mathrm{d}t + C$$

yields a differentiable strictly convex function on $g(I)$. Normalizing $f_g(1) = 0$ underpins C. While the analytic formulation of f_g may not be available for an arbitrary g, this observation allows us to reason directly in terms of the weighting function g rather than f_g.

C.3 Existence of the Generator Function f for Non-decreasing Weighting Function g

Instead of requiring f to be differentiable and strictly convex, the above argument can also be extended to proper, closed, and convex f. The Lagrangian can be written as

$$\mathcal{L}(\pi, \lambda, \nu, \eta) = \sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a})Q(\boldsymbol{a}) - \lambda\left(\sum_{\boldsymbol{a}\in\mathcal{A}} \pi_{\text{old}}(\boldsymbol{a}) f\left(\frac{\pi(\boldsymbol{a})}{\pi_{\text{old}}(\boldsymbol{a})}\right) - \epsilon\right) - \nu(1 - \sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a})). \tag{21}$$

The stationary condition is now a subgradient inclusion:

$$0 \in Q(\boldsymbol{a}) - \nu - \lambda\partial f(\frac{\pi(\boldsymbol{a})}{\pi_{\text{old}}(\boldsymbol{a})}) \Rightarrow \frac{Q(\boldsymbol{a}) - \nu}{\lambda} \in \partial f(\frac{\pi(\boldsymbol{a})}{\pi_{\text{old}}(\boldsymbol{a})}), \tag{22}$$

For a continuous and non-decreasing weighting function g such that $\pi(\boldsymbol{a}) = \pi_{\text{old}}(\boldsymbol{a})g(\frac{Q(\boldsymbol{a})-\nu}{\lambda})$, we can define

$$\begin{aligned} G(x) &= \int_0^x g(y)\mathrm{d}y, \\ f_g(x) &= G^*(x) - G^*(1), \end{aligned} \tag{23}$$

where G^* is the convex conjugate of G. In this case,

$$\frac{\pi(\boldsymbol{a})}{\pi_{\text{old}}(\boldsymbol{a})} = g(\frac{Q(\boldsymbol{a}) - \nu}{\lambda}) \in \partial f_g^*(\frac{Q(\boldsymbol{a}) - \nu}{\lambda}), \tag{24}$$

where ∂f_g^* denotes the subgradient of the convex conjugate f_g^*. By conjugate identity,

$$\frac{\pi(\boldsymbol{a})}{\pi_{\text{old}}(\boldsymbol{a})} \in \partial f_g^*(\frac{Q(\boldsymbol{a}) - \nu}{\lambda}) \Leftrightarrow \frac{Q(\boldsymbol{a}) - \nu}{\lambda} \in \partial f_g(\frac{\pi(\boldsymbol{a})}{\pi_{\text{old}}(\boldsymbol{a})}), \tag{25}$$

i.e., it satisfies the stationary condition.

Therefore, for general non-decreasing weighting function g like $g(\cdot) = [\cdot]_+$, there exists a divergence generator function f_g such that weighting by g solves the corresponding regularized optimization problem.

C.4 Policy Improvement Property of the Target Measure

Finally, we show that even without non-negativity constraints enforced, the target policy still improves over the current policy π_{old} as long as it is normalized.

Theorem C.1. *Assume $g : \mathbb{R} \to \mathbb{R}$ is non-decreasing and the target measure $\tilde{\pi}$ is defined as*

$$\tilde{\pi}(\boldsymbol{a}|\boldsymbol{s}) = \pi(\boldsymbol{a}|\boldsymbol{s})g\left(\frac{Q(\boldsymbol{s},\boldsymbol{a}) - \nu}{\lambda}\right) \quad \forall \boldsymbol{s} \in \mathcal{S},$$

where $\lambda > 0$ and ν is selected such that $\sum_{\boldsymbol{a}\in\mathcal{A}} \tilde{\pi}(\boldsymbol{a}|\boldsymbol{s}) = 1$ holds. Then we have

$$\sum_{\boldsymbol{a}\in\mathcal{A}} \tilde{\pi}(\boldsymbol{a}|\boldsymbol{s})Q(\boldsymbol{s},\boldsymbol{a}) \geqslant \sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a}|\boldsymbol{s})Q(\boldsymbol{s},\boldsymbol{a}) \quad \forall \boldsymbol{s} \in \mathcal{S}.$$

Proof. Denote $A(\boldsymbol{s},\boldsymbol{a}) = \frac{Q(\boldsymbol{s},\boldsymbol{a})-\nu}{\lambda}$, and thus we have $\sum_{\boldsymbol{a}\in\mathcal{A}} \tilde{\pi}(\boldsymbol{a}|\boldsymbol{s}) = \mathbb{E}_\pi[g\left(A(\boldsymbol{s},\boldsymbol{a})\right] = 1$. It suffices to show

$$\sum_{\boldsymbol{a}\in\mathcal{A}} \tilde{\pi}(\boldsymbol{a}|\boldsymbol{s})A(\boldsymbol{s},\boldsymbol{a}) \geqslant \sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a}|\boldsymbol{s})A(\boldsymbol{s},\boldsymbol{a}).$$

Consider the difference

$$\begin{aligned} &\sum_{\boldsymbol{a}\in\mathcal{A}} \tilde{\pi}(\boldsymbol{a}|\boldsymbol{s})A(\boldsymbol{s},\boldsymbol{a}) - \sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a}|\boldsymbol{s})A(\boldsymbol{s},\boldsymbol{a}) \\ &= \sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a}|\boldsymbol{s})A(\boldsymbol{s},\boldsymbol{a})g\left(A(\boldsymbol{s},\boldsymbol{a})\right) - \left(\sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a}|\boldsymbol{s})A(\boldsymbol{s},\boldsymbol{a})\right)\left(\sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a}|\boldsymbol{s})g\left(A(\boldsymbol{s},\boldsymbol{a})\right)\right) \\ &= \mathbb{E}_\pi[A(\boldsymbol{s},\boldsymbol{a})g\left(A(\boldsymbol{s},\boldsymbol{a})\right)] - \mathbb{E}_\pi[A(\boldsymbol{s},\boldsymbol{a})]\mathbb{E}_\pi[g\left(A(\boldsymbol{s},\boldsymbol{a})\right)] \\ &= \mathrm{Cov}_\pi[A(\boldsymbol{s},\boldsymbol{a}), g\left(A(\boldsymbol{s},\boldsymbol{a})\right)]. \end{aligned}$$

Since g is non-decreasing, we know that $\mathrm{Cov}_\pi[A(\boldsymbol{s},\boldsymbol{a}), g\left(A(\boldsymbol{s},\boldsymbol{a})\right)] \geqslant 0$, and thus

$$\sum_{\boldsymbol{a}\in\mathcal{A}} \tilde{\pi}(\boldsymbol{a}|\boldsymbol{s})A(\boldsymbol{s},\boldsymbol{a}) \geqslant \sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a}|\boldsymbol{s})A(\boldsymbol{s},\boldsymbol{a}).$$

□

D Derivation of Closed-Form Solution of Reweighted Conditional Flow Matching.

We temporarily omit conditioning on the state $\boldsymbol{s}$ for clarity.

D.1 Closed-Form Solution of Conditional Flow Matching

We first establish the optimal solution for standard conditional flow matching without a reweighting factor $w(\boldsymbol{x}_0)$.

Lemma D.1 (Conditional flow matching fits marginal velocity Lipman et al. (2022))**.** *The conditional flow matching (CFM) loss is defined as:*

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}[0,1],\ \boldsymbol{x}_0\sim p_{data}(\boldsymbol{x}_0),\ \boldsymbol{x}_t\sim q_{t|0}(\cdot|\boldsymbol{x}_0)} \left[\|D_\theta(\boldsymbol{x}_t,t) - \boldsymbol{v}_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)\|^2\right].$$

Upon convergence, the $D_{\theta^*}(\boldsymbol{x_t},t)$ fits the marginal velocity $\boldsymbol{v}_t(\boldsymbol{x}_t)$.

Proof. Expanding the expectation into integral form and applying Bayes' theorem to reverse the joint distribution, $p_{\text{data}}(\boldsymbol{x}_0)q_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0) = q_t(\boldsymbol{x}_t)q_{0|t}(\boldsymbol{x}_0|\boldsymbol{x}_t)$, we can rewrite the loss as:

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \int_0^1 \int q_t(\boldsymbol{x}_t) \left[\int q_{0|t}(\boldsymbol{x}_0|\boldsymbol{x}_t)\|D_\theta(\boldsymbol{x}_t,t) - \boldsymbol{v}_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)\|^2\, d\boldsymbol{x}_0\right] d\boldsymbol{x}_t\, dt$$

Because $q_t(\boldsymbol{x}_t)$ and dt are non-negative, minimizing the global integral is equivalent to minimizing the inner objective pointwise for any fixed $\boldsymbol{x}_t$ and t. Let $v = D_\theta(\boldsymbol{x}_t,t)$. The local objective becomes:

$$J(v) = \int q_{0|t}(\boldsymbol{x}_0|\boldsymbol{x}_t)\|v - \boldsymbol{v}_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)\|^2\, d\boldsymbol{x}_0$$

Setting the gradient with respect to v to zero yields:

$$\nabla_v J(v) = 2\int q_{0|t}(\boldsymbol{x}_0|\boldsymbol{x}_t)(v - \boldsymbol{v}_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0))\, d\boldsymbol{x}_0 = 0$$

Since v is independent of $\boldsymbol{x}_0$ and the posterior integrates to 1 ($\int q_{0|t}(\boldsymbol{x}_0|\boldsymbol{x}_t)d\boldsymbol{x}_0 = 1$), solving for v provides the optimal vector field:

$$v^* = D_{\theta^*}(\boldsymbol{x}_t,t) = \mathbb{E}_{\boldsymbol{x}_0\sim q_{0|t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}[\boldsymbol{v}_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)]$$

By definition, this matches the marginal velocity field $\boldsymbol{v}_t(\boldsymbol{x}_t) \doteq \mathbb{E}_{\boldsymbol{x}_0\sim q_{0|t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}[\boldsymbol{v}_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)]$, concluding the proof. □

D.2 Closed-Form Solution of Reweighted Conditional Flow Matching

We now present our primary result regarding reweighted conditional flow matching.

Theorem D.2 (Optimal solution of reweighted conditional flow matching)**.** *When $\mathbb{E}_{\boldsymbol{x}_0\sim q_{0|t}}[w(\boldsymbol{x}_0)] > 0$, the reweighted conditional flow matching loss*

$$\mathcal{L}_W(\theta) = \mathbb{E}_{t\sim\mathcal{U},\ \boldsymbol{x}_0\sim p_{data}(\boldsymbol{x}_0),\ \boldsymbol{x}_t\sim q_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)} \left[w(\boldsymbol{x}_0)\|D_\theta(\boldsymbol{x}_t,t) - \boldsymbol{v}_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)\|^2\right]$$

induces the optimal solution:

$$D_{\theta^*}(\boldsymbol{x}_t;t) = \frac{\mathbb{E}_{\boldsymbol{x}_0\sim q_{0|t}}[w(\boldsymbol{x}_0)\boldsymbol{v}_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)]}{\mathbb{E}_{\boldsymbol{x}_0\sim q_{0|t}}[w(\boldsymbol{x}_0)]}$$

Proof. Let $v = D_\theta(\boldsymbol{x}_t, t)$. Expanding the expectation and substituting the marginal-posterior decomposition $q_t(\boldsymbol{x}_t)q_{0|t}(\boldsymbol{x}_0|\boldsymbol{x}_t)$, we isolate the pointwise loss for a fixed $\boldsymbol{x}_t$ and t:

$$J_W(v) = \int q_{0|t}(\boldsymbol{x}_0|\boldsymbol{x}_t)w(\boldsymbol{x}_0)\|v - \boldsymbol{v}_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)\|^2 \, d\boldsymbol{x}_0$$

Expanding the quadratic term, we obtain:

$$J_W(v) = \|v\|^2 \underbrace{\left(\int q_{0|t}(\boldsymbol{x}_0|\boldsymbol{x}_t)w(\boldsymbol{x}_0)\, d\boldsymbol{x}_0\right)}_{M(\boldsymbol{x}_t)} - 2v^T \underbrace{\left(\int q_{0|t}(\boldsymbol{x}_0|\boldsymbol{x}_t)w(\boldsymbol{x}_0)\boldsymbol{v}_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)\, d\boldsymbol{x}_0\right)}_{N(\boldsymbol{x}_t)} + C \tag{26}$$

Setting the gradient to zero yields $2vM(\boldsymbol{x}_t) - 2N(\boldsymbol{x}_t) = 0$. Unlike the unweighted case, the integral $M(\boldsymbol{x}_t)$ acts as a normalizing constant for the weights rather than summing to 1. Provided $M(\boldsymbol{x}_t) > 0$, we can solve for v to find the optimal closed-form solution:

$$v^* = D_{\theta^*}(\boldsymbol{x}_t, t) = \frac{N(\boldsymbol{x}_t)}{M(\boldsymbol{x}_t)} = \frac{\mathbb{E}_{\boldsymbol{x}_0 \sim q_{0|t}}[w(\boldsymbol{x}_0)\boldsymbol{v}_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)]}{\mathbb{E}_{\boldsymbol{x}_0 \sim q_{0|t}}[w(\boldsymbol{x}_0)]}$$

□

D.3 Velocity Field Generates Reweighted Data Distribution

Finally, we prove that D_{θ^*} generates the probability path:

$$\rho_t(\boldsymbol{x}_t|\boldsymbol{s}) = \frac{1}{Z}\int w(\boldsymbol{s}, \boldsymbol{x}_0)p_{\text{data}}(\boldsymbol{x}_0|\boldsymbol{s})q_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)\, d\boldsymbol{x}_0$$

where $Z(\boldsymbol{s}) = \int w(\boldsymbol{s}, \boldsymbol{x}_0)p_{\text{data}}(\boldsymbol{x}_0|\boldsymbol{s})\, d\boldsymbol{x}_0$ is the global normalizing constant.

Proof. To prove $D_{\theta^*}(\boldsymbol{x}_t, t)$ generates this distribution, we verify that it satisfies the continuity equation: $\frac{\partial \rho_t}{\partial t} + \nabla \cdot (\rho_t D_{\theta^*}) = 0$. Let $\tilde{\rho}_t(\boldsymbol{x}_t) = Z\rho_t(\boldsymbol{x}_t)$ denote the unnormalized density. The flow term is:

$$\tilde{\rho}_t D_{\theta^*} = \int w(\boldsymbol{x}_0)p_{\text{data}}(\boldsymbol{x}_0)q_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)\boldsymbol{v}_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)\, d\boldsymbol{x}_0$$

Applying the divergence operator and substituting the conditional continuity equation $\nabla \cdot (q_{t|0}\boldsymbol{v}_{t|0}) = -\frac{\partial q_{t|0}}{\partial t}$, we find:

$$\nabla \cdot (\tilde{\rho}_t D_{\theta^*}) = \int w(\boldsymbol{x}_0)p_{\text{data}}(\boldsymbol{x}_0)\left(-\frac{\partial q_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)}{\partial t}\right) d\boldsymbol{x}_0$$

By the linearity of the derivative, we extract the time derivative:

$$\nabla \cdot (\tilde{\rho}_t D_{\theta^*}) = -\frac{\partial}{\partial t}\int w(\boldsymbol{x}_0)p_{\text{data}}(\boldsymbol{x}_0)q_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0)\, d\boldsymbol{x}_0 = -\frac{\partial \tilde{\rho}_t}{\partial t}$$

Thus, $\frac{\partial \tilde{\rho}_t}{\partial t} + \nabla \cdot (\tilde{\rho}_t v_t^*) = 0$, confirming that the vector field properly evolves ρ_0 to ρ_1. □

E Population-based Estimator for the Normalization Term in Linear and Square Reweighting

In this section, we detail on how to obtain the normalization term $\nu(\boldsymbol{s})$ when using linear and square reweighting.

E.1 General Setup

We omit the dependence on states $\boldsymbol{s}$. We assume we have access to a batch of samples $\{\boldsymbol{a}_{(i)}\}_{i=1}^N$, and their Q-values are $\{Q(\boldsymbol{a}_{(i)})\}_{i=1}^N$ accordingly. Denote $x_{(i)} = Q(\boldsymbol{a}_{(i)})$ for brevity, and we assume $\{x_{(i)}\}_{i=1}^N$ are sorted in descending order such that $x_{(1)} \geqslant x_{(2)} \geqslant \ldots \geqslant x_{(N)}$.

We define the *active set* as $S = \{i \mid x_i > \nu\}$, i.e. the index set of samples which will not be clipped to zero. Let $k = |S|$ denote the number of active samples. For $i \in S$, $\bar{g}_f(x_i - \nu) = x_i - \nu$; otherwise, it is 0 according to linear and square reweighting.

High-level Idea. Because the $\max(\cdot, 0)$ operation truncates values below the threshold ν to zero, only the top k elements in our sorted sequence will actively contribute to the target sum. The core challenge is determining this exact number of active elements, k, without knowing ν beforehand.

We achieve this by testing each possible $k \in \{1, \ldots, N\}$ and calculating an "excess" metric. This metric measures the cumulative difference between the top k elements and the candidate k-th element, $x_{(k)}$. By checking if this excess falls within our target budget (which is exactly 1), we can identify the true number of active samples. Once k is determined, the threshold ν can be analytically solved using the sums of those active elements.

E.2 Case I: Linear Reweighting

For the linear case, the normalization constraint requires that the sum of the active clipped values equals 1:

$$\sum_{i=1}^{k}(x_{(i)} - \nu) = 1$$

Assuming we know k, we can distribute the sum and solve directly for ν:

$$\sum_{i=1}^{k} x_{(i)} - k\nu = 1 \implies \nu = \frac{1}{k}\left(\sum_{i=1}^{k} x_{(i)} - 1\right)$$

To determine k, we use the definition of the active set: for the k-th element to be active, we must have $x_{(k)} > \nu$. Substituting our expression for ν into this inequality yields:

$$x_{(k)} > \frac{1}{k}\left(\sum_{i=1}^{k} x_{(i)} - 1\right)$$

Rearranging the terms gives us our condition:

$$kx_{(k)} > \sum_{i=1}^{k} x_{(i)} - 1 \implies \sum_{i=1}^{k} x_{(i)} - kx_{(k)} < 1$$

The term $\sum_{i=1}^{k} x_{(i)} - kx_{(k)}$ is the "excess" if the threshold were set exactly at $x_{(k)}$. We compute this excess for all possible indices using the cumulative sum of our sorted values. The true number of active particles k is simply the count of indices where this excess is less than or equal to 1. After identifying k, we plug it back into our initial equation to find the final normalization term ν.

E.3 Case II: Square Reweighting

For the square case, the constraint requires the sum of the squared active values to equal 1:

$$\sum_{i=1}^{k}(x_{(i)} - \nu)^2 = 1$$

Expanding the square and distributing the sum gives a quadratic equation in terms of ν:

$$k\nu^2 - 2\left(\sum_{i=1}^{k} x_{(i)}\right)\nu + \left(\sum_{i=1}^{k} x_{(i)}^2 - 1\right) = 0$$

Let $S_1 = \sum_{i=1}^{k} x_{(i)}$ and $S_2 = \sum_{i=1}^{k} x_{(i)}^2$. The equation simplifies to $k\nu^2 - 2S_1\nu + (S_2 - 1) = 0$. Using the quadratic formula and taking the smaller root (to guarantee that ν remains strictly less than the active elements), we obtain:

$$\nu = \frac{S_1 - \sqrt{S_1^2 - kS_2 + k}}{k}$$

Similar to the linear case, we must first find k. The condition $x_{(k)} > \nu$ is algebraically equivalent to requiring that the sum of squared differences from $x_{(k)}$ is less than or equal to our budget of 1:

$$\sum_{i=1}^{k}(x_{(i)} - x_{(k)})^2 \leqslant 1$$

Expanding this excess metric gives:

$$\sum_{i=1}^{k} x_{(i)}^2 - 2x_{(k)} \sum_{i=1}^{k} x_{(i)} + kx_{(k)}^2 \leqslant 1$$

$$S_2 - 2x_{(k)} S_1 + kx_{(k)}^2 \leqslant 1$$

We evaluate this condition for all indices using the cumulative sums of both x and x^2. The active set size k is the number of elements that satisfy this inequality. Finally, we use this k, along with its corresponding S_1 and S_2, to compute the discriminant $\Delta = S_1^2 - kS_2 + k$ and solve for ν.

E.4 Reference Implementation

The above calculation can be fully vectorized in our JAX implementation. In the below, we provide a vectorized JAX implementation for solving the normalization factor for both the linear and square reweighting.

Listing 1: JAX implementation for solving ν.

```python
def solve_normalizer_linear(q: jnp.ndarray, temp: float):
    num_particles = q.shape[-1]
    q_sorted = jnp.sort(q, axis=-1)[..., ::-1]
    q_cumsum = jnp.cumsum(q_sorted, axis=-1)

    k_indices = jnp.arange(1, num_particles+1).reshape(1, -1)
    excess = q_cumsum - k_indices * q_sorted
    active_mask = excess <= temp
    k = jnp.sum(active_mask, axis=-1, keepdims=True)

    sum_active = jnp.take_along_axis(q_cumsum, k-1, axis=-1)
    nu = (sum_active - temp) / k
    return nu

def solve_normalizer_square(q: jnp.ndarray, temp: float):
    q_max = jnp.max(q, axis=-1, keepdims=True)
    q = q - q_max
    num_particles = q.shape[-1]
    target_sum = temp ** 2

    q_sorted = jnp.sort(q, axis=-1)[..., ::-1]
    q_cumsum = jnp.cumsum(q_sorted, axis=-1)
    q_squared_cumsum = jnp.cumsum(q_sorted ** 2, axis=-1)

    k_indices = jnp.arange(1, num_particles+1).reshape(1, -1)
    excess = q_squared_cumsum \
            - 2 * q_cumsum * q_sorted \
            + k_indices * (q_sorted ** 2)
    active_mask = excess <= target_sum
    k = jnp.maximum(jnp.sum(active_mask, axis=-1, keepdims=True), 1)

    S1 = jnp.take_along_axis(q_cumsum, k-1, axis=-1)
    S2 = jnp.take_along_axis(q_squared_cumsum, k-1, axis=-1)
    delta = S1**2 - k * S2 + k * target_sum
    nu = (S1 - jnp.sqrt(jnp.maximum(delta, 0.0))) / k
    return nu + q_max
```

F Auto-tuning the Temperature

The temperature λ in GeMPO controls the sharpness of the reweighting distribution over the N candidate actions $\{a_i\}_{i=1}^N$ drawn from the old policy. Its appropriate scale is tightly coupled with the magnitude of the Q-values, which drifts substantially during training, making a fixed λ brittle. We therefore adapt λ online by constraining the divergence between the reweighting distribution

and the uniform distribution over particles. Let $w_i(\lambda)$ denote the normalized per-particle weights produced by the chosen reweighting scheme (exponential, linear, or squared), so that $\sum_i w_i = 1$. Its Shannon entropy is $\mathcal{H}(w) = -\sum_i w_i \log w_i$, and its KL divergence to the uniform distribution Unif_N satisfies

$$\mathrm{KL}(w \,\|\, \mathrm{Unif}_N) \;=\; \log N - \mathcal{H}(w). \tag{27}$$

Following the dual formulation of SAC (Haarnoja et al., 2018), we treat λ as the Lagrange multiplier of the soft constraint $\mathrm{KL}(w \,\|\, \mathrm{Unif}_N) \leqslant \kappa$, where κ is a target KL budget. Dual ascent yields the surrogate temperature loss

$$\mathcal{L}_\lambda \;=\; \lambda \cdot \big(\kappa + \mathcal{H}(w) - \log N\big) \;=\; \lambda \cdot \big(\kappa - \mathrm{KL}(w \,\|\, \mathrm{Unif}_N)\big), \tag{28}$$

minimized w.r.t. λ jointly with the actor and critic. Intuitively, when the reweighting is too peaky ($\mathrm{KL} > \kappa$) the gradient drives λ upward, smoothing the weights; when it is too diffuse ($\mathrm{KL} < \kappa$) it drives λ downward, sharpening them. Positivity of λ is enforced via a softplus parameterization.

G Matching Policy with the Target Measure via Noise Contrastive Estimation

As an alternative to reweighted flow matching, we can align the current diffusion policy π_θ with the target policy using Noise Contrastive Estimation (NCE). NCE frames density estimation as a classification problem, distinguishing between *positive* samples from the target policy and *negative* samples from a noise proposal distribution. In our setting, the target policy is $\pi^*(\boldsymbol{a}_0|\boldsymbol{s}) \propto \pi_{\mathrm{old}}(\boldsymbol{a}_0|\boldsymbol{s}) w(\boldsymbol{s}, \boldsymbol{a}_0)$, where $w(\boldsymbol{s}, \boldsymbol{a}_0) = g_f\left(\frac{Q(\boldsymbol{s},\boldsymbol{a}_0) - \nu(\boldsymbol{s})}{\lambda}\right)$ is the reweighting factor. We naturally select the old policy $\pi_{\mathrm{old}}(\boldsymbol{a}_0|\boldsymbol{s})$ as our noise proposal distribution.

Consider a set of M actions $\{\boldsymbol{a}_0^{(1)}, \boldsymbol{a}_0^{(2)}, \ldots, \boldsymbol{a}_0^{(M)}\}$ sampled independently from the old policy $\pi_{\mathrm{old}}(\cdot|\boldsymbol{s})$ for a given state $\boldsymbol{s}$. Under the target policy, the categorical probability of selecting a specific action $\boldsymbol{a}_0^{(i)}$ from this set is independent of the global partition function $Z(\boldsymbol{s})$, as it cancels out in the normalization:

$$p^*(\boldsymbol{a}_0^{(i)} | \{\boldsymbol{a}_0^{(j)}\}_{j=1}^M, \boldsymbol{s}) = \frac{\pi^*(\boldsymbol{a}_0^{(i)}|\boldsymbol{s}) / \pi_{\mathrm{old}}(\boldsymbol{a}_0^{(i)}|\boldsymbol{s})}{\sum_{j=1}^M \pi^*(\boldsymbol{a}_0^{(j)}|\boldsymbol{s}) / \pi_{\mathrm{old}}(\boldsymbol{a}_0^{(j)}|\boldsymbol{s})} = \frac{w(\boldsymbol{s}, \boldsymbol{a}_0^{(i)})}{\sum_{j=1}^M w(\boldsymbol{s}, \boldsymbol{a}_0^{(j)})}$$

We construct the categorical probability of select action $\boldsymbol{a}_0^{(i)}$ from this set as:

$$p_\theta(\boldsymbol{a}_0^{(i)} | \{\boldsymbol{a}_0^{(j)}\}_{j=1}^M, \boldsymbol{s}) = \frac{\exp\left(-\frac{1}{\gamma}\mathbb{E}_{t,\boldsymbol{a}_t}[\|D_\theta(\boldsymbol{s}, \boldsymbol{a}_t, t) - \boldsymbol{v}_{t|0}(\boldsymbol{a}_t|\boldsymbol{a}_0^{(i)})\|^2]\right)}{\sum_{j=1}^M \exp\left(-\frac{1}{\gamma}\mathbb{E}_{t,\boldsymbol{a}_t}[\|D_\theta(\boldsymbol{s}, \boldsymbol{a}_t, t) - \boldsymbol{v}_{t|0}(\boldsymbol{a}_t|\boldsymbol{a}_0^{(j)})\|^2]\right)}$$

To align the diffusion policy π_θ with the target measure, we minimize the cross-entropy between the target categorical distribution p^* and the current policy's categorical distribution p_θ over the sampled

set:

$$\begin{aligned}
\min_\theta \mathcal{L}(\theta) &= -\mathbb{E}_{\boldsymbol{s},\boldsymbol{a}_0^{(1):(M)}}\left[\sum_{i=1}^{M} p^*(\boldsymbol{a}_0^{(i)}|\{\boldsymbol{a}_0^{(j)}\}_{j=1}^M,\boldsymbol{s})\log p_\theta(\boldsymbol{a}_0^{(i)}|\{\boldsymbol{a}_0^{(j)}\}_{j=1}^M,\boldsymbol{s})\right] \\
&= \mathbb{E}_{\boldsymbol{s},\boldsymbol{a}_0^{(1):(M)}}\left[\sum_{i=1}^{M}\frac{w(\boldsymbol{s},\boldsymbol{a}_0^{(i)})}{\sum_j w(\boldsymbol{s},\boldsymbol{a}_0^{(j)})}\frac{1}{\gamma}\mathbb{E}_{t,\boldsymbol{a}_t}[\|D_\theta(\boldsymbol{s},\boldsymbol{a}_t,t)-\boldsymbol{v}_{t|0}(\boldsymbol{a}_t|\boldsymbol{a}_0^{(i)})\|^2]\right. \\
&\qquad \left. + \log\sum_{j=1}^{M}\exp\left(-\frac{1}{\gamma}\mathbb{E}_{t,\boldsymbol{a}_t}[\|D_\theta(\boldsymbol{s},\boldsymbol{a}_t,t)-\boldsymbol{v}_{t|0}(\boldsymbol{a}_t|\boldsymbol{a}_0^{(j)})\|^2]\right)\right] \\
&\leqslant \mathbb{E}_{\boldsymbol{s},\boldsymbol{a}_0^{(1):(M)},\boldsymbol{a}_t^{(1):(M)}}\left[\underbrace{\sum_{i=1}^{M}\frac{w(\boldsymbol{s},\boldsymbol{a}_0^{(i)})}{\sum_j w(\boldsymbol{s},\boldsymbol{a}_0^{(j)})}\frac{1}{\gamma}\|D_\theta(\boldsymbol{s},\boldsymbol{a}_t^{(i)},t)-\boldsymbol{v}_{t|0}(\boldsymbol{a}_t^{(i)}|\boldsymbol{a}_0^{(i)})\|^2}_{\text{positive part}}\right. \\
&\qquad \left. \underbrace{+\log\sum_{j=1}^{M}\exp\left(-\frac{1}{\gamma}\|D_\theta(\boldsymbol{s},\boldsymbol{a}_t^{(j)},t)-\boldsymbol{v}_{t|0}(\boldsymbol{a}_t^{(j)}|\boldsymbol{a}_0^{(j)})\|^2\right)}_{\text{negative part}}\right],
\end{aligned}$$

where the final inequality follows from Jensen's inequality and the convexity of the LogSumExp function. The resulting objective consists of two components: (1) a weighted flow matching loss that minimizes the flow matching error, thereby increasing the likelihood of the policy generating high-value actions; and (2) a negative part that penalizes the likelihood of alternative samples, effectively pushing the policy's density away from undesired regions of the action space.

H Additional Experiment Results

H.1 Sweep of Reweighting Functions and Reward Curvature

To see the effect of reward functions under different landscapes in locomotion tasks, we handcrafted different reward functions to change the landscape in two DeepMind Control Suite tasks Tassa et al. (2018), Cheetah-Run and Walker-Run, shown in Figure 5. The task is to run as fast as possible, where rewards are monotonically increasing functions of robot speed. Four different rewards are ordered from flat to greedy as *sqrt*, *linear*, *square*, *exp*. The detailed definition of these reward functions is deferred to Appendix I.2. The results are shown Figure 6, which shows that `Square` reweighting excels at flat reward functions such as *sqrt* and *linear*, while `Linear` reweighting has better performance on steep reward functions, such as *exp*. The finding is consistent with our observations in the bandit problem. Therefore, we can choose `Linear` reweighting for steep reward landscapes and `Square` reweighting for relatively flat reward landscapes. Note that this is only qualitative intuition rather than saying one reweighting function must be better than another.

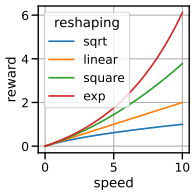


Figure 5: Reward functions.

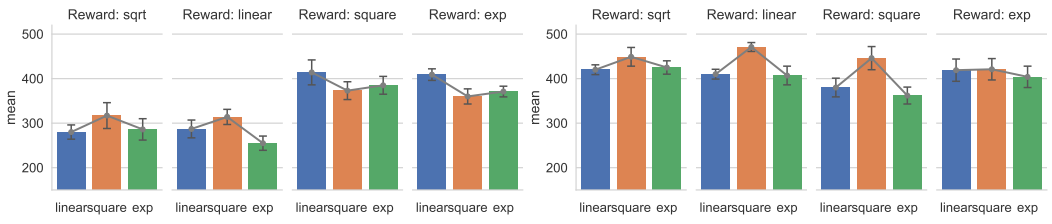


Figure 6: Performance with different reweighting and reward functions on the Cheetah Run (left) and Walker-Run (right) task. Boxes and error bars show mean returns and standard error over 5 seeds at the last checkpoint.

I EXPERIMENTAL DETAILS

I.1 BASELINES

We evaluate our approach against two distinct families of reinforcement learning (RL) algorithms. The first group consists of five state-of-the-art **online diffusion-policy methods**, encompassing both off-policy and on-policy architectures:

- **QSM** Psenka et al. (2023): Implements Langevin dynamics using the gradient of a learned Q-function as the score function.
- **QVPO** Ding et al. (2024): Employs a Q-weighted variational objective for diffusion training, though it is notably constrained by its inability to process negative rewards effectively.
- **DACER** Wang et al. (2024b): Backpropagates gradients directly through the reverse diffusion process and utilizes a GMM entropy regulator to manage the exploration-exploitation trade-off.
- **DIPO** Yang et al. (2023): Uses a two-stage approach where state-action particles are updated via Q-gradients before being fitted to a diffusion model.

The second group includes three **classic model-free baselines**: **TD3** Fujimoto et al. (2018), and **SAC** Haarnoja et al. (2018).

Table 5: Hyperparameters

Name	Value
Critic learning rate	3e-4
Policy learning rate	1e-4
Diffusion steps	20
Diffusion noise schedules	Cosine
Policy network hidden layers	2
Policy network hidden neurons	256
Policy network activation	Mish
Value network hidden layers	2
Value network hidden neurons	256
Value network activation	ReLU
replay buffer size (off-policy only)	1 million
Number of sampled particles	64
Additive Noise	0.2
Diffusion Steps	20
target KL budget κ - Hopper-v5	2.5
target KL budget κ - Ant-v5	1.5
target KL budget κ - Other tasks	2.0

For `GeMPO-Linear`, `GeMPO-Square` and `GeMPO-Exp`, we implement the automatic temperature tuning detailed in Appendix F, and sweep the KL tolerance within $\{1.5, 2.0, 2.5\}$.

I.2 REWARD FUNCTIONS

The Cheetah domain tasks are designed to track a target horizontal velocity $v^* = 10.0$ m/s (limited by the physical capability, the robot cannot actually achieve 10m/s. Therefore, the task is to run as fast as possible.). Let v denote the current horizontal velocity of the torso. We evaluate our method across several reward formulations, categorized into linear, quadratic, and absolute value variations.

We design the following reward functions (with proper transformation to align the scale), ranked from "flat" landscape to "steep" landscape,

- `Sqrt` reward $R_{abs_sqrt}(v) = \sqrt{|v - v^*|}$
- `Linear` reward $R_{\text{linear}}(v) = \frac{v}{v^*}$

- `Square` reward $R_{abs_sq}(v) = |v - v^*|^2$
- `Exponental` reward $R_{abs_exp}(v) = \exp(-|v - v^*|)$

I.3 Compute Resources and Runtime

We run our experiments on a workstation with 8 RTX 5090s. On our workstation, a single run of bandit problem takes around 5 minutes, a single run of MuJoCo/DMC task takes around 2 hours, while a single run of DNA model fine-tuning takes around 35 minutes (estimation might be effected by other users' activities). No excessive computational capabilities are required.

J Detailed Explaination of the Repelling Effects of Negative Reweighting

We provide a detailed explaination about the repelling effect here.

Compared to original optimal solution with non-negative weights in Equation (9), there are two cases if we consider weights being possibly negative, depending on the sign of denominator $\mathbb{E}_{\boldsymbol{a}_0 \sim p_{0|t}}[w(\boldsymbol{s}, \boldsymbol{a}_0)]$:

Case I (discussed in Section 3.3): For all $\boldsymbol{a}_t$ with $\mathbb{E}_{\boldsymbol{a}_0 \sim p_{0|t}}[w(\boldsymbol{s}, \boldsymbol{a}_0)] > 0$, Equation (9) still servers as the optimal solution to (8). However, the negative weights will change the sign of $\boldsymbol{v}_{t|0}$ in the weighted average, pushing the weighted average velocity away from those $\boldsymbol{v}_{t|0}$ with negative weights.

Remark J.1 (Another interpretation from generation of reweighted distributions via signed measures). Even when the weights $w(\boldsymbol{s}, \boldsymbol{a}_0)$ take negative values (as in **Case I**), the closed-form velocity field solution from Equation (9) generates a valid probability path ρ_t, provided the total measure remains positive. The generated path is given by:

$$\rho_t(\boldsymbol{a}|\boldsymbol{s}) = \frac{1}{Z(\boldsymbol{s})} \int w(\boldsymbol{s}, \boldsymbol{a}_0) \pi_{\text{old}}(\boldsymbol{a}_0|\boldsymbol{s}) p_{t|0}(\boldsymbol{a}|\boldsymbol{a}_0)\, d\boldsymbol{a}_0, \tag{29}$$

where $Z(\boldsymbol{s}) = \int w(\boldsymbol{s}, \boldsymbol{a}_0) \pi_{\text{old}}(\boldsymbol{a}_0|\boldsymbol{s}) d\boldsymbol{a}_0$ is the normalization constant. Mathematically, the negative weights introduce negative components to the integral. To satisfy the normalization constraint $\int \rho_t(\boldsymbol{a}|\boldsymbol{s}) d\boldsymbol{a} = 1$, the contribution from regions with positive weights must be amplified to offset the negative mass. Consequently, this mechanism effectively concentrates significantly higher probability mass on positive samples than standard reweighting.

Case II: For all $\boldsymbol{a}_t$ with $\mathbb{E}_{\boldsymbol{a}_0 \sim p_{0|t}}[w(\boldsymbol{s}, \boldsymbol{a}_0)] \leqslant 0$, the quadratic term in loss function in (26) has negative coefficients. Therefore, the loss is unbounded and have a "repelling" effect, consistently pushing D_θ away from the solution in (9) to arbitrary directions, which also pushes generated path $\boldsymbol{a}_t$ away from the negative weights region.

In summary, in both cases the negative weights will *push the generated probability paths away from actions with negative weights*.

Repelling effect helps policy improvement. There are two possible outcomes from the repelling effect: **a)** pushing the actions towards higher advantages such as Remark J.1; **b)** pushing actions to regions not included in the support of old policy π_{old}, encouraging exploration. Both cases provide useful feedback from negative samples.