ELUCIDATING THE DESIGN SPACE OF REGRESSION-BASED DIFFUSION REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

A nascent family of methods that forgoes the policy gradient and reweights a supervised regression instead has garnered momentum in reinforcement learning for diffusion and flow models. DiffusionNFT (Zheng et al., 2026), FlowAWR (Fu et al., 2026), and RAM (Bergmeister et al., 2026) are representative regimes with contrasting motivations. It is yet opaque what, if anything, they share. We substantiate that each is the solution of one divergence-constrained reward-maximization problem, and they are differentiated only by the convex generator that defines the constraint. Under the unified modeling framework, we unravel the relaxations that prior art made during building the advantage-embedded regression target: approximating the KKT condition and posterior normalizer for the linear and exponential tilt shapes DiffusionNFT and FlowAWR respectively, while preserving the exact sparsemax projection onto the probability simplex for linear tilt leads to another superior model type in this work. Beyond the theoretical underpinnings, we further empirically investigate the design space and shed light on the training recipe for regression-style diffusion RL. Retaining the merits discovered during our exploration gives rise to DiffusionRFT, our paradigm that converges faster, trains more stably, and attains the top performance.

1 INTRODUCTION

Post-training with an external reward plays a pivotal role in endowing a diffusion model with human-preference behaviors that pre-training dictates only weakly, such as prompt adherence and text rendering. Reinforcement learning serves as a cornerstone of this utility. The trailblazers in this field are the policy gradient (Liu et al., 2025; Xue et al., 2025) and direct reward backpropagation (Xu et al., 2023; Clark et al., 2024; Prabhudesai et al., 2024) scheme. Nevertheless, they are both cumbersome, in that a long denoising trajectory is indispensable and exhibits high variance. Moreover, the training objective diverges from the simple Mean-Squared Error (MSE) loss prevalent in pre-training, that is believed to simplify the generative distribution modeling to supervised learning, especially at scale.

Recently, a growing body of forward-process policy optimization ushers in a new wave of diffusion RL, including DiffusionNFT, FlowAWR and RAM, which reserve the regular diffusion loss and inject a reward into the regression target, so that a policy update is again a least-square fit on noised samples. These algorithms are grounded in divergent perspectives, ranging from CFG-style contrastive rectification Chen et al. (2025) to stochastic optimal control (Domingo-Enrich et al., 2025). Worse still, they are separately presented as standalone and integrated systems. The missing piece is a principled account of their subtle interweaving, and which design decisions drive the performance.

To bridge this gap, we uncover that one *optimization step* underlying these RL algorithms converges to a common driving force: maximize reward on a frozen rollout buffer under a divergence penalty. The divergence penalty translates the primary algorithmic choice into a single convex generator f (equation 2). The reverse-KL generator yields the exponential tilt, whose group estimator coincides with the FlowAWR objective, while the Pearson χ^2 generator yields a linear tilt, that DiffusionNFT already applies by standardizing the reward within a prompt group and dividing by a heuristic constant, but without such a derivation. As stated above, permitting the realization of the unified optimization objective demands an additional approximation, estimating a posterior normalizer by its endpoint counterpart. We demonstrate it preserves the direction of every step exactly while rescaling its length by a factor with unit mean and variance bounded by the second moment of the advantage.

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.

The Pearson χ^2 case turns out to be the more promising one, to which most of our research endeavor is devoted. Its exact constrained optimum is a Euclidean projection onto the probability simplex, that is the sparsemax weight, whereas the linear advantage heuristically adopted by DiffusionNFT keeps only the interior branch of the Karush–Kuhn–Tucker conditions and silently drops the constraint that relative densities stay non-negative. We articulate the condition under which the dense form is exact, observe that it is violated by the dense linear tilt on most training steps, and the training collapses afterwards. Retaining the projection repairs the collapse and achieves higher rewards. We term our resulting method *Diffusion Regression Fine-tuning*, or DiffusionRFT, a notably more stable and efficient point in this design space.

Beyond the scope of our derivation lie the empirical design choices. The remainder is cast into four axes: the regression variable, the reward shaping and its baseline, the anchor from which the target is displaced, and the law governing the renoising timestep, and we adjust each in isolation. The regression variable lays the groundwork for subsequent algorithmic axes. Empirical results advocate the sparsemax tilt in line with our exact derivation. Meanwhile, an analytically variance-minimizing baseline further facilitates performance stabilization and enhancement across all tilts. A frozen anchor, though supported by RAM, admits no safe operating point, regardless of the update magnitude. Decoupling the renoising timestep from the generating trajectory, which might be assumed to enrich the experience buffer, instead steers the loss toward noise levels the sampler never visits.

2 Reward Maximization under a Divergence Constraint

2.1 Policy Improvement on a Frozen Rollout Buffer

The policy-gradient family, from pioneering PPO (Schulman et al., 2017) to the critic-free estimators of GRPO (Shao et al., 2024), alternates between drawing trajectories from a behavior policy π^{old} and updating the current policy π_θ on the resulting batch. Only the outer loop is online: within one update the sampling distribution is fixed, so the batch defines a static measure $\mathcal{D}$ and the update poses an offline question — raise the expected reward, but not so far from π^{old} that the batch stops being informative about π_θ. As a penalty,

$$\max_{\pi_\theta} \;\; \mathbb{E}_{s\sim\mathcal{D}}\Big[\, \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\, R(s,a) \;-\; \gamma\, D\big(\pi_\theta(a|s) \,\big\|\, \pi^{\mathrm{old}}(a|s)\big)\Big], \tag{1}$$

where s is a state, a an action, R a scalar reward, D a divergence between the two conditional laws, and $\gamma > 0$ sets the exchange rate between reward and conservatism.

It pays to specify D through the density ratio rather than through the policy itself. Writing

$$\rho(a|s) \;=\; \frac{\pi_\theta(a|s)}{\pi^{\mathrm{old}}(a|s)}, \quad D_f\big(\pi_\theta \,\|\, \pi^{\mathrm{old}}\big) \;=\; \mathbb{E}_{a\sim\pi^{\mathrm{old}}(\cdot|s)}\big[\, f(\rho(a|s))\,\big], \quad f \text{ convex},\ f(1)=0, \tag{2}$$

turns π_θ into a scalar reweighting of a distribution π^{old} we can already sample from, and confines the regularizer's entire effect to the generator f. Reverse KL is $f_{\mathrm{KL}}(\rho) = \rho\log\rho$, which we adopt here; §3.3 shows that DiffusionNFT implicitly commits to a different and equally natural generator. With $f = f_{\mathrm{KL}}$ the objective is strictly concave in ρ and separates across states, so a single multiplier for $\mathbb{E}_{\pi^{\mathrm{old}}}[\rho] = 1$ gives the classical solution (Rafailov et al., 2023)

$$\pi^*(a|s) \;=\; \frac{1}{Z(s)}\,\pi^{\mathrm{old}}(a|s)\,\exp\Big(\frac{R(s,a)}{\gamma}\Big), \qquad Z(s) \;=\; \mathbb{E}_{a\sim\pi^{\mathrm{old}}(\cdot|s)}\Big[\exp\Big(\frac{R(s,a)}{\gamma}\Big)\Big]. \tag{3}$$

The optimum is π^{old} times a positive, reward-monotone factor, so mass is redistributed towards high-reward actions rather than assigned to unvisited ones, and that factor is closed-form, so knowing the rewards on the batch means knowing the optimum. Together they make the problem a candidate for regression rather than for policy-gradient estimation. Taking a state to be a prompt c, an action a sample x_0, and $\pi^{\mathrm{old}}(\cdot|c)$ the terminal law $p^{\mathrm{old}}(\cdot|c)$ of the sampler, equation 3 becomes

$$p^*(x_0|c) \;=\; \frac{1}{Z(c)}\,p^{\mathrm{old}}(x_0|c)\,\exp\Big(\frac{R(c,x_0)}{\gamma}\Big), \tag{4}$$

an *exponential tilt* of p^{old}. This is the endpoint distribution an update should aim at, and it is defined without reference to how the batch was explored, in particular no stochastic sampler is needed to make sense of it. It is not yet an objective, because a flow model is parameterized by a velocity field rather than by its endpoint density.

2.2 From a Tilted Endpoint Law to a Velocity Field

Flow matching (Lipman et al., 2023; Liu et al., 2023) sends p_{data} at $t = 0$ to $\mathcal{N}(0, I)$ at $t = 1$ along $x_t = (1 - t)\, x_0 + t\, \epsilon$, with conditional velocity $u_t(x_t|x_0) = \frac{\epsilon - x_t}{1-t}$, and fits a network by regression onto it,

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{t,\, x_0 \sim p_{\text{data}},\, \epsilon \sim \mathcal{N}(0,I)} \Big[\big\| v_\theta(x_t, c, t) - u_t(x_t|x_0) \big\|^2 \Big], \tag{5}$$

whose minimizer is the marginal field $v^*(x_t, c, t) = \mathbb{E}[u_t(x_t|x_0) \,|\, x_t, c]$. The significance of equation 5 is that generative modeling reduces to supervised regression under a plain MSE loss, and the series of forward-process RL emerges as an attempt to keep that property while injecting a reward.

Combining $v(x_t, c, t) = \frac{\hat{\epsilon} - x_t}{1-t}$ with Tweedie's formula $\nabla_{x_t} \log p_t(x_t|c) = -\hat{\epsilon}/t$ (Efron, 2011; Song et al., 2021b; 2023) expresses the field of *any* endpoint law through the score of its noised marginal, $v = -\big(x_t + t \nabla_{x_t} \log p_t\big)/(1 - t)$. Two laws noised by the same kernel would have fields differing only through the score of their ratio:

$$v^*(x_t, c, t) \;=\; v^{\text{old}}(x_t, c, t) \;-\; \frac{t}{1-t}\, \nabla_{x_t} \log \frac{p_t^*(x_t \mid c)}{p_t^{\text{old}}(x_t \mid c)}, \tag{6}$$

in which a reweighting of endpoints becomes an additive correction to the sampling field. It also disposes of the normalizer. $Z(c)$ in equation 4 does not depend on x_t and so leaves no trace in the gradient. The *exact* target therefore never requires a partition function; §3.2 shows that the estimator actually implemented reintroduces one, and quantifies what that costs.

This is still far from a trainable target: the ratio is taken between *noised* marginals whereas the tilt is defined on endpoints, so the gradient is a posterior expectation rather than a per-sample quantity. Converting it is the remaining step, and since it is identical for every tilt we perform it once, in §3.2.

3 From a Heuristic Linear Tilt to the Exact Simplex Projection

DiffusionNFT contrasts two implicit branches of the same model rather than forming a density ratio, so its update needs no likelihood ELBO (Liu et al., 2025; Wallace et al., 2024). For the same reason, it never states which divergence it controls. This section recovers that information and places the family inside §2.1. We find that a single DiffusionNFT update is a least-square regression onto a target of the advantage-weighted form, and that the reward enters that target *linearly* rather than *exponentially*. §3.1 puts the update in regression form and records the general statement that any endpoint reweighting induces such a target; §3.2 derives the reverse-KL counterpart, the exponential tilt, which coincides with the concurrently proposed FlowAWR objective; §3.3 identifies that a linear tilt is exactly what the Pearson χ^2 penalty produces, but that its constrained optimum is a sparsemax projection of which the dense linear form is only the interior branch.

3.1 A Regression Form for the DiffusionNFT Update

Given an optimality probability $r \in [0, 1]$, DiffusionNFT treats the reward-conditioned distribution as a mixture of a "positive" and a "negative" branch, and parameterizes either branch implicitly through the velocity difference it induces relative to p_t^{old}. Because the two branches share the same noising family, that difference reduces to a difference of scores, which is the step we exploit below.

Concretely, the two branches are coupled through a single coefficient β. Factoring that coupling out leaves a sum of two ordinary regression terms (temporal and conditional arguments suppressed),

$$\mathcal{L}_{\text{NFT}}(\theta) = \beta^2 \mathbb{E}\Big[r \Big\| v_\theta - \Big(v^{\text{old}} + \frac{1}{\beta}(u_t - v^{\text{old}}) \Big) \Big\|^2 + (1 - r) \Big\| v_\theta - \Big(v^{\text{old}} - \frac{1}{\beta}(u_t - v^{\text{old}}) \Big) \Big\|^2 \Big]. \tag{7}$$

Each term displaces the regression target from v^{old} along the residual $u_t - v^{\text{old}}$, in opposite directions according to whether the sample is rewarded, and the population optimum of equation 7 is

$$v^*(x_t, c, t) = v^{\text{old}}(x_t, c, t) + \mathbb{E}_{x_0 \sim p^{\text{old}}(\cdot|x_t, c)} \frac{2r - 1}{\beta} \big(u_t(x_t|x_0) - v^{\text{old}}(x_t, c, t) \big). \tag{8}$$

It already recovers the shape we desire: a sampling field plus a rewarded residual $u_t - v^{\text{old}}$, that is not specific to DiffusionNFT, as equation 6 forces it as soon as the target is any reweighting $p^*(x_0|c) \;=$

$p^{\text{old}}(x_0|c)\, w(c, x_0)$ of the sampling law, w.l.o.g., $w \geq 0$ and $\mathbb{E}_{x_0 \sim p^{\text{old}}(\cdot|c)}[w] = 1$. With $z(x_t, c, t) = \mathbb{E}_{x_0 \sim p^{\text{old}}(\cdot|x_t, c)}[w]$ being the posterior mean of the weight, substituting into equation 6 and trading the posterior score for the residual collapses the correction into an expectation,

$$v^*(x_t, c, t) = v^{\text{old}}(x_t, c, t) + \mathbb{E}_{x_0 \sim p^{\text{old}}(\cdot|x_t, c)} \left[\frac{w(c, x_0)}{z(x_t, c, t)} \left(u_t(x_t|x_0) - v^{\text{old}}(x_t, c, t) \right) \right]. \quad (9)$$

Both steps are elementary and appendix B gives them in full. In equation 9, the field is corrected by the residual reweighted by the *relative* tilt $\kappa = w/z$, the weight a sample carries measured against the average weight of the endpoints compatible with the same x_t. Only that ratio appears, so any constant factor in w, a partition function in particular, is canceled out. Furthermore, since v^{old} is by construction the posterior mean of u_t, the residual has zero conditional mean, $\mathbb{E}_{p^{\text{old}}(\cdot|x_t, c)}[u_t - v^{\text{old}}] = 0$, and therefore subtracting any constant b

$$\mathcal{A}(x_0, x_t) \;=\; \kappa \;-\; b \quad (10)$$

leaves equation 9 intact, so no value of b is forced. Since $\mathbb{E}[w \mid x_t] = z$ by definition, the relative tilt has unit posterior mean and $b = 1$ is the value that centers $\mathcal{A}$, turning it into a signed quality score in which positive advantages pull the field towards a sample and negative ones push it away. That is a convention instead of a derivation, and §4.2 uses the remaining freedom to choose b so as to minimize variance. We quote the baseline b below, explicitly with $b = 1$ unless stated otherwise.

Comparing equation 8 to equation 9 now locates DiffusionNFT precisely. Its optimality probability $r = \frac{1}{2} + z/(2C)$ is an affine mapping of the reward, where $z = (R - \bar{R})/\sigma$ is the reward standardized within the prompt group and C is a fixed constant. Substituting into equation 8 gives rise to

$$\mathcal{A}_{\text{NFT}} \;=\; \frac{2r - 1}{\beta} \;=\; \frac{1}{\beta C} \frac{R(c, x_0) - \bar{R}(c)}{\sigma(c)}, \quad (11)$$

so the weight DiffusionNFT applies is *linear* in the reward, with no exponential anywhere, and βC plays the role of a temperature. Appendix C corroborates that equation 9 restores the same linear weight when the branch is established through Bayes' rule and Tweedie's formula.

Nothing in that construction, however, claims the weight *should* be linear, nor what the temperature should be, both are chosen heuristically. Set against the exponential tilt equation 4 that reverse KL produces, DiffusionNFT is thus not solving the KL-regularized problem, which raises two questions: which penalty a linear tilt is its exact solution to, and whether the dense form from DiffusionNFT is that solution? §3.3 answers both, and the answer to the second is negative.

3.2 Returning to the KL Constraint: the Exponential Tilt

The machinery is indifferent to which tilt it is, so the reverse-KL case of §2.1 costs nothing else. Its solution equation 4 is the tilt $w \propto \exp(R/\gamma)$, and equation 9 converts it into the reverse-KL instance

$$\kappa \;=\; \frac{\exp(R(c, x_0)/\gamma)}{\mathbb{E}_{x_0 \sim p^{\text{old}}(\cdot|x_t, c)}[\exp(R(c, x_0)/\gamma)]}, \qquad \mathcal{A} = \kappa - b, \quad (12)$$

to be contrasted in §3.3 with its χ^2 counterpart. The normalizer $Z(c)$ of equation 4 has already been canceled in the ratio κ, and $\mathbb{E}_{p^{\text{old}}(\cdot|x_t, c)}[\kappa] = 1$ as always, so $b = 1$ centers this advantage too.

Equation (12) is still a population statement, as the reweighting $\kappa = w/z$ divides by the posterior mean $z(x_t, c, t) = \mathbb{E}_{x_0 \sim p^{\text{old}}(\cdot|x_t, c)}[w]$, an average over every endpoint that could have produced the same noised state. The linear tilt is in the same position and for the same reason: its threshold τ is fixed by equation 18, itself being a posterior expectation. Both therefore call for the group-relative style estimator (Shao et al., 2024; Ahmadian et al., 2024; Hu et al., 2025): draw a group of G samples $\{x_0^{(i)}\}_{i=1}^G$ for the prompt c and average over the grouped samples,

$$\kappa_i \;=\; \frac{w_i}{z_i} \approx \frac{\exp\left(R(c, x_0^{(i)})/\gamma\right)}{\frac{1}{G}\sum_{j=1}^G \exp\left(R(c, x_0^{(j)})/\gamma\right)}, \qquad \mathcal{A}(x_0^{(i)}) = \kappa_i - b. \quad (13)$$

It is worth being clear about what has just been swapped, because the group is not a sample from the posterior that equation 12 asks about. The $x_0^{(i)}$ are drawn i.i.d. from the endpoint law $p^{\text{old}}(\cdot|c)$, each

from its own initial noise, so their average is unbiased for the normalizer $Z(c) = \mathbb{E}_{x_0 \sim p^{\text{old}}(\cdot|c)}[w]$ of equation 4, and not for the posterior mean $z(x_t, c, t)$, which conditions on the noised state. These two coincide only where the posterior equals the prior. Estimating z itself is self-defeating rather than merely harder: the only samples available are prior draws, and reweighting them by $q(x_t|x_0^{(i)})$ provides a proposal whose efficiency degrades exponentially in higher dimension. At the latent size used here the reweighting concentrates on the single endpoint that generated x_t, hence $\hat{z} \to w_i$, $\kappa \to 1$ and $\mathcal{A} \to 0$, then the honest estimator annihilates the update. Substituting $Z(c)$ keeps the objective non-trivial, and its effect is characterized in appendix D. In broad terms, the net effect is simply a scalar reweighting $r_t(x_t)$. Writing y for the regression target that equation 13 induces,

$$\mathbb{E}\big[y \,\big|\, x_t\big] - v^{\text{old}} = r_t(x_t)\,\big(v^* - v^{\text{old}}\big), \quad r_t(x_t) \;=\; \frac{z(x_t, c, t)}{Z(c)} \;=\; \frac{p_t^*(x_t|c)}{p_t^{\text{old}}(x_t|c)} \;>\; 0. \tag{14}$$

The rescaling r_t is centered at one. Its variance is bounded by $\mathbb{E}[\mathcal{A}^2]$, and decays towards the noisy end of the schedule, so the implemented target is displaced from the anchor in exactly the direction of the optimal field and only its step length is affected. Refer to appendix D for a rigorous proof.

Up to now, one expectation is still alive, and it is the ordinary one that any flow-matching loss carries: the average over the noise level t and the renoising draw ϵ. Estimating it by a single Monte-Carlo pair per sampled endpoint turns equation 9 into a per-sample least-square objective,

$$\mathcal{L}(\theta) = \frac{1}{G}\sum_{i=1}^{G} \mathbb{E}_{t,\epsilon}\left[\left\| v_\theta(x_t^{(i)}, t) - \Big(v^{\text{old}}(x_t^{(i)}, t) + \mathcal{A}(x_0^{(i)}) \cdot \big(u_t(x_t^{(i)}|x_0^{(i)}) - v^{\text{old}}(x_t^{(i)}, t)\big)\Big)\right\|^2\right], \tag{15}$$

in which every term is computable from the pair at hand. This substitution differs in kind from the previous one. It replaces an exact target by a *stochastic* one, and unlike the passage from z to Z it is free. Conditionally on x_t the drawn target has the same mean as the quantity it stands for, so the objective differs by a constant independent of θ. The gradients and minimizer are unchanged, and only the variance of the gradient estimator is affected. Appendix E affords the argument, which is the same tower-property observation that licenses conditional flow matching itself. The constant is identified as the conditional variance of the target, which is also what the baseline of equation 23 tends to reduce later. Combining the two substitutions, equation 15 is an unbiased stochastic objective for the rescaled field $v^{\text{old}} + r_t(v^* - v^{\text{old}})$ of equation 14, rather than for v^* itself.

Equation (15) with the softmax weight in equation 13 echoes the objective proposed concurrently by FlowAWR (Fu et al., 2026), which we thus recover as the reverse-KL corner of equation 2 rather than as a separate algorithm. The Pearson χ^2 corner, and the sense in which it is the one DiffusionNFT occupies, is the subject of §3.3.

3.3 The Exact Solution: Pearson χ^2 and the Sparsemax Tilt

The linear reward weighting that §3.1 extracted from the DiffusionNFT update is not an artefact of that particular construction, it is what the problem of §2.1 returns for a different generator. Retaining the family of equation 2 and replacing $f_{\text{KL}}(\rho) = \rho \log \rho$ by the Pearson χ^2 generator $f_{\chi^2}(\rho) = \frac{1}{2}(\rho - 1)^2$, so that D_{χ^2} is the half second moment of $\rho - 1$ under π^{old}, the objective becomes

$$\max_{\pi_\theta} \mathbb{E}_{s\sim\mathcal{D}}\left[\mathbb{E}_{a\sim\pi_\theta(\cdot|s)} R(s,a) - \gamma D_{\chi^2}(\pi_\theta(\cdot|s)||\pi^{\text{old}}(\cdot|s))\right]. \tag{16}$$

Unlike the reverse KL, whose solution is less constrained in practice, the χ^2 problem must be solved over $\mathcal{K} = \{\rho : \mathbb{E}_{\pi^{\text{old}}}[\rho] = 1, \rho \geq 0\}$, which imposes normalization *and* non-negativity. The objective in equation 16 is strictly concave in ρ, and $\mathcal{K}$ convex and non-empty, so the maximizer exists, is unique, and is fully characterized by the Karush–Kuhn–Tucker conditions (Boyd & Vandenberghe, 2004). Complementary slackness on the non-negativity multiplier splits it into an interior branch and a boundary branch, and coalescing them as in appendix F yields the optimal terminal distribution

$$p^*(x_0|c) \;=\; p^{\text{old}}(x_0|c) \;\cdot\; \frac{1}{\gamma}\Big[\, R(c, x_0) - \tau(c) \,\Big]_+, \qquad [\,z\,]_+ = \max(z, 0), \tag{17}$$

in which the threshold $\tau(c)$ is not free but is pinned by the normalization constraint,

$$\Phi(\tau) \;\equiv\; \mathbb{E}_{x_0\sim p^{\text{old}}(\cdot|c)}\big[\,(R(c, x_0) - \tau)_+\,\big] \;=\; \gamma. \tag{18}$$

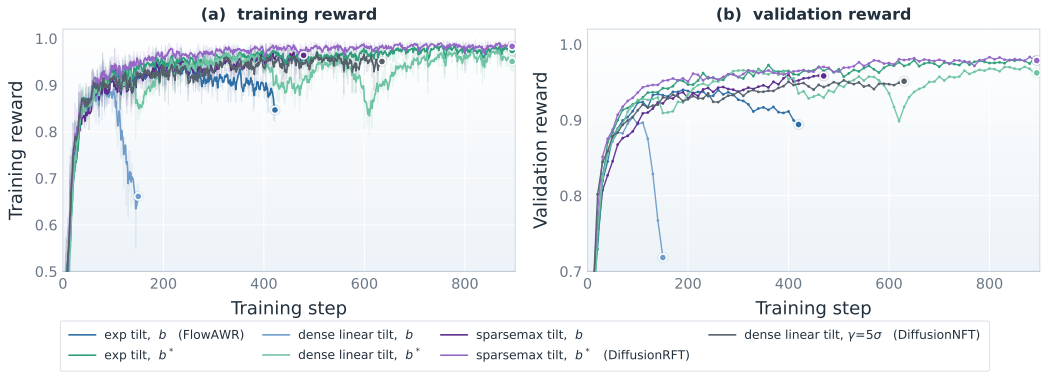

Figure 1: **Tilt × baseline grid.** Three tilts (exponential, dense linear, sparsemax) crossed with two baselines (canonical constant b, variance-minimizing control variate b^*). The sparsemax tilt reduces to the dense linear tilt when the truncation is inactive, i.e., $\min_i \mathcal{A} > -1$. Yet, this criterion fails on 86.8% of the dense run's steps. The optimal control variate b^* stabilizes and strengthens every tilt.

Since Φ is convex, continuous and strictly decreasing on $(-\infty, R_{\max})$ with $\Phi(R_{\max}) = 0$, equation 18 has a unique root, therefore equation 17 admits a transparent reading: τ stands for a *water level* determined by the regularization budget γ, samples above it receive probability mass in proportion to their excess reward, while samples below it receive none. Note that no partition function survives, since equation 18 already enforces $\mathbb{E}_{p^{\text{old}}}[\rho] = 1$: where the exponential tilt has to estimate $Z(c)$ by a group average, the projection obtains it from τ at no cost. This does not exempt it from the posterior normalizer z of equation 9, that each member of the family bypasses rather than calculates.

Two circumstances are separated as follows, contingent upon the KKT condition. When γ is large relative to the spread of the reward, every sample lies above the water level, the constraint $\rho \geq 0$ is relaxed, and equation 18 reduces to $\Phi(\tau) = \bar{R} - \tau$ so that $\tau = \bar{R} - \gamma$. Substituting back recovers the familiar centered form

$$p^*(x_0|c) \;=\; p^{\text{old}}(x_0|c)\left(1 + \frac{R(c,x_0) - \bar{R}(c)}{\gamma}\right), \qquad \mathcal{A}_{\chi^2}(x_0) \;=\; \frac{R(c,x_0) - \bar{R}(c)}{\gamma}, \tag{19}$$

which is exactly the linear weighting of §3.1, and also the group-centered advantage used by GRPO-style estimators. This regime is characterized sharply by

$$\gamma \;>\; \bar{R}(c) - \operatorname{ess\,inf} R(c,\cdot) \quad\Longleftrightarrow\quad \kappa \;>\; 0 \;\; \text{(equivalently } \mathcal{A}_{\chi^2} > -b\text{)} \quad \text{for every sample,} \tag{20}$$

whose right-hand form costs nothing to monitor during training and will be leveraged in §4 to decide empirically which regime a run is in. In the second regime the water level rises above the weakest samples, ρ saturates at zero for them, and equation 19 is no longer the solution of equation 16.

Criterion equation 20 settles one more thing: why DiffusionNFT's heuristic is significant. Its weight equation 11 is equation 19 at temperature $\beta C\sigma$, and a group-standardized reward is bounded by $\sqrt{G-1}$ whatever the rewards are, so equation 20 holds for the whole family exactly when $G \leq 1+(\beta C)^2$ (see appendix F for derivation). DiffusionNFT satisfies this bound only with its designated hyperparameters, while the sparsemax projection is feasible at every temperature and every G.

That the two divergences differ here is structural. The optimality condition reads $\rho = (f')^{-1}\big((R-\lambda)/\gamma\big)$, so f' alone decides the shape of the tilt: $f'_{\text{KL}}(\rho) = \log\rho + 1$ diverges at $\rho \to 0^+$ and acts as a logarithmic barrier, so $\rho \geq 0$ is never active, and inverting f' produces the exponential tilt; whereas $f'_{\chi^2}(\rho) = \rho - 1$ is finite at the boundary, which is reachable, and inverting f' produces a linear tilt with the constraint active when $(R-\lambda)/\gamma < -1$. Every non-barrier f-divergence inherits this behavior, so the truncation in equation 17 is generic to the linear family rather than peculiar to χ^2. Feeding this tilt through equation 9 gives the optimal velocity field with $\kappa = [R-\tau]_+/\gamma$; appendix F carries out the KKT solution, the intermediate marginals and the projection step in full.

Applying to equation 17 the same group estimator in §3.2 makes its structure explicit. Let $w_i = \frac{\rho_i}{G}$ be the probability that π_θ assigns to $x_0^{(i)}$, so that $\mathcal{K}$ becomes the probability simplex Δ^{G-1}. The empirical form of equation 16 is then $\max_{w\in\Delta}\langle w, R\rangle - \frac{\gamma G}{2}\|w - \frac{1}{G}\mathbf{1}\|^2$, and completing the square identifies it as a Euclidean projection onto the simplex:

$$w^\star = \operatorname{sparsemax}\Big(\tfrac{1}{G}\mathbf{1} + \tfrac{R}{\gamma G}\Big), \qquad \mathcal{A}_{\chi^2}(x_0^{(i)}) = \kappa - b = G\,w_i^\star - b = \frac{\big[R(c,x_0^{(i)}) - \tau\big]_+}{\gamma} - b, \tag{21}$$



Figure 2: **Loss geometry: the v-space update is misdirected, not too small.** More advantage signal (b) and largest movement budget (c), but an order-of-magnitude larger residual (d), and correspondingly less reward (a). The dashed line marks the matched horizon among these runs.

where τ now solves $\sum_j (R(c, x_0^{(j)}) - \tau)_+ = \gamma G$, computable in $\mathcal{O}(G \log G)$ by sorting the group rewards. The two tilts are thus in exact correspondence: the exponential tilt is $\mathrm{softmax}$ on the simplex and the linear tilt is the $\mathrm{sparsemax}$ of Martins & Astudillo (2016), the entropic and Euclidean projections of the same linear objective, and equation 19 is the sparsemax solution restricted to its dense regime. This dense form is also adopted by concurrent works on flow RL, either directly (Kveton et al., 2026) or after reshaping the advantage (Sun et al., 2026). Equation 21 reveals what such choices approximate, and equation 20 unveils when the approximation is exact.

Remark. The three algorithms are distinct by the choice of w and by nothing else: the reverse KL yields the softmax weight equation 13, the Pearson χ^2 the sparsemax weight equation 21, and the linear tilt that DiffusionNFT adopts heuristically is the interior branch in equation 19 of the latter. What the derivation does *not* fix is everything outside that choice, and §4 takes it up.

4 Design Space Navigation

The unified optimization objective is $\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{t,\epsilon} \| \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_t) - \boldsymbol{f}_{\text{target}} \|^2$, where

$$\boldsymbol{f}_{\text{target}} = \boldsymbol{f}_{\text{anchor}} + \mathcal{A}(\boldsymbol{f}_{\text{gt}} - \boldsymbol{f}_{\text{old}}), \qquad \boldsymbol{x}_t = (1-t)\,\tilde{\boldsymbol{x}}_0 + t\,\epsilon. \tag{22}$$

Here $\tilde{\boldsymbol{x}}_0$ is the clean sample returned by the sampler through the whole ODE denoising trajectory, and ϵ is the draw used to renoise that sample back to timestep t. $\boldsymbol{f}_{\text{old}}$ is the rollout policy responsible for sampling the trajectory, $\boldsymbol{f}_{\text{anchor}}$ is the field the target is displaced from (the value the regression falls back to when the advantage vanishes, set to $\boldsymbol{f}_{\text{old}}$ by equation 9 but not forced to be), and $\boldsymbol{f}_{\text{gt}}$ is the ground truth, which could be $\boldsymbol{x}$-, ϵ-, or $\boldsymbol{v}$-prediction (Li & He, 2026; Ho et al., 2020; Lipman et al., 2023). There exists a well-documented source of discrepancy in diffusion training, where the choice of parameterization and the associated loss weighting substantially determines convergence speed and stability (Karras et al., 2022; Kingma & Gao, 2023; Esser et al., 2024).

Reading equation 22 together with equation 15 exposes four design choices that the derivation of §3 leaves open and §4.2 varies one at a time:

- **(i) loss function**: the variable space $\boldsymbol{f}$ in which the regression is materialized, and the weighting across noise levels that this choice prefers;
- **(ii) reward shaping**: which tilt produces the weight κ, and which baseline b is subtracted from it;
- **(iii) anchor point**: whether $\boldsymbol{f}_{\text{anchor}}$ is the rollout policy $\boldsymbol{f}_{\text{old}}$, refreshed every epoch, or the pretrained one $\boldsymbol{f}_{\text{ref}}$, held fixed for the whole run; they agree at the first update and diverge from then on;
- **(iv) experience buffer**: the law of the renoising timestep t at which the loss is evaluated.

4.1 Experiment settings

For a fair comparison and reproducibility, we follow DiffusionNFT's codebase to adapt other follow-up variants. The training and evaluation dataset is GenEval (Ghosh et al., 2023), that measures compositional spatial reasoning in text-to-image tasks. All experiments are conducted with the Stable Diffusion 3.5-Medium backbone (Esser et al., 2024) at a 512×512 resolution. Other experimental details and additional results on Pickscore (Kirstain et al., 2023) are provided in appendix G.

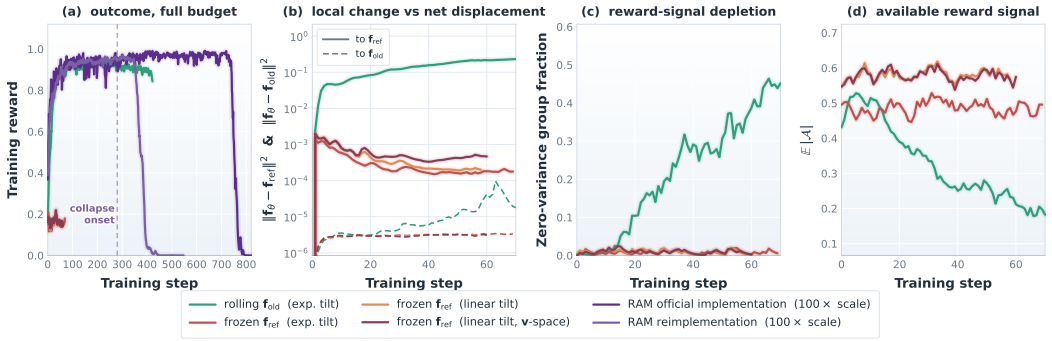

Figure 3: **Anchor-lock is a displacement failure, not a signal failure.** (a) spans the full budget, where both the official and reimplemented RAM escape the lock with the aid of a $100\times$ advantage scale but then collapse. (b)–(d) are restricted to the matched window in which the frozen-anchor runs coexist: dashed, non-zero step updates; solid, no effective accumulation (b) and no prompt group is ever solved (c), even though the advantage is *larger* than under a rolling anchor (d). The three frozen-anchor curves overlap in every panel, which pinpoints the anchor as the root cause.

4.2 Ablation Studies

loss function We start from equation 15, a widely acknowledged exponential tilt, to launch our exploration. We find that the $\boldsymbol{v}$-prediction described in prior literature (Zheng et al., 2026; Fu et al., 2026) converges slowly in training, agreeing with the gap between theoretical justification and empirical implementation, as mentioned by Kveton et al. (2026). Consequently, we adopt $\boldsymbol{x}$-prediction to expedite training hereafter, in light of a sharp gap displayed in figure 2(a). Furthermore, certain subtle implementation details decide training dynamics. Particularly, the loss scaling factor and self-normalization accelerate training convergence. These points of interest are elaborated in another contemporary work in the context of diffusion distillation (Kim et al., 2026).

The training dynamics in figure 2 show that the gap concerns the update direction, not magnitude. At a matched horizon the $\boldsymbol{v}$-prediction run carries the *largest* advantage and spends the *largest* movement budget, but gains the *least* reward, while its regression loss is an order of magnitude higher than $\boldsymbol{x}$-prediction. An empirical decomposition constrains several contributors, e.g. advantage scale, prediction units, and residual excess, only loosely, leaving room for the change in the irreducible floor alone to account for the observed loss ratio. In particular, the $\boldsymbol{x}$-space loss eschews the coefficient t^{-2} with which the $\boldsymbol{v}$-space loss uplifts the endpoint's irreducible inflation. Appendix H delves deeper into the rationale. The resulting $+0.34$ gain in the terminal training reward is pronounced, so we fix this loss geometry throughout rather than treating it as one more knob.

reward shaping According to the previous section, the exponential and linear tilt are both viable paths to diffusion RL outcomes. However, to minimize the variance of $(\mathcal{A}-b)\delta$ during RL training, where $\delta = u_t - v^{\text{old}}$ for brevity, the analytic solution of this control variate is

$$b^* = \mathbb{E}\left[\mathcal{A}\|\delta\|^2\right] \,/\, \mathbb{E}\|\delta\|^2. \tag{23}$$

Different from the pilot experiments above, we extend training iterations to over $10\times$. Three conclusions survive the full 3×2 grid in figure 1. **(1)** Collapse requires *both* a large effective update and the absence of a control variate, since either the projection or b^* prevents it alone. Once b^* is in place the three tilts are all stabilized. **(2)** These two strategies are complementary rather than interchangeable: the projection keeps the linear tilt inside the region where it is the optimum, while b^* advances the envelope of any tilt. Their combination, DiffusionRFT, is the safest and strongest configuration, at 0.983 evaluation reward. **(3)** The dashed reference line of DiffusionNFT also sits at the dense linear tilt cell, at a hard-coded temperature $\gamma = 5\sigma$, against $\gamma = \sigma$ for the plain dense linear tilt. The gap between the two curves is predicated on equation 20: at $\gamma = 5\sigma$ and $G = 24$ the interior branch is always legal, while at $\gamma = \sigma$ it is violated on 86.8% of steps. Appendix H affords a deeper account of the validity diagnostics.

anchor point DiffusionNFT anchors on a rolling copy, $\boldsymbol{f}_{\text{anchor}} = \boldsymbol{f}_{\text{old}} \leftarrow (1-\mu_i)\boldsymbol{f}_{\text{old}} + \mu_i \boldsymbol{f}_\theta$, trading staleness for stability. RAM instead anchors on the frozen pretrained model $\boldsymbol{f}_{\text{ref}}$. Replacing $\boldsymbol{f}_{\text{old}}$ by $\boldsymbol{f}_{\text{ref}}$ alone produces an *anchor-lock* phenomenon: the training reward of three frozen-anchor variants (with different tilt and regression space) rises to below 0.25 and stagnates, so the ceiling is a property of the anchor and not of one particular setup.



Figure 4: **Decoupling the renoising timestep makes the batch heavy-tailed, and stratification makes it worse.** (a) Training reward. (b) Within-batch spread of the policy-EMA gap $\|\boldsymbol{f}_\theta - \boldsymbol{f}_{\text{old}}\|^2$ on a logarithmic axis: lowest for the trajectory-coupled schedule, $\sim 2\times$ once the timestep is decoupled, and $\sim 8\times$ once the draws are stratified. (c) Advantage magnitude: the trajectory-coupled run declines as it solves its prompt groups, while the decoupled runs remain high. (d) Fraction of prompt groups whose rewards have zero variance, which carry no advantage. A uniform distribution spends its evaluations on uninformative states the sampler hardly visits instead of the ones it resolves.

The culprit of lock is limited displacement $\|\boldsymbol{f}_\theta - \boldsymbol{f}_{\text{anchor}}\|^2$, not a starved signal. In figure 3, the frozen-anchor runs carry *larger* advantages than the rolling-anchor one, yet they never leave the pre-trained field, even though the local change is non-negligible from the outset. With the anchor fixed the target is $\boldsymbol{f}_{\text{ref}} + \mathcal{A}\,\delta$, so each update re-centers the policy to $\boldsymbol{f}_{\text{ref}}$ and the attainable displacement is bounded by the advantage-weighted residual itself, whereas a rolling anchor $\boldsymbol{f}_{\text{old}}$ tracks $\boldsymbol{f}_\theta$ closely. Escaping the lock requires a larger update, and both RAM implementations, official and our reproduction of its skeleton, show that the escape is temporary. They peak at 0.977 and 0.961 and then fall to zero. The frozen anchor therefore has no safe operating point under our test: small updates lock, large ones diverge. In contrast, a rolling anchor needs no such handling, because re-centering keeps the displacement of order one at every epoch. Appendix H analyzes the mechanism in full.

experience buffer The training states $\boldsymbol{x}_t$ are renoised from the generated clean samples, where the renoising timesteps are tied to the backward denoising trajectory, which is reminiscent of the classic replay buffer in RL (Mnih et al., 2015). Differently, RAM decouples the forward renoising timestep schedule from the denoising trajectory: renoise timesteps are drawn from a uniform distribution and repeated $K = 8$ times, accumulating an $8\times$ experience buffer for one-step optimization.

RAM's sampling conflates two factors at once, the law of t and the number of copies K. Crossing the two achieves evaluation rewards of 0.972 & 0.975 for the trajectory-coupled law at $K = 1$ & $K = 8$, against 0.964 & 0.953 for the uniform law, the last of these losing a further 0.086 before the budget ends (figure 4). Expanding sampling K is therefore neutral and it is the law of t that counts. The underlying cause is that decoupling alters the *support* of t: the sampler-oriented distribution never visits $t < 0.43$ (figure 7), while $t \sim \mathcal{U}[0, 1]$ places 43% of the loss evaluations there. It also clarifies why two variance-reduction remedies, stratified sampling and shared (t, ϵ) sampling across copies (Tang et al., 2026), still fail, and why stratification, that enforces the uniform coverage, is even the more damaging of these two. Renoising along the generating trajectory only once ($K = 1$) keeps the training states on the support the policy occupies, which is both cheaper and better conditioned, so we retain it by default. Appendix H incorporates more in-depth diagnostics.

5 Conclusion

Regression-based reinforcement learning for diffusion models admits a unified formulation with a free parameter. The convex generator in the divergence penalty determines the particular choice of this parameter: an exponential tilt for FlowAWR; a linear tilt whose exact solution, the sparsemax projection, leads to our proposed DiffusionRFT; DiffusionNFT heuristically adopts the dense linear form that lies in the interior branch of the latter. The empirical picture favors getting the tilt done right. Standing on its own, the projected linear tilt performs the best among them, and it is the only candidate that does not surrender its peak reward; a variance-minimizing baseline then stabilizes and improves all three tilts, and the two together approach the fastest convergence and highest reward. Besides the tilt choice, what bears emphasis includes $\boldsymbol{x}$-prediction with appropriate loss calibration, an online cumulative anchor point, and a backward-aligned forward renoising schedule.

AI USE STATEMENT

In this work, we used generative AI tools for manuscript polish and LaTeX checks during revision. We have not used generative AI tools for anything else. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We believe the experimental details in appendix G are ready for reproduction. In addition, we would release the code upon publication to foster future research.

REFERENCES

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to Basics: Revisiting REINFORCE Style Optimization for Learning from Human Feedback in LLMs. *arXiv e-prints*, art. arXiv:2402.14740, 2024.

Andreas Bergmeister, Stefanie Jegelka, Nikolas Nüsken, Carles Domingo-Enrich, and Jakiw Pidstrigach. Reinforce Adjoint Matching: Scaling RL Post-Training of Diffusion and Flow-Matching Models. *arXiv e-prints*, art. arXiv:2605.10759, 2026.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2004.

Huayu Chen, Kai Jiang, Kaiwen Zheng, Jianfei Chen, Hang Su, and Jun Zhu. Visual generation without guidance. In *ICML*, 2025.

Kevin Clark, Paul Vicol, Kevin Swersky, and David J. Fleet. Directly fine-tuning diffusion models on differentiable rewards. In *ICLR*, 2024.

Carles Domingo-Enrich, Michal Drozdzal, Brian Karrer, and Ricky T. Q. Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. In *ICLR*, 2025.

John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. Efficient projections onto the l1-ball for learning in high dimensions. In *ICML*, 2008.

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

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *ICML*, 2024.

Zheming Fu, Ruizhe He, Wei Shang, Xiaoxiao Ma, Lei Wang, Chang Liu, and Siming Fu. FlowAWR: Online Adaptive Flow Reinforcement via Advantage-Weighted Rectification. *arXiv e-prints*, art. arXiv:2606.30376, 2026.

Yu Gao, Haoyuan Guo, Tuyen Hoang, Weilin Huang, Lu Jiang, Fangyuan Kong, Huixia Li, Jiashi Li, Liang Li, Xiaojie Li, Xunsong Li, Yifu Li, Shanchuan Lin, Zhijie Lin, Jiawei Liu, Shu Liu, Xiaonan Nie, Zhiwu Qing, Yuxi Ren, Li Sun, Zhi Tian, Rui Wang, Sen Wang, Guoqiang Wei, Guohong Wu, Jie Wu, Ruiqi Xia, Fei Xiao, Xuefeng Xiao, Jiangqiao Yan, Ceyuan Yang, Jianchao Yang, Runkai Yang, Tao Yang, Yihang Yang, Zilyu Ye, Xuejiao Zeng, Yan Zeng, Heng Zhang, Yang Zhao, Xiaozheng Zheng, Peihao Zhu, Jiaxin Zou, and Feilong Zuo. Seedance 1.0: Exploring the Boundaries of Video Generation Models. *arXiv e-prints*, art. arXiv:2506.09113, 2025.

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. In *NeurIPS*, 2023.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks. *arXiv e-prints*, art. arXiv:1406.2661, 2014.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *nat*, 645(8081):633–638, 2025.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *ICLR*, 2022.

Jian Hu, Jason Klein Liu, Haotian Xu, and Wei Shen. REINFORCE++: Stabilizing Critic-Free Policy Optimization with Global Advantage Normalization. *arXiv e-prints*, art. arXiv:2501.03262, 2025.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *NeurIPS*, 2022.

Youngjoong Kim, Deokyeong Lee, and Jaesik Park. Distribution Matching Distillation without Fake Score Network. *arXiv e-prints*, art. arXiv:2605.19256, 2026.

Diederik Kingma and Ruiqi Gao. Understanding diffusion objectives as the elbo with simple data augmentation. In *NeurIPS*, 2023.

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. In *NeurIPS*, 2023.

Branislav Kveton, Anup Rao, Subhojyoti Mukherjee, Krishna Kumar Singh, and Viet Dac Lai. AdvantageFlow: Advantage-Weighted Least Squares for RL in Flow Models. *arXiv e-prints*, art. arXiv:2605.26013, 2026.

Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. In *CVPR*, 2026.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *ICLR*, 2023.

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. In *NeurIPS*, 2025.

Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan LI, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. In *NeurIPS*, 2022.

Andre Martins and Ramon Astudillo. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *ICML*, 2016.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *NeurIPS*, 2022.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, 2023.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-Weighted Regression: Simple and Scalable Off-Policy Reinforcement Learning. *arXiv e-prints*, art. arXiv:1910.00177, 2019.

Mihir Prabhudesai, Russell Mendonca, Zheyang Qin, Katerina Fragkiadaki, and Deepak Pathak. Video Diffusion Alignment via Reward Gradients. *arXiv e-prints*, art. arXiv:2407.08737, 2024. doi: 10.48550/arXiv.2407.08737.

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. In *NeurIPS*, 2023.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. *arXiv e-prints*, art. arXiv:1707.06347, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. *arXiv e-prints*, art. arXiv:2402.03300, 2024.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021a.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021b.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *ICML*, 2023.

Haoyuan Sun, Jing Wang, Yuxin Song, Yu Lu, Bo Fang, Yifu Luo, Jun Yin, Pengyu Zeng, Miao Zhang, Tiantian Zhang, Xueqian Wang, and Shijian Lu. Power Reinforcement Post-Training of Text-to-Image Models with Super-Linear Advantage Shaping. *arXiv e-prints*, art. arXiv:2605.10937, 2026.

Bingda Tang, Yuhui Zhang, Xiaohan Wang, Jiayuan Mao, Ludwig Schmidt, and Serena Yeung-Levy. V-GRPO: Online Reinforcement Learning for Denoising Generative Models Is Easier than You Think. *arXiv e-prints*, art. arXiv:2604.23380, 2026.

Team Seedance, De Chen, Liyang Chen, Xin Chen, Ying Chen, Zhuo Chen, Zhuowei Chen, Feng Cheng, Tianheng Cheng, Yufeng Cheng, Mojie Chi, Xuyan Chi, Jian Cong, Qinpeng Cui, Fei Ding, Qide Dong, Yujiao Du, Haojie Duanmu, Junliang Fan, Jiarui Fang, Jing Fang, Zetao Fang, Chengjian Feng, Yu Gao, Diandian Gu, Dong Guo, Hanzhong Guo, Qiushan Guo, Boyang Hao, Hongxiang Hao, Haoxun He, Jiaao He, Qian He, Tuyen Hoang, Heng Hu, Ruoqing Hu, Yuxiang Hu, Jiancheng Huang, Weilin Huang, Zhaoyang Huang, Zhongyi Huang, Jishuo Jin, Ming Jing, Ashley Kim, Shanshan Lao, Yichong Leng, Bingchuan Li, Gen Li, Haifeng Li, Huixia Li, Jiashi Li, Ming Li, Xiaojie Li, Xingxing Li, Yameng Li, Yiying Li, Yu Li, Yueyan Li, Chao Liang, Han Liang, Jianzhong Liang, Ying Liang, Wang Liao, J. H. Lien, Shanchuan Lin, Xi Lin, Feng Ling, Yue Ling, Fangfang Liu, Jiawei Liu, Jihao Liu, Jingtuo Liu, Shu Liu, Sichao Liu, Wei Liu, Xue Liu, Zuxi Liu, Ruijie Lu, Lecheng Lyu, Jingting Ma, Tianxiang Ma, Xiaonan Nie, Jingzhe Ning, Junjie Pan, Xitong Pan, Ronggui Peng, Xueqiong Qu, Yuxi Ren, Yuchen Shen, Guang Shi, Lei Shi, Yinglong Song, Fan Sun, Li Sun, Renfei Sun, Wenjing Tang, Boyang Tao, Zirui Tao, Dongliang Wang, Feng Wang, Hulin Wang, Ke Wang, Qingyi Wang, Rui Wang, Shuai Wang, Shulei Wang, Weichen Wang, Xuanda Wang, Yanhui Wang, Yue Wang, Yuping Wang, Yuxuan Wang, Zijie Wang, Ziyu Wang, Guoqiang Wei, Meng Wei, Di Wu, Guohong Wu, Hanjie Wu, Huachao Wu, Jian Wu, Jie Wu, Ruolan Wu, Shaojin Wu, Xiaohu Wu, Xinglong Wu, Yonghui Wu, Ruiqi Xia, Xin Xia, Xuefeng Xiao, Shuang Xu, Bangbang Yang, Jiaqi Yang, Runkai Yang, Tao Yang, Yihang Yang, Zhixian Yang, Ziyan Yang, Fulong Ye, Bingqian Yi, Xing Yin, Yongbin You, Linxiao Yuan, Weihong Zeng, Xuejiao Zeng, Yan Zeng, Siyu Zhai, Zhonghua Zhai, Bowen Zhang, Chenlin Zhang, Heng Zhang, Jun Zhang, Manlin Zhang, Peiyuan Zhang, Shuo Zhang, Xiaohe Zhang, Xiaoying Zhang, Xinyan Zhang, Xinyi Zhang, Yichi Zhang, Zixiang Zhang, Haiyu Zhao, Huating Zhao, Liming Zhao, Yian Zhao, Guangcong Zheng, Jianbin Zheng, Xiaozheng Zheng, Zerong Zheng, Kuan Zhu, and Feilong Zuo. Seedance 2.0: Advancing Video Generation for World Complexity. *arXiv e-prints*, art. arXiv:2604.14148, 2026.

Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization. In *CVPR*, 2024.

Jing Wang, Jiajun Liang, Jie Liu, Henglin Liu, Gongye Liu, Jun Zheng, Wanyuan Pang, Ao Ma, Zhenyu Xie, Xintao Wang, Meng Wang, Pengfei Wan, and Xiaodan Liang. Grpo-guard: Mitigating implicit over-optimization in flow matching via regulated clipping. In *CVPR*, 2026.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-Image Technical Report. *arXiv e-prints*, art. arXiv:2508.02324, 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. In *NeurIPS*, 2023.

Shuchen Xue, Chongjian GE, Shilong Zhang, Yichen Li, and Zhi-Ming Ma. Advantage weighted matching: Aligning RL with pretraining in diffusion models. In *ICML*, 2026.

Zeyue Xue, Jie Wu, Yu Gao, Fangyuan Kong, Lingting Zhu, Mengzhao Chen, Zhiheng Liu, Wei Liu, Qiushan Guo, Weilin Huang, and Ping Luo. DanceGRPO: Unleashing GRPO on Visual Generation. *arXiv e-prints*, art. arXiv:2505.07818, 2025.

Xianghui Yang, Huiwen Shi, Bowen Zhang, Fan Yang, Jiacheng Wang, Hongxu Zhao, Xinhai Liu, Xinzhou Wang, Qingxiang Lin, Jiaao Yu, Lifu Wang, Jing Xu, Zebin He, Zhuo Chen, Sicong Liu, Junta Wu, Yihang Lian, Shaoxiong Yang, Yuhong Liu, Yong Yang, Di Wang, Jie Jiang, and Chunchao Guo. Hunyuan3D 1.0: A Unified Framework for Text-to-3D and Image-to-3D Generation. *arXiv e-prints*, art. arXiv:2411.02293, 2024.

Bing Zhao, Chenfei Wu, Deqing Li, Hao Meng, Jiahao Li, Jie Zhang, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kuan Cao, Kun Yan, Liang Peng, Lihan Jiang, Niantong Li, Ningyuan Tang, Shengming Yin, Tianhe Wu, Xiao Xu, Xiaoyue Chen, Xihua Wang, Yan Shu, Yanran Zhang, Yi Wang, Yilei Chen, Ying Ba, Yixian Xu, Yujia Wu, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhendong Wang, Zihao Liu, Zikai Zhou, An Yang, Chen Cheng, Chenxu Lv, Dayiheng Liu, Fan Zhou, Hantian Xiong, Hongzhu Shi, Hu Wei, Huihong Zhao, Ivy Liu, Jianwei Zhang, Jiawei Zhang, Kai Chen, Kang He, Levon Xue, Lin Qu, Linhan Tang, Luwen Feng, Minggang Wu, Minmin Sun, Na Ni, Rui Men, Shuai Bai, Sishou Zheng, Tao Lan, Tianqi Zhang, Tingkun Wen, Wei Wang, Weixu Qiao, Weiyi Lu, Wenmeng Zhou, Xiaodong Deng, Xiaoxiao Xu, Xinlei Fang, Xionghui Chen, Yanan Wang, Yang Fan, Yichang Zhang, Yixuan Xu, Yu Wu, Zhiyuan Ma, and Zhizhi Cai. Qwen-Image-2.0 Technical Report. *arXiv e-prints*, art. arXiv:2605.10730, 2026.

Zibo Zhao, Zeqiang Lai, Qingxiang Lin, Yunfei Zhao, Haolin Liu, Shuhui Yang, Yifei Feng, Mingxin Yang, Sheng Zhang, Xianghui Yang, Huiwen Shi, Sicong Liu, Junta Wu, Yihang Lian, Fan Yang, Ruining Tang, Zebin He, Xinzhou Wang, Jian Liu, Xuhui Zuo, Zhuo Chen, Biwen Lei, Haohan Weng, Jing Xu, Yiling Zhu, Xinhai Liu, Lixin Xu, Changrong Hu, Shaoxiong Yang, Song Zhang, Yang Liu, Tianyu Huang, Lifu Wang, Jihong Zhang, Meng Chen, Liang Dong, Yiwen Jia, Yulin Cai, Jiaao Yu, Yixuan Tang, Hao Zhang, Zheng Ye, Peng He, Runzhou Wu, Chao Zhang, Yonghao Tan, Jie Xiao, Yangyu Tao, Jianchen Zhu, Jinbao Xue, Kai Liu, Chongqing Zhao, Xinming Wu, Zhichao Hu, Lei Qin, Jianbing Peng, Zhan Li, Minghui Chen, Xipeng Zhang, Lin Niu, Paige Wang, Yingkai Wang, Haozhao Kuang, Zhongyi Fan, Xu Zheng, Weihao Zhuang, YingPing He, Tian Liu, Yong Yang, Di Wang, Yuhong Liu, Jie Jiang, Jingwei Huang, and Chunchao Guo. Hunyuan3D 2.0: Scaling Diffusion Models for High Resolution Textured 3D Assets Generation. *arXiv e-prints*, art. arXiv:2501.12202, 2025.

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. In *ICLR*, 2026.

Yuanzhi Zhu, Xi Wang, Stéphane Lathuilière, and Vicky Kalogeiton. Diffusion Reinforcement Learning via Centered Reward Distillation. *arXiv e-prints*, art. arXiv:2603.14128, 2026.

A Related work

A.1 Diffusion Generative Modeling

Inspired by thermodynamics (Sohl-Dickstein et al., 2015) in physics, the advent of diffusion models occurred one decade ago. In the following years, after briefly overshadowed by Generative Adversarial Networks (GANs) (Goodfellow et al., 2014), improved empirical practice (Ho et al., 2020) and a firmer theoretical footing (Song et al., 2021b) ignited the resurgence of diffusion generative modeling. The system is further scaled up with advanced architecture engineering (Rombach et al., 2022; Peebles & Xie, 2023), distribution coupling (Lipman et al., 2023; Liu et al., 2023), network capacity reallocation (Karras et al., 2022; Esser et al., 2024; Kingma & Gao, 2023), and higher-order solvers (Song et al., 2021a; Lu et al., 2022). Embracing diffusion as the generative modeling backend, recent years have witnessed an unprecedented achievement across a variety of visual generative applications, such as text-to-image (Wu et al., 2025; Zhao et al., 2026), video (Gao et al., 2025; Team Seedance et al., 2026), and 3D geometry (Yang et al., 2024; Zhao et al., 2025). Thanks to the pixel-level generative modeling, diffusion models permit a supervised learning MSE loss to be established and optimized stably (much more than GANs' brittle minimax game). Nevertheless, a

coin always has two sides: this training objective also falls short of straightforwardly modeling the sample-level human preference, such as visual aesthetics, physical plausibility, among others.

A.2 Diffusion Reinforcement Learning

Reinforcement learning has gained tremendous traction in Large Language Models (LLMs), contributing to both alignment with human preference (Ouyang et al., 2022) and incentivization of Chain-of-Thought (CoT) reasoning (Guo et al., 2025). The transfer of RL in autoregressive modeling to diffusion models poses a challenge, as there is no well-defined tractable likelihood formulation in diffusion modeling. One paradigm converts the deterministic ODE solver into a stochastic SDE, in order to frame each transition as a step of a Markovian Decision Process (MDP) and supplies the exploration noise that policy-gradient estimators require (Liu et al., 2025; Wang et al., 2026). Another strand sidesteps the likelihood calculation via directly back-propagating a differentiable reward through the sampler (Xu et al., 2023; Clark et al., 2024; Prabhudesai et al., 2024). Instead, DiffusionNFT innovatively presents the first diffusion-native RL prototype, through the lens of implicit contrastive learning between positive and negative representation fields (Zheng et al., 2026). However, the underlying insight of its reward-integrated regression target is not disclosed. Likewise, several recent works (Fu et al., 2026; Xue et al., 2026; Kveton et al., 2026) also lie in the lineage of advantage-weighted regression (Peng et al., 2019), where a reward-tilted distribution is fitted by weighted least squares rather than followed by a gradient. Mindful of its design elegance and the built-in compatibility with diffusion loss, we first link the DiffusionNFT formulation to the classic constrained reward maximization problem, and then venture as far as possible into the design space.

B Derivation of the reweighted velocity field

This appendix derives equation 9. Let $p^*(x_0|c) = p^{\text{old}}(x_0|c)\, w(c, x_0)$ be any tilt, as in §3.1. The derivation has four steps: push the tilt through the noising kernel, differentiate the resulting log-ratio, evaluate the conditional score in closed form, and collect the prefactors.

Step 1: noised marginals. Both laws are noised by the same kernel $p(x_t|x_0)$, so

$$p_t^*(x_t|c) = \int p(x_t|x_0)\, p^*(x_0|c)\, \mathrm{d}x_0 = \int p(x_t|x_0)\, p^{\text{old}}(x_0|c)\, w(c, x_0)\, \mathrm{d}x_0. \tag{24}$$

Bayes' rule for the reference process reads $p(x_t|x_0)\, p^{\text{old}}(x_0|c) = p_t^{\text{old}}(x_t|c)\, p^{\text{old}}(x_0|x_t, c)$. Substituting it into equation 24 and taking the x_0-independent factor out of the integral yields

$$\begin{aligned} p_t^*(x_t|c) &= p_t^{\text{old}}(x_t|c) \int p^{\text{old}}(x_0|x_t, c)\, w(c, x_0)\, \mathrm{d}x_0 = p_t^{\text{old}}(x_t|c) \cdot z(x_t, c, t), \\ z \;&\equiv\; \mathbb{E}_{x_0 \sim p^{\text{old}}(\cdot|x_t, c)}\big[w(c, x_0)\big]. \end{aligned} \tag{25}$$

The tilt on endpoints thus becomes a tilt on noised marginals, but by the *posterior mean* of w rather than by w itself. This is the only place the noising kernel is used, and it is why z cannot be evaluated per sample.

Step 2: differentiate the log-ratio. By equation 25 the ratio of noised marginals is exactly z, so equation 6 becomes

$$v^* - v^{\text{old}} = -\frac{t}{1-t}\, \nabla_{x_t} \log \frac{p_t^*(x_t|c)}{p_t^{\text{old}}(x_t|c)} = -\frac{t}{1-t}\, \nabla_{x_t} \log z = -\frac{t}{1-t}\, \frac{\nabla_{x_t} z}{z}. \tag{26}$$

For the numerator, differentiation of equation 25 under the integral sign is legitimate whenever w is p^{old}-integrable and the Gaussian kernel dominates, which holds here since $w \geq 0$ has unit mean. Multiply w by the posterior and then differentiate the integral:

$$\begin{aligned} \nabla_{x_t} z &= \int \nabla_{x_t} p^{\text{old}}(x_0|x_t, c)\; w\, \mathrm{d}x_0 \\ &= \int p^{\text{old}}(x_0|x_t, c)\, \nabla_{x_t} \log p^{\text{old}}(x_0|x_t, c)\; w\, \mathrm{d}x_0 \\ &= \mathbb{E}_{x_0 \sim p^{\text{old}}(\cdot|x_t, c)}\Big[w\, \nabla_{x_t} \log p^{\text{old}}(x_0|x_t, c)\Big], \end{aligned} \tag{27}$$

in which the middle equality is the log-derivative identity $\nabla\pi = \pi\,\nabla\log\pi$. Dividing by z converts equation 27 into an expectation of the conditional score weighted by the relative tilt w/z.

Step 3: conditional score in a closed form. Bayes' rule once more, now in the form

$$\log p^{\text{old}}(x_0|x_t,c) = \log p(x_t|x_0) + \log p^{\text{old}}(x_0|c) - \log p_t^{\text{old}}(x_t|c). \tag{28}$$

We kill the middle term under ∇_{x_t} and leave the other two scores. The forward noising kernel of §2.2 is $x_t|x_0 \sim \mathcal{N}\big((1-t)x_0,\, t^2 I\big)$, whose score is affine,

$$\nabla_{x_t}\log p(x_t|x_0) = -\frac{x_t-(1-t)x_0}{t^2} = \frac{(1-t)x_0 - x_t}{t^2}, \tag{29}$$

while the marginal score follows from Tweedie's formula, which for this kernel states $\hat{x}_0^{\text{old}} \equiv \mathbb{E}[x_0|x_t,c] = \big(x_t + t^2\nabla_{x_t}\log p_t^{\text{old}}(x_t|c)\big)/(1-t)$ and hence $\nabla_{x_t}\log p_t^{\text{old}}(x_t|c) = \big((1-t)\hat{x}_0^{\text{old}} - x_t\big)/t^2$. Subtracting the two,

$$\begin{aligned}\nabla_{x_t}\log p^{\text{old}}(x_0|x_t,c) &= \frac{(1-t)x_0 - x_t}{t^2} - \frac{(1-t)\hat{x}_0^{\text{old}} - x_t}{t^2}\\ &= \frac{1-t}{t}\cdot\frac{x_0-\hat{x}_0^{\text{old}}}{t}\\ &= -\frac{1-t}{t}\big(u_t(x_t|x_0) - v^{\text{old}}(x_t,c,t)\big),\end{aligned} \tag{30}$$

where the last equality is just a renaming: $u_t(x_t|x_0) = (x_t - x_0)/t$ is the conditional target and $v^{\text{old}}(x_t,c,t) = (x_t - \hat{x}_0^{\text{old}})/t$ the reference field, so $(x_0 - \hat{x}_0^{\text{old}})/t = -(u_t - v^{\text{old}})$. The posterior score has been traded for a quantity computable from one sample.

Step 4: collection. Chaining equation 26, equation 27 and equation 30,

$$v^* - v^{\text{old}} = \underbrace{-\frac{t}{1-t}}_{\text{equation 26}}\cdot\mathbb{E}\Bigg[\frac{w}{z}\cdot\underbrace{\Big(-\frac{1-t}{t}\Big)}_{\text{equation 30}}\big(u_t - v^{\text{old}}\big)\Bigg] = \mathbb{E}_{x_0\sim p^{\text{old}}(\cdot|x_t,c)}\Bigg[\frac{w(c,x_0)}{z(x_t,c,t)}\big(u_t - v^{\text{old}}\big)\Bigg], \tag{31}$$

since the two prefactors are reciprocal and the two signs cancel. This is equation 9 exactly. It may be inserted, along with any constant b, because $v^{\text{old}} = \mathbb{E}[u_t|x_t,c]$ forces $\mathbb{E}_{p^{\text{old}}(\cdot|x_t,c)}[u_t - v^{\text{old}}] = 0$ and hence

$$\mathbb{E}\Big[\big(\tfrac{w}{z} - b\big)\big(u_t - v^{\text{old}}\big)\Big] = \mathbb{E}\Big[\tfrac{w}{z}\big(u_t - v^{\text{old}}\big)\Big] - b\,\underbrace{\mathbb{E}\big[u_t - v^{\text{old}}\big]}_{=0} = v^* - v^{\text{old}}, \quad \forall b\in\mathbb{R}. \tag{32}$$

Among a plethora of potential choices, $b=1$ is distinguished by $\mathbb{E}_{p^{\text{old}}(\cdot|x_t,c)}[w/z] = 1$, immediately from $\mathbb{E}[w|x_t] = z$, which makes the reweighting zero-mean, while $b = b^*$ of equation 23 instead minimizes the second moment.

C The linear weight implied by the DiffusionNFT branches

This appendix derives equation 11. DiffusionNFT introduces a binary optimality variable o and identifies the reward with $p(o=1 \mid x_0, c)$, so its positive branch is the sampling law conditioned on $o=1$. Bayes' rule at noise level t therefore relates the two noised marginals by the optimality posterior,

$$\begin{aligned}p_t^+(x_t \mid c) &= p_t^{\text{old}}(x_t \mid c)\,\frac{p(o=1 \mid x_t,c)}{p(o=1 \mid c)},\\ p(o=1 \mid x_t,c) &= \mathbb{E}_{x_0\sim p^{\text{old}}(\cdot|x_t,c)}\big[R(c,x_0)\big] = \alpha(x_t,c),\end{aligned} \tag{33}$$

where the second identity is the tower property. Comparing equation 33 with the generic tilt of §3.1 identifies the branch as a tilt whose weight is the reward itself, up to normalization:

$$w(x_0,c) \propto R(c,x_0), \qquad z(x_t,c,t) \propto \alpha(x_t,c). \tag{34}$$

This is already the substance of the claim, since the weight is affine in the reward rather than exponential in it, but it is worth seeing the velocity-level statement as well. Applying equation 6 to the pair $(p_t^+, p_t^{\mathrm{old}})$ and canceling the x_t-independent denominator of equation 33,

$$v^+(x_t, c, t) - v^{\mathrm{old}}(x_t, c, t) = -\frac{t}{1-t}\,\nabla_{x_t} \log \frac{p_t^+(x_t \mid c)}{p_t^{\mathrm{old}}(x_t \mid c)} = -\frac{t}{1-t}\,\nabla_{x_t} \log \alpha(x_t, c), \tag{35}$$

and expanding the remaining gradient exactly as in appendix B (differentiate under the integral, then substitute equation 30) yields a weight affine in the reward, $v^* = v^{\mathrm{old}} + \mathbb{E}[(R/\alpha - R_b)(u_t - v^{\mathrm{old}})]$, with R_b free through the zero-mean property of the residual. The denominator here is the posterior quantity $\alpha(x_t, c) = \mathbb{E}_{x_0 \sim p^{\mathrm{old}}(\cdot|x_t,c)}[R]$ of equation 33, which is the z of equation 9 for the weight $w = R$. Replacing it by its endpoint counterpart $\bar{R}(c)$ is the same substitution that §3.2 makes for the exponential tilt and appendix F for the projected one, and absorbing the constant into R_b leaves a weight linear in the reward with no posterior quantity in it. The cost is the factor $r_t = \alpha(x_t, c)/\bar{R}(c)$ proved by appendix D, a rescaling of the step length that leaves its direction unchanged. Setting $R_b = \bar{R}(c)$ and scaling by $1/(\beta C \sigma)$ then arrive at equation 11, the group-centered advantage equation 19 at temperature $\beta C \sigma$. The implementation and the construction therefore match: the linear tilt is what DiffusionNFT's own branch model implies, and what is chosen without derivation is only the temperature.

D Normalization at the endpoint and intermediate level

Nothing in this section is specific to one tilt: w is an arbitrary non-negative weight, so the statements below apply verbatim to all three members of the family, with $w = \exp(R/\gamma)$ and $Z = Z(c)$ for the exponential tilt, $w = [R - \tau]_+/\gamma$ and $Z = 1$ for the projected linear tilt, and $w = R$ with $Z = \bar{R}(c)$ for DiffusionNFT's dense linear tilt. This is the step the three share, and the only one whose cost has to be argued rather than derived.

Throughout this section, c and $t \in (0, 1)$ are fixed and suppressed where possible. Write $q(x_t|x_0)$ for the Gaussian noising kernel of equation 22, $w = w(x_0)$ for the tilt, and

$$Z \;=\; \mathbb{E}_{x_0 \sim p^{\mathrm{old}}(\cdot|c)}[w], \qquad z(x_t) \;=\; \mathbb{E}_{x_0 \sim p^{\mathrm{old}}(\cdot|x_t,c)}[w]. \tag{36}$$

Step 1: the ratio of the two normalizers is the density ratio at level t. Marginalizing the tilted endpoint law equation 4 through the kernel and inserting $p_t^{\mathrm{old}}(x_t)$,

$$\begin{aligned} p_t^*(x_t) &= \int q(x_t|x_0)\, p^{\mathrm{old}}(x_0)\, \frac{w}{Z}\, \mathrm{d}x_0 \;=\; \frac{p_t^{\mathrm{old}}(x_t)}{Z} \int \frac{q(x_t|x_0) p^{\mathrm{old}}(x_0)}{p_t^{\mathrm{old}}(x_t)}\, w\, \mathrm{d}x_0 \\ &= \frac{p_t^{\mathrm{old}}(x_t)}{Z}\, \mathbb{E}_{x_0 \sim p^{\mathrm{old}}(\cdot|x_t)}[w] \;=\; p_t^{\mathrm{old}}(x_t)\, \frac{z(x_t)}{Z}, \end{aligned} \tag{37}$$

where the third equality is Bayes' rule. Hence

$$r_t(x_t) \;:=\; \frac{z(x_t)}{Z} \;=\; \frac{p_t^*(x_t|c)}{p_t^{\mathrm{old}}(x_t|c)} \;>\; 0, \tag{38}$$

which is the quantity whose logarithmic gradient equation 9 is built from. Positivity is immediate since $w > 0$.

Step 2: replacing z by Z rescales the displacement. Let $\delta = u_t(x_t|x_0) - v^{\mathrm{old}}(x_t, c, t)$. We assume throughout this step that v^{old} is the exact marginal field of p_t^{old}, so that

$$\mathbb{E}\big[\delta \,\big|\, x_t\big] \;=\; \mathbb{E}_{x_0 \sim p^{\mathrm{old}}(\cdot|x_t,c)}\big[u_t(x_t|x_0)\big] - v^{\mathrm{old}}(x_t, c, t) \;=\; 0. \tag{39}$$

For any baseline b and any positive λ not depending on x_0,

$$\mathbb{E}\Big[\Big(\tfrac{w}{\lambda} - b\Big)\delta \,\Big|\, x_t\Big] = \tfrac{1}{\lambda}\,\mathbb{E}\big[w\,\delta \,\big|\, x_t\big] \;-\; b\,\underbrace{\mathbb{E}\big[\delta \,\big|\, x_t\big]}_{=\,0} = \tfrac{1}{\lambda}\,\mathbb{E}\big[w\,\delta \,\big|\, x_t\big], \tag{40}$$

so the conditional mean is proportional to λ^{-1} and excludes b. Taking $\lambda = z(x_t)$ recovers the exact displacement of equation 9, $v^* - v^{\mathrm{old}} = z^{-1}\mathbb{E}[w\delta|x_t]$. The implemented target is the bracketed

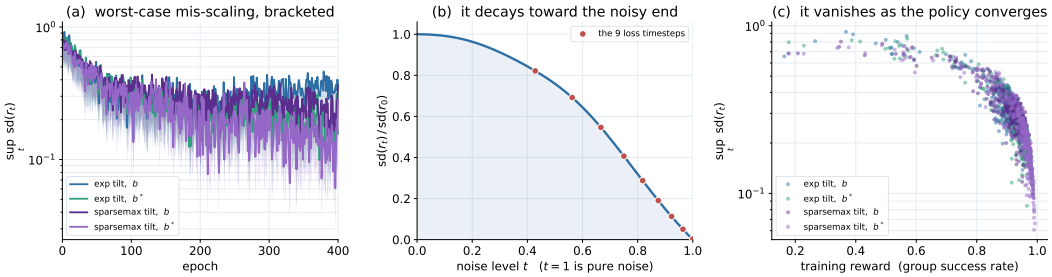

Figure 5: **How far the implemented target is from the exact one.** By equation 45 the rescaling r_t of equation 14 has mean one at every noise level and variance bounded by $\mathbb{E}[\mathcal{A}^2]$, a property of the reward group alone. **(a)** That bound, measured on four runs and plotted as a bound on the standard deviation, so every quantity is shown as a square root. No reward model is presumed. For any centered $\mathcal{A}$, $(\mathbb{E}|\mathcal{A}|)^2 \le \mathbb{E}[\mathcal{A}^2] \le \max_i |\mathcal{A}_i|\, \mathbb{E}|\mathcal{A}|$. The band is that bracket and the line its upper end. **(b)** The decay of $\mathrm{sd}(r_t)$ with t, computed in a closed form for a Gaussian-mixture endpoint law, which is exact because both exponential tilting and Gaussian smoothing preserve the family. Dots mark the first nine timesteps a $T{=}10$ trajectory-coupled buffer supplies under the $\mathrm{shift} = 3$ schedule. When averaged over them, the dispersion is 35% of its worst case. It is a surrogate law and not the model, so this panel is illustrative whereas (a) and (c) are measured. **(c)** The same bound against the reward it was measured at. As the policy converges the group becomes homogeneous, the advantages shrink and the approximation becomes exact in the limit that matters.

quantity of equation 15, namely $y = v^{\mathrm{old}} + \hat{\mathcal{A}}\delta$ with $\hat{\mathcal{A}} = w/Z - b$. Conditionally on x_t the anchor $v^{\mathrm{old}}(x_t, c, t)$ is deterministic and leaves the expectation, so $\mathbb{E}[y|x_t] - v^{\mathrm{old}} = \mathbb{E}[(w/Z - b)\delta \,|\, x_t]$, and this is equation 40 evaluated at $\lambda = Z(c)$. Dividing by the same quantity at $\lambda = z(x_t)$,

$$\mathbb{E}[y|x_t] - v^{\mathrm{old}} = \frac{1}{Z}\,\mathbb{E}[w\delta|x_t] = \frac{z(x_t)}{Z} \cdot \frac{1}{z(x_t)}\,\mathbb{E}[w\delta|x_t] = r_t(x_t)\,\big(v^* - v^{\mathrm{old}}\big), \tag{41}$$

which is equation 14. Two remarks follow immediately. First, the factor r_t is a positive scalar, so the direction of the displacement is preserved exactly and only its length changes. Second, since equation 40 holds for every b, no choice of baseline can repair or worsen the discrepancy. In other words, the baseline controls variance, never this bias.

Note that in practice, the scope of this statement deserves revising. Replacing the posterior mean $z(x_t, c, t)$ by the population endpoint normalizer $Z(c)$ rescales the displacement by a known positive factor, which is the subject of equation 14. The second moment of this factor appears in equation 45, and at $t = 0$ the posterior degenerates, so $r_0 = w/Z$ and $\mathrm{Var}(r_0) = \mathbb{E}[(w/Z - 1)^2]$, which is $\mathbb{E}[\mathcal{A}^2]$ at $b = 1$. Replacing $Z(c)$ in turn by the group average $\hat{Z} = \frac{1}{G}\sum_j w_j$ leads to equation 13, the self-normalized estimator in our implementation. $\hat{Z}$ is unbiased for Z, but $\kappa_i = w_i/\hat{Z}$ is a ratio in which $\hat{Z}$ sits in its own denominator, so the conditional expectation does not pass through it and an $\mathcal{O}(1/G)$ bias term remains. At the default setting $G = 24$ the bias is not negligible, and we do not claim it is entirely absorbed by the rescaling. The bound is therefore a statement about Z, while the diagnostics in figure 5 report $\hat{Z}$, and the two differ by the $\mathcal{O}(1/G)$ term above. The b^* of equation 23 bears a term of the same kind at the smaller micro-batch size.

Step 3: a logarithmic derivative becomes a plain one. Substituting equation 38 into the score-to-residual identity equation 30 used in appendix B gives the exact target as

$$v^* - v^{\mathrm{old}} \;=\; -\frac{t}{1-t}\,\nabla_{x_t} \log r_t(x_t), \qquad \mathbb{E}[y|x_t] - v^{\mathrm{old}} \;=\; -\frac{t}{1-t}\,\nabla_{x_t} r_t(x_t), \tag{42}$$

the second following from the first by multiplying by r_t, as Step 2 instructs, since $r_t \nabla \log r_t = \nabla r_t$. The exact update follows the gradient of $\log r_t$, while the implemented one follows the gradient of r_t. They agree in direction everywhere and in magnitude whenever $r_t = 1$.

Step 4: the dispersion of r_t is monotone in the noise level. Rather than expand near $t = 1$, note that the whole path is a smoothing kernel. Dividing $x_t = (1-t)x_0 + t\epsilon$ by $(1-t)$ yields

$$\frac{x_t}{1-t} \;=\; x_0 + \sigma\epsilon, \qquad \sigma \;=\; \frac{t}{1-t} \quad \text{strictly increasing on } (0,1), \tag{43}$$

so up to a deterministic, x_0-independent rescaling the noising is Gaussian smoothing at level σ. A deterministic invertible rescaling changes both densities by the same Jacobian and therefore leaves their ratio unchanged, so r_t equals the density ratio of p^* against p^{old} after smoothing at level $\sigma(t)$. Gaussian smoothing is a Markov semigroup, for $\sigma_2 > \sigma_1$ the law at σ_2 is obtained from the law at σ_1 by convolution with $\mathcal{N}(0, (\sigma_2^2 - \sigma_1^2)I)$. Two consequences follow. First,

$$\mathbb{E}_{p_t^{\text{old}}}\big[r_t\big] \;=\; \int \frac{p_t^*}{p_t^{\text{old}}}\, p_t^{\text{old}} \;=\; \int p_t^* \;=\; 1\,, \forall\, t$$

so the rescaling is centered at one and cannot introduce a systematic inflation or contraction of the step. Second, $\mathrm{Var}_{p_t^{\text{old}}}(r_t) = \chi^2(p_t^* \| p_t^{\text{old}})$, and since the χ^2 divergence is an f-divergence with $f(u) = (u-1)^2$, the data-processing inequality applied to the convolution kernel gives

$$\chi^2\big(p_{t_2}^* \,\|\, p_{t_2}^{\text{old}}\big) \;\le\; \chi^2\big(p_{t_1}^* \,\|\, p_{t_1}^{\text{old}}\big),\, \forall\, t_2 > t_1, \tag{44}$$

with supremum at $t \to 0$, where the smoothing disappears and the ratio reduces to the endpoint ratio w/Z itself:

$$\sup_{t\in(0,1)} \mathrm{Var}_{p_t^{\text{old}}}\big(r_t\big) \;=\; \mathrm{Var}_{x_0\sim p^{\text{old}}(\cdot|c)}\Big(\frac{w}{Z}\Big) \;=\; \mathbb{E}\big[\mathcal{A}^2\big], \tag{45}$$

using $\mathcal{A} = w/Z - 1$ and $\mathbb{E}[\mathcal{A}] = 0$. This is precise and non-asymptotic, and it bounds the discrepancy by a statistic of the reward group that any run already records, which is what figure 5 exploits. It also localizes the error correctly. The substitution is worst at the clean end of the timestep schedule and asymptotically exact at the noisy end, so a trajectory-coupled buffer, whose timesteps concentrate near $t = 1$, samples it where it is smallest.

Consequence for the minimizer. Appendix E shows that drawing a single (t, ϵ) pair leaves the minimizer of the regression untouched. That argument applies to whichever conditional mean the target happens to have. Taking equation 14 into account, the objective equation 15 actually implemented is minimized, over an unconstrained field, at

$$v_\theta(x_t, c, t) \;=\; v^{\text{old}}(x_t, c, t) + r_t(x_t)\,\big(v^*(x_t, c, t) - v^{\text{old}}(x_t, c, t)\big), \tag{46}$$

not at v^*. The gap is a pure step-length modulation and it is not removable by tuning b, but equation 45 bounds its dispersion by $\mathbb{E}[\mathcal{A}^2]$ uniformly in t. Two practical readings follow. Where the tilted law already puts more mass than the sampling law, $r_t > 1$ and the update is amplified, and where it puts less, the update is damped. And because r_t multiplies the entire displacement, it acts exactly as a state-dependent learning rate, which is why the loss scale s of equation 57 and the temperature γ are the two most sensitive knobs in §4.2.

Remark. The substitution also has a reading that renders it less like an approximation but more like a choice. Since $p^*(x_0|c)/p^{\text{old}}(x_0|c) = w/Z(c)$ by equation 4, the implemented advantage is $\hat{\mathcal{A}} = p^*(x_0|c)/p^{\text{old}}(x_0|c) - 1$, the importance weight of the tilted endpoint law against the sampling law, centered by one, consistent with the weight of Advantage-Weighted Regression (AWR) (Peng et al., 2019). The substitution is in fact avoidable, yet at another price. Taking logarithms in equation 4 turns $Z(c)$ into an additive, prompt-only term that any within-group centering removes exactly, which is the route of Zhu et al. (2026) and requires estimating the log-ratio of two diffusion models through an ELBO surrogate. Working in the velocity field instead never forms a likelihood ratio, and r_t is what is paid for that. By equation 14 the flow-optimal target differs from plain AWR by exactly this factor, which converts the endpoint-level ratio into the level-t ratio.

E WHY THE PER-SAMPLE TARGET MAY REPLACE THE EXACT ONE

Equation (9) expresses the optimal field as a posterior expectation, so the ideal training objective is,

$$\mathcal{L}_{\text{oracle}}(\theta) = \mathbb{E}_{t,\, x_t\sim p_t^{\text{old}}}\Big[\big\|v_\theta(x_t, c, t) - v^*(x_t, c, t)\big\|^2\Big]\,.$$

Evaluating $v^*(x_t, c, t)$ at a single x_t would require averaging over all endpoints consistent with it. The objective actually used replaces that average by one draw,

$$\mathcal{L}_{\text{proxy}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}[0,1],\,\epsilon\sim\mathcal{N}(0,I),\,x_0\sim p^{\text{old}}(\cdot|c)}\Bigg[\Big\| v_\theta(x_t,c,t) - \Big(v^{\text{old}}(x_t,c,t) + \mathcal{A}(x_0)\cdot\big(u_t(x_t|x_0) - v^{\text{old}}(x_t,c,t)\big)\Big)\Big\|^2\Bigg], \tag{47}$$

which is computable, since every term is a function of the pair (x_0, x_t) at hand.

Substituting a draw for an expectation is legitimate here for the same reason it is legitimate in flow matching itself (Lipman et al., 2023). Writing $y(x_0, x_t)$ for the bracketed target of equation 47 and conditioning on x_t,

$$\mathbb{E}\big[\|v_\theta - y\|^2 \,\big|\, x_t\big] = \big\|v_\theta - \mathbb{E}[y \mid x_t]\big\|^2 + \underbrace{\mathbb{E}\big[\|y - \mathbb{E}[y \mid x_t]\|^2 \,\big|\, x_t\big]}_{\text{independent of }\theta}, \tag{48}$$

and $\mathbb{E}[y \mid x_t] = v^*(x_t, c, t)$ is by definition in equation 9. Integrating equation 48 over x_t thus yields

$$\mathcal{L}_{\text{proxy}}(\theta) \;=\; \mathcal{L}_{\text{oracle}}(\theta) \;+\; C, \qquad C \;=\; \mathbb{E}_{t,x_t}\Big[\,\text{Var}\left(y \,\middle|\, x_t\right)\Big] \;\perp\; \theta, \tag{49}$$

so the two objectives differ only by a θ-independent constant, their gradients coincide, and minimizing the tractable one drives v_θ to v^*.

This section closes with two concluding remarks. First, the constant C is the conditional variance of the stochastic target, so it does not bias the optimum but does set the variance of the gradient estimator. This is the quantity that the baseline in equation 23 intends to reduce, and §4.2 examines how much that choice matters in practice. Second, because the argument uses only the tower property, it is indifferent to which tilt supplies the reweighting. It applies verbatim to the exponential and linear weightings of §3. The practical consequence is that reward maximization is carried out entirely by supervised regression on reweighted rollouts, with no likelihood ratio, no stochastic sampler, and no differentiation through the reward.

F Derivation of the χ^2 optimum and its group estimator

This section completes the three steps that §3.3 states without proof: the KKT solution of the χ^2-constrained problem, the intermediate marginals it induces, and the exact group estimator together with its $\mathcal{O}(G \log G)$ solution.

Step 1: constrained optimum. Fix a prompt c and write $\rho(x_0) = \frac{\pi_\theta(x_0|c)}{p^{\text{old}}(x_0|c)}$ as in equation 2. Problem equation 16 reads

$$\max_{\rho}\;\; J[\rho] \;=\; \mathbb{E}_{p^{\text{old}}}[\rho R] - \frac{\gamma}{2}\,\mathbb{E}_{p^{\text{old}}}\big[(\rho-1)^2\big] \qquad \text{subject to} \qquad \rho \in \mathcal{K}, \tag{50}$$

with $\mathcal{K}$ the feasible set defined in §3.3. The map $\rho \mapsto J[\rho]$ is strictly concave, being the sum of a linear term and the strictly concave $-\frac{\gamma}{2}(\rho-1)^2$, and $\mathcal{K}$ is convex and contains $\rho \equiv 1$. The maximizer therefore exists and is unique, and the KKT conditions are necessary and sufficient. Introducing $\lambda \in \mathbb{R}$ for the normalization and $\mu(x_0) \geq 0$ for the non-negativity constraint,

$$\mathcal{L}[\rho, \lambda, \mu] = \mathbb{E}_{p^{\text{old}}}\big[\rho R - \tfrac{\gamma}{2}(\rho-1)^2\big] - \lambda\Big(\mathbb{E}_{p^{\text{old}}}[\rho] - 1\Big) + \mathbb{E}_{p^{\text{old}}}[\mu\rho]\,, \tag{51}$$

pointwise stationarity $\delta\mathcal{L}/\delta\rho = 0$ gives

$$R(c,x_0) - \gamma\big(\rho(x_0)-1\big) - \lambda + \mu(x_0) = 0 \qquad\Longrightarrow\qquad \rho(x_0) = 1 + \frac{R(c,x_0) - \lambda + \mu(x_0)}{\gamma}. \tag{52}$$

Complementary slackness $\mu\rho = 0$ with $\mu \geq 0$ leaves two possibilities at each x_0. Where $\rho > 0$ we must have $\mu = 0$, so $\rho = 1 + (R - \lambda)/\gamma$, and this is consistent only if $R > \lambda - \gamma$. Where the

constraint binds, $\rho = 0$ and $\mu = \lambda - \gamma - R \geq 0$, i.e. $R \leq \lambda - \gamma$. Writing $\tau = \lambda - \gamma$ and combining the branches,

$$\rho^\star(x_0) = \left[1 + \frac{R(c, x_0) - \lambda}{\gamma}\right]_+ = \frac{1}{\gamma}\Big[R(c, x_0) - \tau\Big]_+, \tag{53}$$

which is indeed equation 17. Imposing $\mathbb{E}_{p^{\text{old}}}[\rho^\star] = 1$ yields equation 18 for τ. Uniqueness of its root stems from the fact that $\Phi(\tau) = \mathbb{E}_{p^{\text{old}}}[(R - \tau)_+]$ is convex (a supremum of affine functions of τ), non-increasing, strictly decreasing wherever $\Pr[R > \tau] > 0$, with $\lim_{\tau \to -\infty} \Phi(\tau) \to \infty$ and $\Phi(R_{\max}) = 0$. If the truncation is inactive, $(R - \tau)_+ = R - \tau$ almost surely and equation 18 becomes $\bar{R} - \tau = \gamma$, giving $\tau = \bar{R} - \gamma$ and hence $\lambda = \bar{R}$. Substituting into equation 53 recovers equation 19. Inactivity requires $\rho^\star > 0$ everywhere, i.e. $\operatorname{ess\,inf} R > \bar{R} - \gamma$, which is equation 20.

The group-size bound quoted in §3.3 follows from this condition. Writing $z_i = (R_i - \bar{R})/\sigma$ for the reward standardized within the group, with σ the population standard deviation, $\sum_i z_i = 0$ and $\sum_i z_i^2 = G$ force $|z_i| \leq \sqrt{G - 1}$, with equality when one sample differs and the remaining $G - 1$ coincide. DiffusionNFT's weight is equation 19 with $\mathcal{A}_i = z_i/(\beta C)$, so $\min_i \mathcal{A}_i > -1$ is implied by $\beta C \geq \sqrt{G - 1}$, that is $G \leq 1 + (\beta C)^2$, and the implication is tight: a one-hot group at $G > 1 + (\beta C)^2$ violates it. The condition involves only the temperature and the group size, never the rewards. The bound is a worst case over reward configurations and the quantifier matters: exceeding it withdraws the guarantee rather than asserting a violation, and whether a particular run is in fact feasible is addressed by monitoring $\min_i \mathcal{A} > -b$, as §4.2 does. The default configuration sets $\beta C = 5$ and $G = 24$, inside that bound with a margin of 4% in the temperature, the ceiling in G being 26: DiffusionNFT's heuristic keeps itself feasible by construction, but only at its own group size and without saying so, whereas the projection is feasible at every temperature and every G.

Step 2: intermediate marginals. The threshold τ is determined by the terminal marginal $p^{\text{old}}(\cdot|c)$ and is therefore a constant, not a function of x_t. Specializing equation 25 to $w = \rho^\star$ gives

$$\begin{aligned} p_t^*(x_t|c) &= p_t^{\text{old}}(x_t|c) \cdot z(x_t, c, t), \\ z^\star(x_t, c, t) &= \mathbb{E}_{x_0 \sim p^{\text{old}}(\cdot|x_t, c)}\big[\rho^\star(x_0)\big] = \frac{1}{\gamma}\,\mathbb{E}_{x_0 \sim p^{\text{old}}(\cdot|x_t, c)}\big[(R - \tau)_+\big]. \end{aligned} \tag{54}$$

No *endpoint* partition function appears, because equation 18 already normalizes $\rho^\star$. Here we have $Z = \mathbb{E}_{p^{\text{old}}(\cdot|c)}[\rho^\star] = 1$ exactly, whereas the exponential tilt must estimate its $Z(c)$. The posterior normalizer $z^\star$ is a different matter. The truncation prevents the expectation from passing through the tilt, so $z^\star$ has no closed form in terms of $\bar{R}(x_t)$, and Step 3 does not compute it at all. Normalizing on the group is normalizing at the endpoint level, so the implemented advantage $\rho_i - b$ is equation 9 with $z^\star$ set to 1. This is the same substitution §3.2 makes for the exponential tilt, and it is the one step of the construction that every member of the family shares. Its cost is characterized in appendix D and takes a simpler form here. Since $Z = 1$, the factor $r_t = z/Z$ of equation 14 is $z^\star$ itself, which is exactly the ratio p_t^*/p_t^{old} appearing in equation 54. The direction of the update is therefore preserved and only its length is rescaled, by a factor that is centered at one and tends to one at the noisy end of the schedule. Substituting equation 54 into equation 6 and reusing the score-to-residual identity equation 30, which is independent of the tilt, yields equation 15 with $\mathcal{A}_{\chi^2} = \rho^\star/z^\star - b$, the χ^2 counterpart of the optimal velocity field.

Step 3: the group estimator is a simplex projection. Replace $p^{\text{old}}(\cdot|c)$ by the empirical measure on a group $\{x_0^{(i)}\}_{i=1}^G$, so each atom carries mass $1/G$, and let $w_i = \rho_i/G$ be the probability assigned by π_θ. The constraints $\mathbb{E}[\rho] = 1$ and $\rho \geq 0$ become $\sum_i w_i = 1$ and $w_i \geq 0$, that is $w \in \Delta^{G-1}$, and equation 50 becomes

$$\max_{w \in \Delta^{G-1}} \ \langle w, R\rangle - \frac{\gamma G}{2}\left\|w - \tfrac{1}{G}\mathbf{1}\right\|^2 \iff \min_{w \in \Delta^{G-1}} \ \left\|w - \left(\tfrac{1}{G}\mathbf{1} + \tfrac{R}{\gamma G}\right)\right\|^2, \tag{55}$$

where the equivalence follows by expanding the square and discarding the terms independent of w. The right-hand problem is by definition the Euclidean projection onto the simplex, i.e. $\mathrm{sparsemax}$ operator of (Martins & Astudillo, 2016), which establishes equation 21. Its solution is obtained by sorting (Duchi et al., 2008): let $R_{(1)} \geq \cdots \geq R_{(G)}$ and

$$\tau_k = \frac{1}{k}\left(\sum_{j \leq k} R_{(j)} - \gamma G\right), \qquad k^\star = \max\big\{\, k : R_{(k)} > \tau_k \,\big\}, \qquad \tau = \tau_{k^\star}, \tag{56}$$

with $k^\star$ being the support size. The set in equation 56 is never empty, since $R_{(1)} > \tau_1 = R_{(1)} - \gamma G$ for any $\gamma > 0$, so τ is always defined. The cost is $\mathcal{O}(G \log G)$, dominated by the sort. Finally, because τ is chosen so that $\sum_i \rho_i = G$, the empirical partition satisfies $\hat{z} = \frac{1}{G}\sum_i \rho_i = 1$ exactly, which is why z need not be estimated. Note that this identity holds by construction of τ, and not, as in the dense case, because the centered weights happen to average to one.

G Implementation and experimental details

Every run in §4 adopts the following setup inherited from DiffusionNFT, although it may place our methods at a disadvantage.

Backbone and adapters. SD3.5-Medium at 512^2 pix resolution, loaded in `fp16` mixed precision. Only LoRA (Hu et al., 2022) adapters are trained, at rank $r = 32$ with scaling $\alpha = 64$, while the backbone is frozen. This makes the frozen-anchor variants of §4.2 inexpensive, since the pretrained field $\boldsymbol{f}_{\text{ref}}$ is recovered by disabling the adapters, so no second copy of the weights is held in memory.

Prompt pool and evaluation set. Both splits are the GenEval metadata shipped with the DiffusionNFT codebase. The training pool holds $50{,}000$ records spanning $33{,}199$ distinct prompts and the evaluation split holds $2{,}212$ records over the 553 distinct prompts of the GenEval benchmark, four records per prompt, and is evaluated in full at every evaluation round rather than subsampled.

Rollout and buffer. An epoch draws 48 prompts and samples each $G = 24$ times, for $1{,}152$ images per epoch. The prompt count follows from the sampler rather than being set directly: with 8 devices, a per-device batch of 9 and $G = 24$ images per prompt, each batch covers $8 \times 9/24 = 3$ prompts, and an epoch runs 16 such batches. Sampling uses a $T = 10$ step first-order Euler solver on the flow-matching schedule with shift 3.0, and is CFG-free (Ho & Salimans, 2021). Each buffer is consumed by a single optimization iteration before the next rollout, so no endpoint is revisited and the frozen-buffer premise of §2.1 holds. By default the loss is evaluated at the timesteps the rollout itself visited, which is the coupling that axis **(iv)** isolates. A fraction 90% of the schedule is retained, which merely excludes the degenerated endpoint.

Optimization. AdamW (Loshchilov & Hutter, 2019) with $\beta = (0.9, 0.999)$, $\epsilon = 10^{-8}$ and weight decay 10^{-4} is utilized, at a constant learning rate of 3×10^{-4}. The micro-batch size is 9 latents per device with 16 gradient-accumulation steps. Gradients are clipped to unit norm and an exponential moving average of the adapters is maintained. Both are kept fixed across all runs, so the gradient norms reported in §4.2 are the pre-clipping values and remain comparable between configurations. The loss scale is $s = 5.0$, which is also the s entering η_{eff} in equation 57. For the exponential tilt, γ is set adaptively to the reward standard deviation pooled across the batch and floored at 0.01, and no advantage clipping is applied unless an ablation asks for it. All runs use seed 42 for reproducibility.

Evaluation. Evaluation runs every 10 epochs with a 40-step first-order solver, four times the rollout budget, so that a policy cannot appear to improve merely by adapting to a coarse sampler. No run is stopped early on a training episode. Each is trained until it either plateaus or collapses, within a maximum budget of $1{,}000$ epochs, and the horizon shown in a figure is the horizon that was run.

Diagnosis metrics. We also fix three quantities that we record for every run and that turn out to explain most of what we observe in the incoming section. The first is the *effective step size*

$$\eta_{\text{eff}} \;=\; s \cdot \mathbb{E}\big|\mathcal{A}\big|, \tag{57}$$

where s is the loss scale of equation 22. Since the regression target departs from the anchor by exactly $\mathcal{A}\,\delta$, the quantity η_{eff} measures, up to the residual norm, how far each update asks the policy to move; it is the natural common currency for comparing an exponential and a linear tilt, whose advantages have different scales by construction. The second is $\|\boldsymbol{f}_\theta - \boldsymbol{f}_{\text{old}}\|^2$. Here $\boldsymbol{f}_{\text{old}}$ is the exponential moving average of the policy, refreshed once per epoch and maintained in every configuration, so this is the policy–EMA gap: a proxy for how much the policy is currently changing. When the anchor is frozen it is in particular not the distance to the anchor, which is reported separately as $\|\boldsymbol{f}_\theta - \boldsymbol{f}_{\text{ref}}\|^2$, the squared distance to the adapter-disabled pretrained field averaged over the

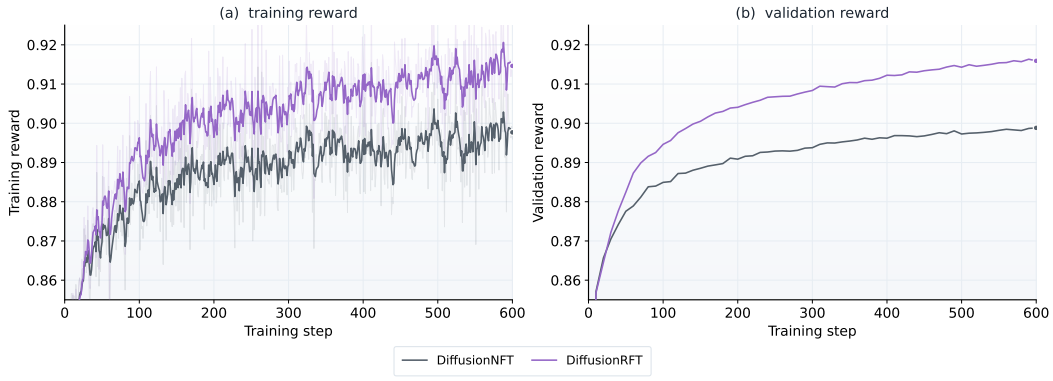

Figure 6: **Generalization to Pickscore reward.** DiffusionRFT achieves consistent superiority over DiffusionNFT, as displayed by training reward (a) and validation reward (b).

batch and over the sampled t with the weights the loss itself uses. We report that second quantity as an implicit KL because it is proportional to the per-step KL under the Gaussian transitions, and all read-outs of both are medians over an explicitly stated epoch window. The third is the fraction of prompt groups whose reward has zero variance, which measures how much of the batch still carries a usable learning signal as the reward saturates.

Generalization experiments. To substantiate the generality and scalability of our method, we replicated the experiments on Pickscore, a learned preference model instead of a rule-based reward. As demonstrated in figure 6, the results are consistent with the main paper. Our method delivers markedly faster convergence and reaches a higher performance ceiling than the competitor.

H Per-run read-outs for the ablations

H.1 The prediction space and the irreducible residual

This section explains figure 2: why the same update, written in $\boldsymbol{v}$ rather than in $\boldsymbol{x}$, leaves an order of magnitude more regression loss at the same movement budget.

The residual carries no reward. Write $\Delta = v_\theta - v^{\text{old}}$ for the displacement the network realizes and $\delta = u_t - v^{\text{old}}$ for the per-sample residual, so that the objective is $L = \mathbb{E}\|\Delta - \mathcal{A}\delta\|^2$. Minimizing over Δ at fixed x_t gives $\Delta^\star = \mathbb{E}[\mathcal{A}\delta|x_t]$ and

$$L^\star \;=\; \mathbb{E}\big[\mathcal{A}^2\|\delta\|^2\big] \;-\; \big\|\mathbb{E}[\mathcal{A}\delta \,|\, x_t]\big\|^2. \tag{58}$$

This is a bias–variance decomposition, in which the two terms answer different questions. The regression target $\mathcal{A}\delta$ is random even after x_t is fixed: it depends on which endpoint x_0 was drawn and on the reward that endpoint received, neither of which v_θ can see. A function of x_t alone can therefore match the conditional mean of the target and nothing better, and the subtracted term $\|\Delta^\star\|^2$ measures how much of the target's size that mean accounts for. Therefore, $\mathbb{E}\|\mathcal{A}\delta - \Delta^\star\|^2$ is the conditional variance of the target: the spread of the individual $\mathcal{A}\delta$ around the direction the network moves in. No parameterization and no amount of training removes it, so it sets the floor that a loss curve descends towards, and comparing two runs by their loss is comparing their floors.

The first term of equation 58 factorizes when $\mathcal{A}^2$ and $q = \|\delta\|^2$ are close to uncorrelated. This is a second-order condition and it does not follow from the small optimal baseline, which constrains $\mathbb{E}[\mathcal{A}q]$ and hence $\mathrm{Cov}(\mathcal{A}, q)$ instead in §H.3. Probing the within-batch correlation of $\mathcal{A}^2$ against q over a 50-epoch run gives a median of 0.013, within a range of $[-0.086, 0.097]$, and no drift as the reward saturates, the correlation against the reward being -0.105. Note that each logged value aggregates that epoch's micro-batches, so its spread of 0.040 reflects an effective sample size of several hundred rather than a group of 24. Writing $\mathbb{E}[\mathcal{A}^2 q] = \mathbb{E}[\mathcal{A}^2]\,\mathbb{E}[q]\,(1 + \varrho\,\mathrm{cv}(\mathcal{A}^2)\,\mathrm{cv}(q))$ with

ϱ being that correlation, the product misstates the first term by a few percent at the dispersions these runs carry. Then

$$L^\star \;\approx\; \underbrace{\mathbb{E}\big[\mathcal{A}^2\big]}_{\text{a property of the reward group}} \cdot \underbrace{\mathbb{E}\|\delta\|^2}_{\text{a property of the noising kernel}}, \tag{59}$$

It approximates $L^\star$ when the correlation above is small, which is measurable. Figure 2(d) plots the regression loss with the loss scale divided out rather than a floor extracted from it. What the scale does say is that the advantage enters only through $\mathbb{E}[\mathcal{A}^2]$, so with the advantage held comparable across runs the second factor is what remains varied, which is the sense in which that panel is labeled reward-independent.

The prediction space sets that factor. By the affine mapping, $\delta = -\big(x_0 - \hat{x}_0^{\mathrm{old}}\big)/t$, so

$$\mathbb{E}\big[\|\delta\|^2 \,\big|\, x_t\big] \;=\; \frac{\mathrm{Var}\,\big(x_0 \,\big|\, x_t\big)}{t^2}. \tag{60}$$

The numerator is the posterior spread of the endpoint given the noised state, an irreducible quantity of the data and the kernel. For a fixed rollout distribution, this posterior spread cannot be removed by fitting the conditional mean more accurately, and at the noisy end of the schedule it is of the order of the data variance itself. The denominator is decided by the prediction space. Regressing in $\boldsymbol{x}$ means comparing $\hat{x}_0 = x_t - t\,v$ against $x_t - t\,v_{\text{target}}$, so the residual acquires a factor t and the t^{-2} of equation 60 cancels exactly. The $\boldsymbol{x}$ and $\boldsymbol{v}$ objectives share a minimizer and differ only in the weight they afford each noise level. In $\boldsymbol{v}$ the factor stands out, and over the nine timesteps the schedule retains, $t \in [0.429, 1]$, the mean of t^{-2} is 2.08.

Why this is a misdirection and not a short step. A large regression loss admits two understandings, and they call for opposite remedies. Either the network has not yet moved far enough, in which case the loss is an excess over the floor and more optimization removes it, or it has moved as far as the target allows and the loss is the floor itself, in which case more optimization changes nothing and the target has to be reformulated. Equation 58 exactly discriminates these two cases, the excess being what Δ can still absorb and the floor being what it cannot, so the two understandings are distinguished by whether the movement budget has been spent. Figure 2 illustrates that the budget has been spent. The $\boldsymbol{v}$-space run moves as far as the $\boldsymbol{x}$-space one, 0.190 against 0.177 in $\|\boldsymbol{f}_\theta - \boldsymbol{f}_{\text{ref}}\|^2$, and it is handed more signal to move with $\mathbb{E}|\mathcal{A}| = 0.538$ against 0.420. Nonetheless, it ends with $8.9\times$ the residual and less reward, 0.442 against 0.693. A run that has spent its budget and still sits high above its counterpart is not taking short steps.

That said, $8.9\times$ is not itself a pure measure of how much worse the fit is, because by equation 59 the two runs do not share a floor. The floor is a product of two factors and both differ. The advantage factor is $\mathbb{E}[\mathcal{A}^2]_{\boldsymbol{v}}/\mathbb{E}[\mathcal{A}^2]_{\boldsymbol{x}}$, and we do not log the second moment directly. Each run supplies $L \le \mathbb{E}[\mathcal{A}^2] \le U$ with $L = (\mathbb{E}|\mathcal{A}|)^2$ and $U = \max_i |\mathcal{A}_i|\,\mathbb{E}|\mathcal{A}|$, and a ratio of two bracketed quantities is bracketed only across sides, $L_{\boldsymbol{v}}/U_{\boldsymbol{x}} \le \mathbb{E}[\mathcal{A}^2]_{\boldsymbol{v}}/\mathbb{E}[\mathcal{A}^2]_{\boldsymbol{x}} \le U_{\boldsymbol{v}}/L_{\boldsymbol{x}}$, which at epoch 39 gives $[0.69, 4.07]$. The kernel factor is the t^{-2} of equation 60, worth 2.08 over the retained timesteps, so the floor ratio is known only to lie in $[1.45, 8.47]$, and the measured 8.9 nearly falls inside it. The excess over the floor is therefore not identified, and these statistics facilitate a scale analysis rather than a decomposition. That is enough for the conclusion it was meant to support: a regression loss is not comparable across prediction spaces, since here the change of floor alone can account for the entire measured ratio.

One of the two floor factors deserves a comment. The advantage factor is larger for the $\boldsymbol{v}$-space run, but that is downstream of the failure rather than upstream. Its reward is the lower, 0.442 against 0.693, so its groups are more heterogeneous and its advantages the larger. The ratio is 1.2 over the first 15 epochs and 3.3 over epochs 30 to 45. A gap that widens as the reward gap widens is consistent with tracking the outcome rather than causing it, but the ordering alone does not establish that. The t^{-2} factor provides a plausible mechanism for the observed gap, and it acts twice: once explicitly, by weighting the floor, and once implicitly, via the weaker policy that weighting generates.

H.2 Why the dense linear tilt collapses.

The instability of the unregularized linear tilt is not, however, a property of the linear family as such, and §3.3 tells us where to look. Criterion equation 20 states that the centered form equation 19

is the χ^2 optimum only while every relative tilt stays positive, equivalently $\min_i \mathcal{A} > -b$. At the $b = 1$ case used here the quantity to monitor is simply the smallest advantage in the batch. Doing so is decisive: the centered linear tilt spends 86.8% of its training steps outside that regime and reaches $\min_i \mathcal{A} = -2.46$, whereas over 928 steps of exponential-tilt training the bound $\mathcal{A} \geq -1$ is never approached, as the barrier property of f_{KL} requires. Among the three runs the terminal drop is monotone in the effective step size equation 57, $\eta_{\text{eff}} = 0.71, 0.90$ and 2.11 giving drops of 0.001, 0.046 and 0.185. We posit that the collapse is the signature of a violated density constraint, and equation 21 supplies the corresponding repair: retain the projection in place of the dense approximation. We therefore probe the projection-only variant where the sparsemax advantage replaces the centered one, holding the *constant* baseline fixed so that the projection is the only variable that changes. Retaining the projection restores stability on its own: the sparsemax tilt ends 0.001 below its peak against 0.185 for the dense linear tilt.

H.3 How the optimal baseline comes into play.

Two remarks are important regarding the constant b in equation 12, which the projection leaves free and which equation 23 then fixes by minimizing variance. First, $b = 1$ is the centering choice for *every* tilt considered here, and not a convention that happens to suit one of them: the relative tilt κ has group mean exactly one for the exponential weight of equation 13, for the sparsemax weight of equation 21 by the construction of τ, and for the dense linear weight, whose advantage $(R - \bar{R})/\gamma$ in equation 19 is precisely $\kappa - 1$ once $\kappa = 1 + (R - \bar{R})/\gamma$ is recognized. What varies between tilts is therefore not the centering constant but the spread of κ around it. Second, the baseline shift and the simplex projection of §3.3 act on different objects and compose without interference: the projection enforces $\rho \geq 0$ and fixes the feasible set, while b^* is a variance-reduction displacement applied afterwards, unbiased for any constant b because $\mathbb{E}[u_t - v^{\mathrm{old}}] = 0$. Note finally that b^* is estimated on the same micro-batch that it reweights, so the self-normalised control variate involves an $\mathcal{O}(1/B)$ finite-batch bias. With the small micro-batches used here this term is not negligible, and the empirically observed sign of b^* corresponds to a slight systematic pull of the regression target toward the sampled data.

How much variance the control variate removes. The baseline of equation 12 leaves a constant free, and equation 23 spends it on variance: b^* is the additional shift that minimizes the second moment of the regression target. How much it removes has an exact answer. Write $q = \|\delta\|^2$, and recall that $\mathcal{A}$ is already centered, $\frac{1}{G}\sum_i \mathcal{A}_i = 0$. The quantity equation 23 minimizes is then $M(b) = \mathbb{E}\big[(\mathcal{A} - b)^2 q\big]$, a quadratic in b whose vertex is $b^* = \mathbb{E}[\mathcal{A}q]/\mathbb{E}[q]$, which is equation 23 itself. Since $\mathcal{A}$ is centered this is exactly $\mathrm{Cov}(\mathcal{A}, q)/\mathbb{E}[q]$. Evaluating M at the vertex,

$$M(0) - M(b^*) \;=\; \frac{\mathrm{Cov}(\mathcal{A}, q)^2}{\mathbb{E}[q]} \;=\; (b^*)^2\,\mathbb{E}[q], \qquad \frac{M(0) - M(b^*)}{M(0)} \;=\; \frac{\mathrm{Cov}(\mathcal{A}, q)^2}{\mathbb{E}[q]\;\mathbb{E}\big[\mathcal{A}^2 q\big]}. \tag{61}$$

Two features of equation 61 matter later. The reduction is governed by the covariance between the advantage and the squared residual, so it vanishes exactly when the two are uncorrelated, however large either of them is. And $M(0) = \mathbb{E}[\mathcal{A}^2 q]$ is the second moment already at the centering baseline, so the ratio says what b^* adds to $b = 1$ and not what a baseline achieves from scratch.

Putting numbers to it, we notice $|b^*|/\mathbb{E}|\mathcal{A}| \approx 1\%$ in practice. When $\mathcal{A}^2$ and q are close to uncorrelated as stated above, the denominator factorizes as $\mathbb{E}[\mathcal{A}^2]\,\mathbb{E}[q]$, so the relative reduction is about $(b^*)^2/\mathbb{E}[\mathcal{A}^2]$, at most $(|b^*|/\mathbb{E}|\mathcal{A}|)^2 = 10^{-4}$. The reward contribution is correspondingly small, $+0.011 \pm 0.002$ at a matched budget. By the first feature above, b^* is a covariance divided by $\mathbb{E}[q]$, so a small b^* says that the advantage and the squared residual are nearly uncorrelated in these batches. A parameter that removes 10^{-4} of the second moment cannot be what keeps the dense linear tilt from collapsing, and the next subsection asks what it does.

What the control variate does. The grid above is unambiguous on the effect: without a control variate the dense linear tilt ends at 0.718, with one it ends at 0.965. Since variance reduction of order 10^{-4} cannot account for that, the stabilization must come from some other property of b^*, and there are two candidates. It is a random variable, re-estimated on every micro-batch, so it has both a typical size and a tail; either could be doing the work. We separated them with four variants of the dense linear cell, each a single-key change.

Two of the four keep the typical size and remove the tail: one replaces b^* by the constant equal to its logged median, -1.5×10^{-3}, and the other clips $|b^*|$ at a running 90th percentile. One keeps only the tail, zeroing b^* below that quantile. The fourth caps $|b^*|$ at 1% of the batch advantage magnitude. At a matched 390 epochs their terminal evaluation rewards are 0.941, 0.111, 0.847 and 0.000, against 0.718 with no control variate and 0.964 with the optimal one. The constant settles the question. It carries no tail, no per-batch adaptivity and no variation of any kind, and it recovers 90% of the gap.

What the set does establish is a statement about the configuration rather than about b^*. None of these runs restores feasibility, with $\min_i \mathcal{A}$ staying between -1.06 and -1.32 throughout so that equation 20 is violated in all of them, and yet a perturbation worth 0.36% of the advantage scale decides whether training ends at 0.718 or at 0.941. A configuration on which so small a perturbation is decisive is one at a stability boundary, and single-seed runs at such a boundary cannot separate a mechanism from sensitivity to the perturbation itself. That sensitivity is what the projection removes without a control variate at all: the projected tilt holds its peak to within 0.001.

H.4 What confounds anchor-lock with RAM's escape.

The lock is not a property of the frozen anchor alone, and separating what it depends on also explains why RAM itself does not appear to suffer from the lock. Two quantities are confounded in a faithful RAM configuration. The first is how far the target is permitted to move off the anchor. RAM multiplies its advantage by 100 before adding it to the frozen base velocity. At unit multiplier the correction is roughly $40\times$ too weak to displace $\boldsymbol{f}_\theta$ from $\boldsymbol{f}_{\text{ref}}$ at all, which is exactly the regime the three stagnated variants sit in, and exactly the η_{eff} of equation 57 being too small for the displacement the anchor demands. The second is where the anchor sits. RAM trains with classifier-free guidance at scale 2.0, whereas our grid is CFG-free throughout. The base model's own training reward is 0.164 without guidance against 0.410 with it, so the two setups anchor to pretrained behaviors of very different quality. Zhu et al. (2026) identify precisely the second effect, observing that anchoring to the weaker non-CFG conditional model can dominate the reward term and pull the policy towards low-quality regions, and propose keeping the sampler CFG-free while anchoring the regularizer to the CFG-guided model. RAM does not make that separation: it simply guides both, but still enjoys the benefit.

Raising the multiplier does lift the ceiling. Our faithful port reaches 0.711 from its higher starting point, and the official implementation reaches 1.000 in training reward. However, neither survives. Both collapse to zero afterwards, and the diagnostics of figure 3 offer an explanation. With the anchor fixed, the displacement the target demands of a single update grows as the policy improves, while γ contracts towards its floor as the reward saturates, so the same multiplier that was barely sufficient early turns to be far too large late. In the sequel, we shed further light on the lock phenomenon of frozen anchors.

We concentrate on the signal first. Frozen anchors give $\mathbb{E}|\mathcal{A}| = 0.49$ to 0.58 against 0.29 for the rolling anchor, and their reward has not saturated. The zero-variance group fraction stays at 0.000 and the within-group reward standard deviation at 0.40, whereas under the rolling anchor the first rises from 0 to 0.479 and the second falls from 0.485 to 0.094. The configuration that wins is the one running out of signal, and the configuration that keeps its signal is the one that stalls.

Two further quantities separate local change from net displacement. The first is the distance to $\boldsymbol{f}_{\text{old}}$. The frozen speed stays within 2.6 to 3.7×10^{-6} in all three runs, while the rolling speed crosses that ceiling at epoch 8 and ends four orders of magnitude above it. The second is the distance to $\boldsymbol{f}_{\text{ref}}$, which is the quantity we report as an implicit KL and to which it is proportional under the Gaussian transitions. It separates the two configurations with no crossover at all, 1.9 to 4.5×10^{-4} for the frozen runs against 1.5×10^{-1}, and split by fifths of the window it grows $4.4\times$ under the rolling anchor and not at all under a frozen one, ending at 0.3 to 0.8 of where it started. The frozen policy is held in a neighborhood, it is not drifting slowly out of one.

Terminal training rewards for the three frozen variants are 0.249, 0.246 and 0.248, against 0.980 for the rolling-anchor run. For the two RAM implementations, comparing medians over the 80 epochs before the evaluation peak with the 120 after it, we find that the official run moves from 0.98 to 0.92 in reward and from 0.23 to 0.36 in advantage magnitude, while its output-delta norm, a norm rather than the squared distance above, grows $7.2\times$ ($0.031 \to 0.224$), its target norm from 836 to

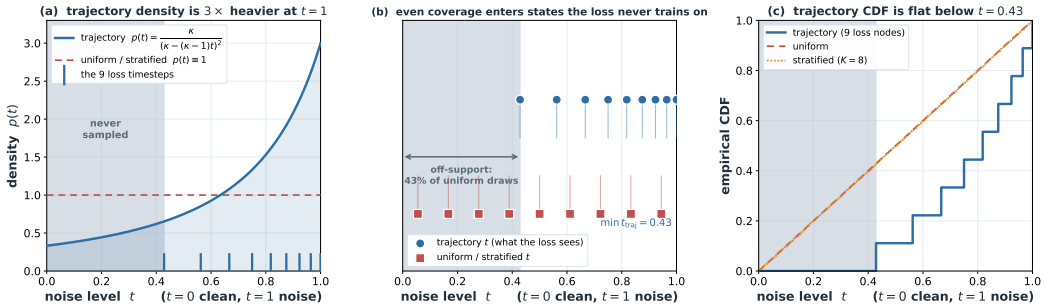

Figure 7: **The trajectory schedule covers only part of the noise axis.** t is the noise level, $t = 0$ being clean data. (a) Pushing the uniform grid through the SD3.5 shift $\kappa = 3$ gives a density $\kappa^2 = 9\times$ heavier at $t = 1$ than at $t = 0$, and the nine loss timesteps inherit it. (b) Those nine nodes never fall below $t = 0.43$, so the shaded band is off-support for the *loss*: drawing $t \sim \mathcal{U}[0, 1]$ sends 43% of the loss evaluations there, and stratification guarantees it. (c) The same statement as a CDF. This is the mechanism behind figure 4: decoupling the renoising timestep does not merely re-weight the noise axis, it regresses at noise levels the policy is never asked to act on.

1437, and its gradient norm $3.4\times$ ($0.46 \to 1.55$). Our reproduction logs the gradient norm but not the displacement, and shows the same runaway in milder form, $1.3\times$ ($0.57 \to 0.76$). The failure is therefore a runaway of the target rather than a loss of signal. The anchor is fixed while $\mathcal{A}\,\delta$ is scaled by 100, the target separates from the field the sampler uses, and the regression amplifies its own error. A rolling anchor removes this feedback path, because the anchor follows the policy and the target cannot separate from the sampling distribution indefinitely.

H.5 Why the decoupled buffer is worse.

The diagnostics in figure 4 are clearly demarcated: both split by the law of t and are almost invariant in K. $\mathbb{E}|\mathcal{A}|$ is 0.118 and 0.105 under the trajectory law against 0.237 and 0.250 under the uniform one, and the zero-variance group fraction is 0.60 and 0.62 against 0.34 and 0.32. The within-batch spread of the displacement, as the ratio of batch maximum to batch median, is 1.3×10^3 and 3.0×10^3 against 2.1×10^3 and 5.1×10^3, rising to 1.1×10^4 once the uniform draws are stratified. What decoupling changes is not the variance of t but its *support*. The sampler walks the 10-step shifted grid of $t \in \{0.43, \ldots, 1.00\}$ in figure 7, so the trajectory schedule never regresses below $t = 0.43$, whereas $t \sim \mathcal{U}[0, 1]$ places 43% of the loss evaluations in a near-clean band the policy scarcely visits when sampling.