ADVANTAGEFLOW: REGULARIZED ADVANTAGE-WEIGHTED RL IN FLOW MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

We present AdvantageFlow, a forward-process reinforcement learning (RL) algorithm for rectified flow models. The algorithm minimizes an advantage-weighted prediction loss, which maximizes reward, regularized by the rollout policy, which convexifies the objective and makes its optimization stable. Our objective can be viewed as fitting a local reward-improving target distribution. The rollout regularization arises as a variance reduction step. We evaluate AdvantageFlow empirically on text-to-image generation with Stable Diffusion 3.5 Medium and FLUX.1, and compare it to both forward- and reverse-process RL algorithms.



Figure 1: Images generated by `AdvantageFlow`, `DiffusionNFT`, and base model (Stable Diffusion 3.5 Medium) with cfg. `AdvantageFlow` improves object generation, counting, color understanding, and position understanding; all without classifier free guidance (cfg). The images are obtained after training for 60 A100 GPU hours.

1 INTRODUCTION

Flow and diffusion models are standard models for generating high-quality images (Lipman et al., 2023; Liu et al., 2023; Albergo et al., 2023). The models are usually pre-trained by minimizing the flow-matching loss and then post-trained to improve human preference alignment, prompt following, text rendering, and image composition. At inference time, images are generated by sampling noise and following a learned denoising trajectory.

Given a pre-trained image generation model and a reward model, we want to adapt the pre-trained model to generate more rewarding images, while preserving its quality and diversity. Most *reinforcement learning (RL)* algorithms for flow models optimize this sampling process by treating image generation as a sequential process along the reverse denoising trajectory and apply a variant of

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.

policy gradients (Black et al., 2024; Fan et al., 2023; Liu et al., 2025a; Clark et al., 2024; Prabhudesai et al., 2023; Domingo-Enrich et al., 2024). These methods are powerful but complex. For instance, they need to introduce additional stochasticity just to compute policy gradients. Moreover, since only the final generated image is rewarded, reward attribution to intermediate denoising steps where mistakes happen must be learned from data. Several recent papers try to improve reward attribution in RL for flow models (Savani et al., 2026; He et al., 2025; Zhou et al., 2025). Forward-process RL offers a simpler alternative. Instead of optimizing along the reverse sampling process, we sample complete images, score them with rewards, and train the model using the same forward-process loss as in pre-training but with reward-dependent weights.

We introduce `AdvantageFlow`, a forward-process RL algorithm for rectified flow models. The key idea is to sample a batch of images per prompt and compute their advantages: how much better or worse each image is relative to the batch average. Then we train the model to generate images with higher advantages more frequently. The optimized objective is a *regularized advantage-weighted prediction loss*. The advantage weighting can be viewed as fitting a local linearly tilted target distribution that shifts probability mass towards more rewarding samples. The regularization arises naturally as a variance reduction step: it replaces noisy per-sample targets with their conditional means in the reward-independent part of the loss. It also ensures that the loss is strictly convex per sample. We evaluate `AdvantageFlow` on text-to-image generation with Stable Diffusion 3.5 Medium and FLUX.1. Our algorithm improves the quality of generated images across all tested reward models. It outperforms both `Flow-GRPO` (Liu et al., 2025a), and state-of-the-art forward-process RL baselines based on negative-aware fine-tuning (Zheng et al., 2026) and advantage-weighted matching (Xue et al., 2025a). To mitigate a potential concern of reward hacking, we complement our evaluation by GenEval (Ghosh et al., 2023).

Our contributions are: (1) We introduce `AdvantageFlow`, a least-squares algorithm for reinforcement learning in flow models using the forward process. (2) We derive the algorithm from a local reward-improving target distribution, where the rollout regularization arises as a variance reduction step for noisy sampled targets. (3) We evaluate `AdvantageFlow` on text-to-image generation with Stable Diffusion 3.5 Medium and FLUX.1. `AdvantageFlow` mostly outperforms all baselines and generates high-quality images.

2 Background

We briefly review flow matching for image generation (Lipman et al., 2023; Liu et al., 2023; Albergo et al., 2023). *Rectified flow matching models* learn continuous-time normalizing flows by training on a linear interpolation of clean images and Gaussian noise. Let c be a prompt and $q(\cdot \mid c)$ be a distribution over clean images. Let $x_0 \sim q(\cdot \mid c)$ be a random clean image, $\epsilon \sim \mathcal{N}(0, I)$ be Gaussian noise, and $t \sim [0, 1]$ be a random time. Then the flow matching objective is minimizing

$$\mathbb{E}_{x_0 \sim q(\cdot|c),\, \epsilon,\, t}[\|v_\theta(x_t, t, c) - (\epsilon - x_0)\|_2^2]\,, \tag{1}$$

where $x_t = (1-t)x_0 + t\epsilon$ is a random interpolation of x_0 and ϵ, $v_\theta(x_t, t, c)$ is the *predicted velocity* at x_t, θ is the *learned model parameter*, and $\epsilon - x_0$ is the ground-truth velocity. The objective (1) can also be viewed as prediction loss minimization. Specifically, the *predicted clean image* at x_t is $f_\theta(x_t, t, c) = x_t - t\, v_\theta(x_t, t, c)$ and therefore $\|f_\theta(x_t, t, c) - x_0\|_2^2 = t^2 \, \|v_\theta(x_t, t, c) - (\epsilon - x_0)\|_2^2$. In this work, our goal is to develop and justify algorithms for minimizing

$$\mathbb{E}_{x_0 \sim q(\cdot|c),\, \epsilon,\, t}[\|f_\theta(x_t, t, c) - x_0\|_2^2] \tag{2}$$

weighted by rewards. Without the rewards, the minimizer is the conditional mean of x_0.

Sampling in flow models is implemented using a *deterministic ODE*. Specifically, it starts with noise x_T and follows a denoising *trajectory* $x_T, x_{T-\Delta t}, \ldots, x_0$, where

$$x_{t-\Delta t} = x_t - v_\theta(x_t, t, c)\, \Delta t\,. \tag{3}$$

The key idea in most existing RL algorithms for flow models, such as `Flow-GRPO`, is treating this trajectory as a sequential process with *reward* $r(x_0, c)$, for final generated image x_0 given prompt c. There are two challenges. The first challenge is that policy gradients (Williams, 1992) cannot be applied to deterministic trajectories, such as the one generated by (3). To address this, `Flow-GRPO`

replaces (3) with an SDE

$$x_{t-\Delta t} = x_t - \left[v_\theta(x_t, t, c) - \frac{\sigma_t^2}{2t} \hat{x}_1 \right] \Delta t + \sigma_t \sqrt{\Delta t} \xi \,, \quad \hat{x}_1 = x_t + (1-t)\, v_\theta(x_t, t, c) \,, \tag{4}$$

where $\xi \sim \mathcal{N}(0, I)$ and σ_t is the noise schedule of the stochastic sampler. This defines a Gaussian transition $\pi_\theta(x_{t-\Delta t} \mid x_t, c)$ at denoising step t with covariance $\sigma_t^2 \Delta t\, I$, allowing policy gradients to be applied. The second challenge is that the reward $r(x_0, c)$ is assigned only to the final generated image, while $\pi_\theta(x_{t-\Delta t} \mid x_t, c)$ needs to be optimized. This creates a reward attribution problem and many recent works try to improve it (Savani et al., 2026; He et al., 2025; Zhou et al., 2025).

Forward-process RL has neither of these problems. It uses the ODE sampler in (3) to generate clean images and thus is better aligned with the inference-time denoising. The clean image is predicted directly using $f_\theta(x_t, t, c)$ and therefore no reward attribution is needed.

3 Algorithm

We introduce `AdvantageFlow`, a forward-process RL algorithm for rectified flow models. The key idea is to minimize the prediction loss in (2) weighted by advantages and regularized by the rollout policy, which convexifies the objective and makes its optimization stable. The *rollout policy* is used for sampling training images. To prevent the loss of capabilities that we do not optimize for, we also regularize by a *reference policy*. We denote predicted clean images under the learned, rollout, and reference policies by f_θ, f_{old}, and f_{ref}, respectively.

3.1 Intuition

Suppose that we sample multiple images per prompt and score them with rewards. Some images are better than the average and some are worse. We want to generate the good images more often than the bad ones. A natural idea is to minimize the prediction loss in (2) weighted by advantages, rewards minus the mean reward, which upweight above-average images and downweight the rest. Specifically, the loss in (2) becomes $A(x_0, c)\, \|f_\theta(x_t, t, c) - x_0\|_2^2$, where $A(x_0, c)$ is the advantage of clean image x_0 given prompt c. Perhaps surprisingly, this is not sufficient. When an image has a negative advantage, $A(x_0, c)\, \|f_\theta(x_t, t, c) - x_0\|_2^2$ becomes negative. This can cause instability and divergence because the loss is not convex in f_θ anymore (Section 3.3).

To get around this issue, we incorporate the advantage through fitting a target distribution that upweights high-advantage samples (Section 4.1). Specifically, we linearly tilt the rollout distribution p_{old}: its density at x_0 is reweighted by $1 + \eta A(x_0, c)$ for $\eta > 0$. We cannot sample from the tilted distribution directly. Therefore, we sample from p_{old} and reweight the samples using importance weights $1 + \eta A(x_0, c)$, as can be seen in (11). When the importance weight is expanded, the loss becomes $\eta A(x_0, c)\, \|f_\theta(x_t, t, c) - x_0\|_2^2 + \|x_0 - f_\theta(x_t, t, c)\|_2^2$, where the latter term can be rewritten in expectation as $\|f_\theta(x_t, t, c) - f_{\text{old}}(x_t, t, c)\|_2^2$ and provably reduces variance (Section 4.2). It can be also interpreted as rollout policy regularization. We also regularize with respect to the reference policy $f_{\text{ref}}(x_t, t, c)$. This prevents the loss of capabilities not captured by rewards.

Putting it all together, our objective is

$$\mathbb{E}_{x_0 \sim p_{\text{old}}(\cdot|c),\, \epsilon,\, t} [\ell_{\theta,\, \theta_{\text{old}}}(A(x_0, c), x_0, x_t, t, c)] \,, \tag{5}$$

where $\ell_{\theta,\, \theta_{\text{old}}}(A(x_0, c), x_0, x_t, t, c) =$

$$A(x_0, c)\, \|f_\theta(x_t, t, c) - x_0\|_2^2 + \gamma\, \|f_\theta(x_t, t, c) - f_{\text{old}}(x_t, t, c)\|_2^2 + \lambda\, \|f_\theta(x_t, t, c) - f_{\text{ref}}(x_t, t, c)\|_2^2 \,.$$

We refer to the three terms as advantage-weighted prediction loss, rollout regularization, and reference regularization. The *advantage-weighted prediction loss* fits the policy to generated images weighted by their advantages. Positive $A(x_0, c)$ pull the policy towards good images and negative $A(x_0, c)$ push it away from the bad images. The *rollout regularization* penalizes deviations from the rollout policy. It ensures that the loss is strictly convex per sample, by making the total quadratic coefficient $A(x_0, c) + \gamma + \lambda$ positive (Section 3.3). The *reference regularization* penalizes deviations from the initial pre-trained model and thus prevents forgetting behaviors not captured by rewards. The weights $\gamma, \lambda \geq 0$ control the amount of rollout and reference regularization, respectively.

Algorithm 1 `AdvantageFlow`

Input: Pre-trained model θ_{ref}, reward model r, prompt dataset $\mathcal{C}$, batch size L, group size K, number of inner steps $M \geq 1$, regularization strengths $\gamma, \lambda \geq 0$, and EMA update rate $\rho \geq 0$
Initialize $\theta \leftarrow \theta_{\text{ref}}$ and $\theta_{\text{old}} \leftarrow \theta_{\text{ref}}$
for each outer iteration **do**
  Sample L prompts $c_1, \ldots, c_L \sim \mathcal{C}$
  For each c_i, generate K images $x_0^{i,1}, \ldots, x_0^{i,K}$ using (3) with rollout velocity v_{old}
  Compute per-image advantage $A^{i,k}$ using (6), for all $(i,k) \in [L] \times [K]$
  for M inner steps **do**
    for all $(i,k) \in [L] \times [K]$ **do**
      $x_t^{i,k} \leftarrow (1 - t^{i,k})\, x_0^{i,k} + t^{i,k} \epsilon^{i,k}$, for freshly sampled $t^{i,k}$ and $\epsilon^{i,k}$
    Update θ by gradient descent on $\frac{1}{KL} \sum_{i=1}^{L} \sum_{k=1}^{K} \ell_{\theta,\, \theta_{\text{old}}}(A^{i,k}, x_0^{i,k}, x_t^{i,k}, t^{i,k}, c_i)$
  Update rollout policy as $\theta_{\text{old}} \leftarrow \rho\, \theta_{\text{old}} + (1 - \rho)\, \theta$
Output: θ

3.2 AdvantageFlow

The pseudo-code of our algorithm is given in Algorithm 1 and we call it `AdvantageFlow`, since it optimizes the flow model using advantages. The algorithm is iterative. In the outer loop, we sample L prompts, generate K images per prompt using the rollout policy, score the images by rewards, and compute their advantages. In the inner loop, we take $M \geq 1$ gradient steps on the learned policy using the same batch of images, before updating the rollout policy. Since the objective (5) is a fixed expectation under the rollout policy p_{old}, multiple inner steps require no importance correction: our approach is off-policy by design.

Rollout policy. The images are generated by the rollout policy, which is parameterized by θ_{old}. The parameter θ_{old} is slowly updated using an *exponential moving average (EMA)* of θ, as in Zheng et al. (2026). Both θ and θ_{old} parametrize velocity models v_θ and v_{old}. Therefore, they can be used to sample images, through (3), or predict clean images, through f_θ and f_{old}.

Advantages. The advantages can be computed in many different ways (Shao et al., 2024; Liu et al., 2025b;a). As in Zheng et al. (2026), we standardize them per-batch as follows. Let $r^{i,k}$ denote the reward for image $k \in [K]$ and prompt c_i, where $i \in [L]$. Then the corresponding advantage is

$$A^{i,k} = \text{clip}\left(\frac{r^{i,k} - \hat{r}^i}{Z}, -1, 1\right), \quad \hat{r}^i = \frac{1}{K}\sum_{k=1}^{K} r^{i,k}, \quad Z = \sqrt{\frac{1}{KL}\sum_{i=1}^{L}\sum_{k=1}^{K}(r^{i,k} - \hat{r}^i)^2}, \tag{6}$$

where $\hat{r}^i$ is the per prompt mean and Z is the batch standard deviation. The advantage of a sample is positive if its reward is above the average and negative if it is below.

Prediction loss minimization. We minimize a finite-sample approximation of (5), where the expectation over prompts, clean images, time, and Gaussian noise is approximated over KL samples in a batch. The images $x_0^{i,k}$ and advantages $A^{i,k}$ are computed once per outer iteration; the time t and noise ϵ are drawn fresh per sample and per inner-loop step.

Computational cost. The time complexity of each outer iteration of `AdvantageFlow` is linear in KL for sampling the images, and linear in MKL for the inner loop, which requires predictions by f_θ, f_{old}, and f_{ref} for each of the KL noisy images at each of the M inner steps.

Space complexity. The space complexity is mostly driven by the dimensionality of θ and θ_{old}. In our work, both are encoded using LoRA (Hu et al., 2022) on the top of the initial pre-trained model, which represents the reference policy.

3.3 Strict Convexity

The loss inside (5) is quadratic in f_θ for any sample, with a *quadratic coefficient* $A(x_0, c) + \gamma + \lambda$. It is strictly convex, and therefore has a unique minimum in the prediction space, when

$$A(x_0, c) + \gamma + \lambda > 0\,. \tag{7}$$

Without rollout regularization, $\gamma = 0$ and negative advantages $A(x_0, c)$ could make the quadratic coefficient negative, thus turning the loss concave and making its optimization unstable.

Since the advantages in (6) are clipped to $[-1, 1]$, any $\gamma > 1$ would guarantee that (7) holds for any sample, irrespective of $\lambda \geq 0$. In our experiments, we set γ *adaptively per prompt*. Specifically, it is $\gamma(c) = A_{\max}(c) + \gamma_0$, where $A_{\max}(c)$ is the maximum absolute value of advantages per prompt c and γ_0 is a small constant that guarantees a minimum level of strict convexity. We show that this choice outperforms a fixed γ in Section 5.

4 Analysis

This section provides theory supporting our design choices in Section 3.1: linear tilting provides a natural-gradient justification for advantage weighting and using the rollout prediction as the regression target reduces variance. Neither result is needed to follow the algorithm.

4.1 Linearly Tilted Target Distribution

Fix a prompt c and let $p_{\text{old}}(\cdot \mid c)$ be the rollout distribution. We define the centered advantage

$$A(x_0, c) = r(x_0, c) - \mathbb{E}_{x_0' \sim p_{\text{old}}(\cdot|c)}[r(x_0', c)]\,. \tag{8}$$

Consider the distribution obtained by tilting p_{old} by a factor linear in the advantage,

$$q_\eta(x_0 \mid c) = (1 + \eta A(x_0, c))\, p_{\text{old}}(x_0 \mid c)\,. \tag{9}$$

It increases the probability of above-average samples ($A(x_0, c) > 0$) and decreases that of below-average samples ($A(x_0, c) < 0$), proportionally to how far each sample is from the mean reward. Since the advantage is centered, $\mathbb{E}_{x_0 \sim p_{\text{old}}(\cdot|c)}[A(x_0, c)] = 0$, the perturbation integrates to zero and q_η is automatically normalized: $\int q_\eta(x_0 \mid c)\, \mathrm{d}x_0 = \int p_{\text{old}}\, \mathrm{d}x_0 + \eta \int A(x_0, c)\, p_{\text{old}}\, \mathrm{d}x_0 = 1 + 0 = 1$. If $1 + \eta A(x_0, c) \geq 0$ for all x_0 (equivalently, the rescaled form of the pointwise positivity condition $A(x_0, c) + \gamma > 0$ from Section 3.3 with $\gamma = 1/\eta$ and $\lambda = 0$), $q_\eta(x_0 \mid c) \geq 0$ and therefore a valid probability distribution. It also improves the expected reward by $\eta \operatorname{Var}_{p_{\text{old}}(\cdot|c)}(r(\cdot, c)) \geq 0$, since $\mathbb{E}_{q_\eta}[r] - \mathbb{E}_{p_{\text{old}}}[r] = \eta\, \mathbb{E}_{p_{\text{old}}}[A(x_0, c)\, r] = \eta\, \mathbb{E}_{p_{\text{old}}}[A(x_0, c)^2] = \eta \operatorname{Var}_{p_{\text{old}}}(r)$, using $\mathbb{E}_{p_{\text{old}}}[A(x_0, c)] = 0$ and the definition of variance. The linear tilt q_η in (9) is the first-order natural-gradient update for maximizing expected reward under the Fisher-Rao metric on probability distributions (Amari, 1998; Kakade, 2001). We make this connection precise.

Proposition 1 (Fisher-Rao natural-gradient direction)**.** *Fix a prompt c and let $F(p) = \mathbb{E}_p[r(\cdot, c)]$. Under the Fisher-Rao metric, the natural-gradient direction of F at $p = p_{\text{old}}(\cdot \mid c)$ is the tangent vector*

$$\delta p(x_0) = A(x_0, c)\, p_{\text{old}}(x_0 \mid c)\,, \qquad A(x_0, c) = r(x_0, c) - \mathbb{E}_{p_{\text{old}}(\cdot|c)}[r(\cdot, c)]\,.$$

A first-order step of size η along this direction yields the additive tilt $q_\eta(\cdot \mid c) = (1 + \eta A(\cdot, c))\, p_{\text{old}}(\cdot \mid c)$ from (9).

The proof is in Appendix B. The Fisher-Rao metric is the canonical Riemannian metric on probability distributions and underlies natural policy gradient methods in reinforcement learning (Kakade, 2001). Proposition 1 carries this picture over to flow models: `AdvantageFlow` can be read as natural-gradient ascent on expected reward, with the rollout distribution as the base point and η as the step size.

Formally, the prediction-loss objective under the tilted distribution is

$$\mathcal{L}_{q_\eta}(\theta) = \mathbb{E}_{x_0 \sim q_\eta(\cdot|c),\, \epsilon,\, t}[\|x_0 - f_\theta(x_t, t, c)\|_2^2]\,. \tag{10}$$

Changing measure to the rollout distribution gives

$$\mathcal{L}_{q_\eta}(\theta) = \mathbb{E}_{x_0 \sim p_{\text{old}},\, \epsilon,\, t}[(1 + \eta A(x_0, c))\, \|x_0 - f_\theta(x_t, t, c)\|_2^2]\,. \tag{11}$$

Suppressing the dependence of A on x_0, c and expanding the weight gives

$$\mathcal{L}_{q_\eta}(\theta) = \underbrace{\mathbb{E}_{x_0 \sim p_{\text{old}},\, \epsilon,\, t}[\|x_0 - f_\theta(x_t, t, c)\|_2^2]}_{\text{reward-independent}} + \eta\, \underbrace{\mathbb{E}_{x_0 \sim p_{\text{old}},\, \epsilon,\, t}[A(x_0, c)\, \|x_0 - f_\theta(x_t, t, c)\|_2^2]}_{\text{reward-dependent}}\,. \tag{12}$$

4.2 Variance Reduction by Rollout Policy

Variance reduction in policy optimization is a well-studied topic (Sutton et al., 2000; Baxter & Bartlett, 2001; Munos, 2006). The reward-independent term is defined on individual samples x_0, but the population minimizer of this term is the conditional mean $\mathbb{E}_{x_0 \sim p_{\text{old}}, \epsilon}[x_0 \mid x_t, t, c]$. If the rollout policy is well-trained, its prediction $f_{\text{old}}(x_t, t, c)$ approximates this conditional mean. We then are better off by replacing x_0 with the prediction $f_{\text{old}}(x_t, t, c)$ in reward-independent term of Equation (12).

Proposition 2 (Variance reduction)**.** *Fix a prompt c and suppose $f_{\text{old}}(x_t, t, c) = \mathbb{E}_{x_0 \sim p_{\text{old}}(\cdot|c), \epsilon}[x_0 \mid x_t, t, c]$. Up to a θ-independent constant,*

$$\mathbb{E}_{x_0 \sim p_{\text{old}},\, \epsilon,\, t}[\|x_0 - f_\theta(x_t, t, c)\|_2^2] = \mathbb{E}_{x_0 \sim p_{\text{old}},\, \epsilon,\, t}[\|f_{\text{old}}(x_t, t, c) - f_\theta(x_t, t, c)\|_2^2]\,.$$

Moreover, the single-sample gradient of the right-hand side has no larger variance than that of the left-hand side.

The proof is in Appendix C. The key observation is that x_0 is a single noisy draw from $p_{\text{old}}(\cdot \mid c)$, while $f_{\text{old}}(x_t, t, c)$ is trained to match its conditional mean. Using f_{old} as the regression target removes sample noise. The population minimizer is unchanged, and the gradient estimator is lower-variance by the Rao-Blackwell argument (Casella & Robert, 1996). Substituting this into (12) and rescaling by $\eta = 1/\gamma$ gives the objective of `AdvantageFlow` in Section 3 without reference regularization,

$$\mathbb{E}_{x_0 \sim p_{\text{old}},\, \epsilon,\, t}[A(x_0, c)\ \|x_0 - f_\theta(x_t, t, c)\|_2^2 + \gamma\ \|f_\theta(x_t, t, c) - f_{\text{old}}(x_t, t, c)\|_2^2]\,. \tag{13}$$

5 Experiments

Most of our experiments are with Stable Diffusion 3.5 Medium (Esser et al., 2024) and 512×512 images, in the `DiffusionNFT` code base[1] (Zheng et al., 2026). We experiment with two kinds of reward models: rule- and model-based. *OCR* is ruled-based and evaluates text rendering. *GenEval* (Ghosh et al., 2023) is ruled-based and evaluates image composition. *PickScore* (Kirstain et al., 2023), *HPSv2.1* (Wu et al., 2023), and *CLIPScore* (Hessel et al., 2021) are learned and evaluate image quality, human preference, and text-to-image alignment. For OCR, we use their training and test sets. For other reward models, we use the Pick-a-Pic dataset (Kirstain et al., 2023).

`AdvantageFlow` is implemented as described in Sections 3.2 and 3.3. We compare it to `Flow-GRPO` (Liu et al., 2025a), arguably the most popular RL algorithm for flow models; and state-of-the-art forward-process RL algorithms `DiffusionNFT` (Zheng et al., 2026) and `AWM` (Xue et al., 2025a). All algorithms are trained on 10 denoising steps (SDE in `Flow-GRPO` and ODE otherwise) and evaluated on 40. The learned and rollout policies are implemented using LoRA with $\alpha = 64$ and $r = 32$. We depart from the setting of Zheng et al. (2026) only slightly by having an effective batch size 512: 32 prompts (1.5 times fewer) with 16 samples per prompt (1.5 times fewer). This leads to almost identical results at less than a half of their computational cost. We run all algorithms for 250 iterations. Each run takes up to 70 A100 GPU hours and all runs can be reproduced in a week on a single 8-GPU A100 node. To reduce the variability of runs, we set the reference regularization strength in all forward-process RL algorithms to $\lambda = 0.001$. No other hyper-parameters have been modified. All reported rewards are measured on test sets and plotted as a function of training time.

Baseline comparison. We start by comparing `AdvantageFlow` to all baselines. Our results across all reward models are reported in Figure 2a-d.[2] We observe two main trends. First, all forward-process RL algorithms outperform `Flow-GRPO` by a large margin, which is consistent with observations in prior works (Zheng et al., 2026; Xue et al., 2025a). Second, `AdvantageFlow` outperforms `DiffusionNFT` in all problems and `AWM` in 3 problems out of 4. `AdvantageFlow` and `AWM` are comparable in the CLIPScore problem (Figure 2c). Some gains are major. For instance, in the HPSv2.1 problem (Figure 2b), `AdvantageFlow` attains the highest reward of `DiffusionNFT` in a quarter of its computational cost. Because the methods differ only in their loss functions, this shows the superiority of our least-squares RL objective. In all remaining experiments, we only compare to the best performing baselines `DiffusionNFT` and `AWM`.

[1] https://github.com/NVlabs/DiffusionNFT

[2] The OCR experiment in Zheng et al. (2026) is with `DiffusionNFT` for $\beta = 0.1$. To get a similar behavior in `AdvantageFlow` (Section 6), we multiply its rollout regularization constant by 0.1.

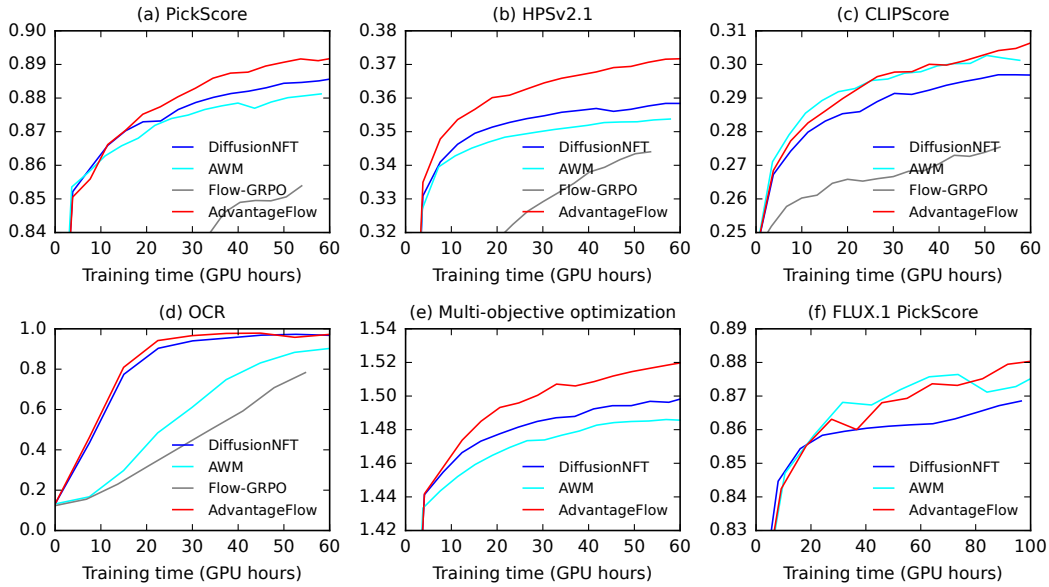

Figure 2: Evaluation on PickScore, HPSv2.1, CLIPScore, OCR, and multi-objective problems.

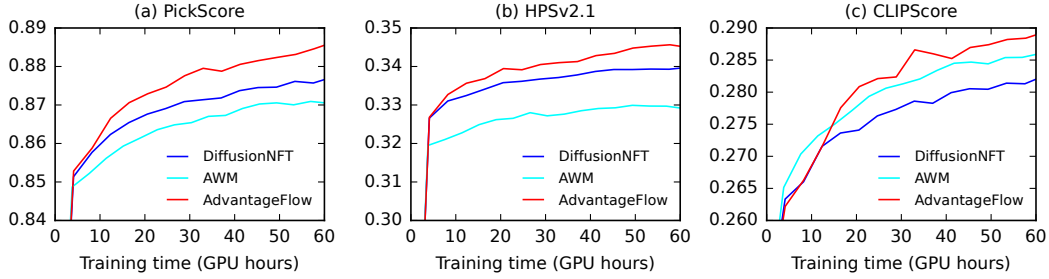

Figure 3: Evaluation on all rewards in multi-objective optimization where we optimize the sum of PickScore, HPSv2.1, and CLIPScore.

Multi-objective optimization. Now we experiment with optimizing multiple rewards by training on the sum of PickScore, HPSv2.1, and CLIPScore. We report the combined reward in Figure 2e and individual rewards in Figure 3. We observe two main trends. First, although we optimize the combined reward, all rewards in Figure 3 improve. This shows that simple variants of our method can be used for multi-objective optimization. Second, `AdvantageFlow` attains the highest reward of the best baseline `DiffusionNFT` in a half of its computational cost.

FLUX.1 experiments. To show that our gains are not limited to Stable Diffusion 3.5 Medium, we experiment with PickScore maximization in FLUX.1 (Black Forest Labs et al., 2025) in Figure 2f. We observe that `AdvantageFlow` outperforms `DiffusionNFT`. In addition, it is comparable to `AWM` initially and outperforms it later. These experiments are done in Flow-Factory (Ping et al., 2026) with default implementations of all baselines.

Rollout regularization strength. One technical contribution of our work is rollout regularization and setting it adaptively (Section 3.3). We ablate this in Figure 4 where `AdvantageFlow`(γ) corresponds to a fixed rollout regularization γ. When the regularization is low, `AdvantageFlow(0.2)` performs well in some problems (Figure 4a) but is unstable in others (Figure 4b). When the regularization is high, `AdvantageFlow(5)` is stable but conservative in both problems. In both problems, the adaptive regularization performs the best.

Qualitative evaluation and GenEval. To mitigate a potential concern of reward hacking, we show qualitative examples of image improvements in Figure 1 and Appendix A, for the learned policies in

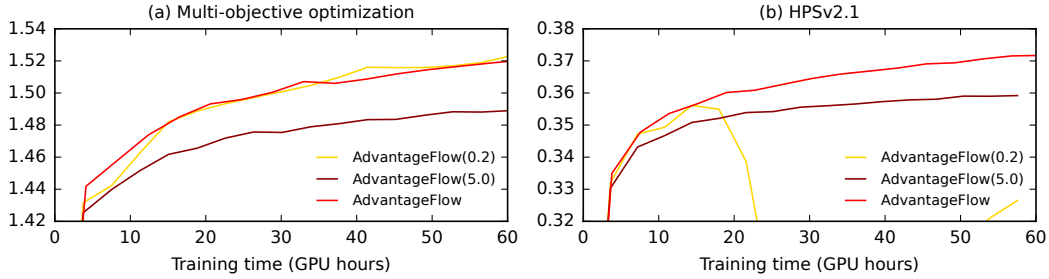

Figure 4: Comparison of AdvantageFlow with adaptive and fixed regularization on multi-objective optimization and HPSv2.1 problems.

Method	Single object	Two objects	Counting	Color	Position	Color binding	Overall
Without cfg							
Base	0.750	0.191	0.181	0.478	0.055	0.072	0.273
DiffusionNFT	**0.981**	0.810	0.581	**0.824**	0.197	0.557	0.647
AWM	0.953	0.792	0.571	**0.824**	0.192	0.555	0.637
AdvantageFlow	0.978	**0.830**	**0.621**	0.797	**0.257**	**0.615**	**0.673**
With cfg							
Base	0.981	0.772	0.587	0.805	0.190	0.525	0.631
DiffusionNFT	**0.987**	0.911	0.687	0.843	0.247	0.645	0.710
AWM	**0.987**	0.919	0.737	0.861	**0.292**	**0.685**	**0.737**
AdvantageFlow	0.984	**0.929**	**0.743**	**0.864**	**0.292**	0.655	0.734

Table 1: GenEval evaluation of AdvantageFlow, DiffusionNFT, AWM, and base model.

Figure 2e. We also evaluate the policies on GenEval (Ghosh et al., 2023). Our results are reported in Table 1 and we observe the following trends. First, AdvantageFlow improves over the original base model, with or without *classifier-free guidance (cfg)*. Second, AdvantageFlow without cfg mostly outperforms the base model with cfg. This shows that RL can learn high-quality policies. Finally, we note that AdvantageFlow generally performs better than the baselines without cfg and comparably to AWM with cfg.

Random seed. Due to the high computational cost of our experiments, all our experiments are with a single random seed 42. To show sensitivity to multiple random seeds, we repeat the experiment in Figure 2e with other seeds in Appendix A.

6 Related Work

Reward fine-tuning of diffusion and flow models has developed along three lines: trajectory-level RL on the reverse sampling process, reward backpropagation, and forward-process RL. Our method belongs to the third family. We briefly describe each and then discuss closest related methods.

Trajectory-level RL. DDPO (Black et al., 2024) and DPOK (Fan et al., 2023) treat denoising as a finite-horizon Markov decision process and apply PPO-style updates to per-step Gaussian transitions. Flow-GRPO (Liu et al., 2025a) brings this idea to flow models by converting the sampling ODE into an SDE with tractable per-step likelihoods; later variants improve credit assignment and rollout efficiency (Xue et al., 2025b; He et al., 2025; Li et al., 2025a;b). These methods optimize along the reverse trajectory, requiring credit assignment across many denoising steps. Forward-process RL avoids this by using the same prediction loss as pretraining.

Reward backpropagation. DRaFT (Clark et al., 2024), AlignProp (Prabhudesai et al., 2023), and Adjoint Matching (Domingo-Enrich et al., 2024) differentiate the reward through the sampler or an equivalent control objective. This is effective for differentiable rewards but does not directly apply to the black-box rule-based rewards, such as OCR, used in many text-to-image benchmarks.

Forward-process RL. The core idea (generate samples, score them, and retrain with the pretraining loss using reward-dependent weights) has appeared in several forms. Reward-weighted likelihood maximization (Lee et al., 2023; Peters & Schaal, 2007) and ORW-CFM-W2 (Fan et al., 2025b) weight the flow-matching loss by reward values. Centered reward distillation (Zhu et al., 2026) and AC-Flow (Fan et al., 2025a) also fit in this view.

`DiffusionNFT` (Zheng et al., 2026) fixes a rollout policy v_{old} and constructs implicit positive and negative velocities from a reward-derived optimality weight $r \in [0,1]$, $v_\theta^+ = (1-\beta)v_{\text{old}} + \beta v_\theta$ and $v_\theta^- = (1+\beta)v_{\text{old}} - \beta v_\theta$, fit to the flow-matching target v by the branch loss $r\left\|v_\theta^+ - v\right\|_2^2 + (1-r)\left\|v_\theta^- - v\right\|_2^2$. This direct regression construction is motivated by classifier-free guidance: the single trainable velocity v_θ is pulled toward the flow-matching target through the positive branch and pushed away from it through the negative branch. The algorithm can take multiple gradient steps on a batch of samples collected under a fixed v_{old} before that policy is updated by an exponential moving average. `AdvantageFlow` identifies an explicit reward-improved target distribution and fits the flow model to it directly (Section 3). In fact, after converting `DiffusionNFT` to prediction space, its loss can be viewed as in `AdvantageFlow` with an advantage-dependent rollout-regularization coefficient (Appendix D), under additional assumptions.

Advantage weighted matching (`AWM`) (Xue et al., 2025a) starts from the GRPO policy-gradient objective, treating the generated image x_0 as the action of a sequence-level policy $p_\theta(x_0 \mid c)$, and substitutes the flow-matching ELBO surrogate for its intractable log-likelihood. On-policy, this yields the advantage-weighted flow-matching gradient. Because the loss is derived as a policy gradient, reusing off-policy samples requires a correction, which AWM approximates by an importance ratio built from exponentiated differences of flow-matching losses. `AdvantageFlow` instead writes its loss directly as an expectation under the fixed rollout policy p_{old}, using the tilted target distribution in (9). So it is a stationary batch-optimization problem by construction and needs no importance correction for repeated optimization steps.

7 Conclusions

We introduce `AdvantageFlow`, a forward-process RL algorithm for rectified flow models based on advantage-weighted least squares. The algorithm fits a local additive reward-improving distribution, where the rollout regularization reduces variance and improves stability, by keeping the per-sample quadratic loss strictly convex. We evaluate `AdvantageFlow` on text-to-image generation, and show that it outperforms both `Flow-GRPO` and state-of-the-art forward-process RL algorithms.

Limitations and future work. Our analysis is local and at a population level. We assume a centered population advantage and that the rollout predictor f_{old} is equal to the conditional mean under p_{old}. In practice, finite prompt groups, empirical reward normalization, clipped advantages, LoRA, and finite optimization are used. Therefore, while our theory motivates our objective, it is not the exact description of the finite-sample training dynamics.

The performance of `AdvantageFlow`, similarly to any other RL algorithm, depends on the quality of reward models, which we do not investigate. Learned reward models may be misspecified. On the other hand, rule-based reward models capture only a narrow aspect of image quality. Finally, the rollout and reference regularization strengths are free hyper-parameters of `AdvantageFlow`, and a more complete theory for selecting them is needed.

AI USE STATEMENT

We used generative AI tools in the followig ways:

- **Assist in the writing of proofs.** The Fisher-Rao derivation in Appendix B was developed with assistance from LLMs.
- **Edit a research paper to improve readability.** We used LLMs to improve the exposition and wording in theoretical parts of the manuscript.
- **Create or edit software code.** `AWM` was ported to the DiffusionNFT code base by coding agents and then verified by authors.

ETHICS STATEMENT

We read the Code of Ethics and will adhere to it. We do not believe that our work raises any ethics concerns. We propose a new algorithm for an existing framework and evaluate it on existing datasets. Our work does not involve human subjects.

REPRODUCIBILITY STATEMENT

Sufficient details to reproduce our work are stated in Section 5. Specifically:

- **Stable Diffusion 3.5 Medium.** All experiments are conducted in a public code base and on benchmarks in it: `https://github.com/NVlabs/DiffusionNFT`. Our algorithm is implemented by replacing the DiffusionNFT loss in DiffusionNFT/config/nft.py with our loss. We will try to release the code publicly. We use default hyper-parameters unless stated in Section 5.
- **FLUX.1.** All experiments are conducted in a public code base and on benchmarks in it: `https://github.com/X-GenGroup/Flow-Factory`. Our algorithm is implemented by replacing the DiffusionNFT loss with our loss. We will try to release the code publicly. We use default hyper-parameters unless stated in Section 5.

REFERENCES

Michael S. Albergo, Nicholas M. Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023.

Shun-ichi Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, 1998.

Jonathan Baxter and Peter Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350, 2001.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *ICLR*, 2024.

Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, Kyle Lacey, Yam Levi, Cheng Li, Dominik Lorenz, Jonas Muller, Dustin Podell, Robin Rombach, Harry Saini, Axel Sauer, and Luke Smith. FLUX.1 Kontext: Flow matching for in-context image generation and editing in latent space. *CoRR*, abs/2506.15742, 2025. URL https://arxiv.org/abs/2506.15742.

George Casella and Christian Robert. Rao-Blackwellisation of sampling schemes. *Biometrika*, 83 (1):81–94, 1996.

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. *arXiv preprint arXiv:2409.08861*, 2024.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Muller, 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 *Proceedings of the 41th International Conference on Machine Learning*, 2024.

Jiajun Fan, Chaoran Cheng, Shuaike Shen, Xiangxin Zhou, and Ge Liu. Fine-tuning flow matching generative models with intermediate feedback. *arXiv preprint arXiv:2510.18072*, 2025a.

Jiajun Fan, Shuaike Shen, Chaoran Cheng, Yuxin Chen, Chumeng Liang, and Ge Liu. Online reward-weighted fine-tuning of flow matching with Wasserstein regularization. In *ICLR*, 2025b.

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

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. GenEval: An object-focused framework for evaluating text-to-image alignment. In *Advances in Neural Information Processing Systems 36*, 2023.

Xiaoxuan He, Siming Fu, Yuke Zhao, Wanli Li, Jian Yang, Dacheng Yin, Fengyun Rao, and Bo Zhang. TempFlow-GRPO: When timing matters for GRPO in flow models. *arXiv preprint arXiv:2508.04324*, 2025.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. CLIPScore: A reference-free evaluation metric for image captioning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021.

Edward 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 *Proceedings of the 10th International Conference on Learning Representations*, 2022.

Sham M. Kakade. A natural policy gradient. In *NeurIPS*, 2001.

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 *Advances in Neural Information Processing Systems 36*, 2023.

Kimin Lee, Hao Liu, Moonkyung Ryu, Olivia Watkins, Yuqing Du, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, and Shixiang Shane Gu. Aligning text-to-image models using human feedback. *arXiv preprint arXiv:2302.12192*, 2023.

Junzhe Li, Yutao Cui, Tao Huang, Yinping Ma, Chun Fan, Yiming Cheng, Miles Yang, Zhao Zhong, and Liefeng Bo. MixGRPO: Unlocking flow-based GRPO efficiency with mixed ODE-SDE. *arXiv preprint arXiv:2507.21802*, 2025a.

Yuming Li, Yikai Wang, Yuying Zhu, Zhongyu Zhao, Ming Lu, Qi She, and Shanghang Zhang. BranchGRPO: Stable and efficient GRPO with structured branching in diffusion models. *arXiv preprint arXiv:2509.06040*, 2025b.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt 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. *arXiv preprint arXiv:2505.05470*, 2025a.

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

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding R1-Zero-like training: A critical perspective. In *Conference on Language Modeling*, 2025b.

Remi Munos. Geometric variance reduction in Markov chains: Application to value function and gradient estimation. *Journal of Machine Learning Research*, 7:413–427, 2006.

Jan Peters and Stefan Schaal. Reinforcement learning by reward-weighted regression for operational space control. In *ICML*, 2007.

Bowen Ping, Chengyou Jia, Minnan Luo, Hangwei Qian, and Ivor Tsang. Flow-Factory: A unified framework for reinforcement learning in flow-matching models. *CoRR*, abs/2602.12529, 2026. URL https://arxiv.org/abs/2602.12529.

Mihir Prabhudesai, Anirudh Goyal, Deepak Pathak, and Katerina Fragkiadaki. Aligning text-to-image diffusion models with reward backpropagation. *arXiv preprint arXiv:2310.03739*, 2023.

Yash Savani, Branislav Kveton, Yuchen Liu, Yilin Wang, Jing Shi, Subhojyoti Mukherjee, Nikos Vlassis, and Krishna Kumar Singh. Stepwise credit assignment for GRPO on flow-matching models. *arXiv preprint arXiv:2603.28718*, 2026.

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 preprint arXiv:2402.03300*, 2024.

Richard Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 12*, pp. 1057–1063, 2000.

Ronald Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 1992.

Xiaoshi Wu, Keqiang Sun, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score: Better aligning text-to-image models with human preference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

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

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 preprint arXiv:2505.07818*, 2025b.

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. arXiv:2509.16117.

Yujie Zhou, Pengyang Ling, Jiazi Bu, Yibin Wang, Yuhang Zang, Jiaqi Wang, Li Niu, and Guangtao Zhai. Fine-grained GRPO for precise preference alignment in flow models. *arXiv preprint arXiv:2510.01982*, 2025.

Yuanzhi Zhu, Xi Wang, Stéphane Lathuilière, and Vicky Kalogeiton. Diffusion reinforcement learning via centered reward distillation. *arXiv preprint arXiv:2603.14128*, 2026.



Figure 5: Images generated by `AdvantageFlow`, `DiffusionNFT`, and base model (Stable Diffusion 3.5 Medium) with cfg. We show improvements in object generation.

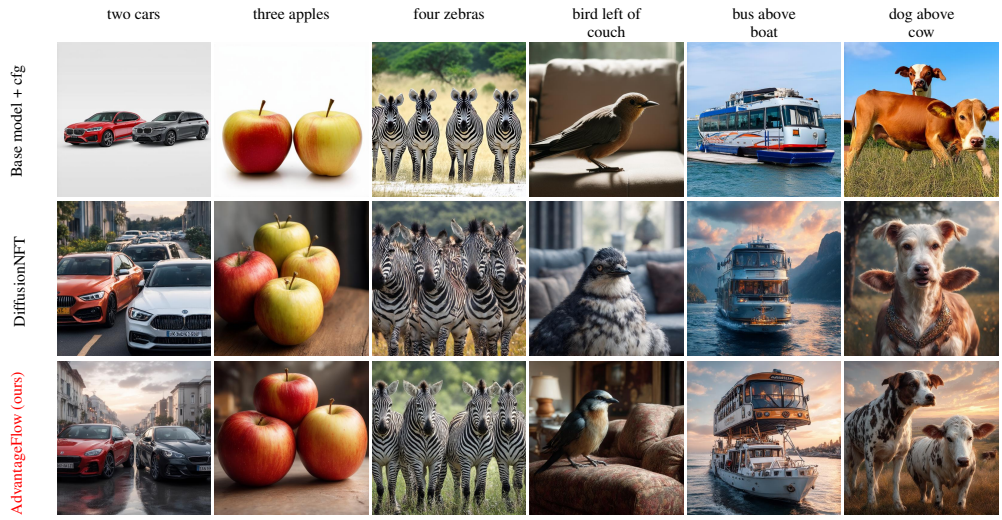

Figure 6: Images generated by `AdvantageFlow`, `DiffusionNFT`, and base model (Stable Diffusion 3.5 Medium) with cfg. We show improvements in counting and position understanding.

A Additional Experiments

We show qualitative examples of image improvements for the learned policies in Figure 2e next. Figure 5 shows improvements in object generation, Figure 6 shows improvements in counting and position understanding, and Figure 7 shows improvements in color understanding.

Due to the high computational cost of our experiments, and similarly to Zheng et al. (2026) and Xue et al. (2025a), all our experiments are with a single random seed 42. To show sensitivity to multiple random seeds, we repeat the experiment in Figure 2e with seeds 43 and 44. Our results are reported in Figure 8. While the plots change with the seed, the overall conclusion does not: `AdvantageFlow` outperforms `DiffusionNFT`, which in turn outperforms `AWM`.

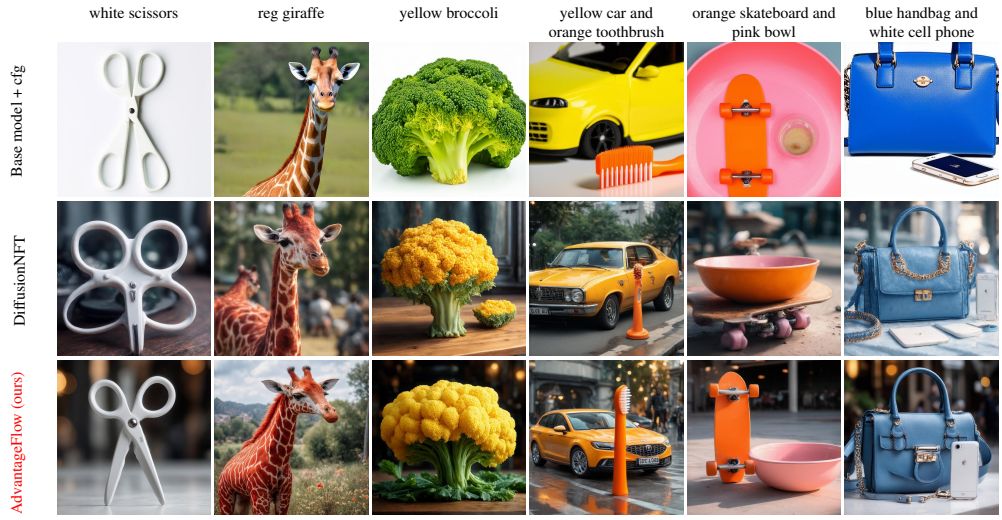

Figure 7: Images generated by `AdvantageFlow`, `DiffusionNFT`, and base model (Stable Diffusion 3.5 Medium) with cfg. We show improvements in color understanding.

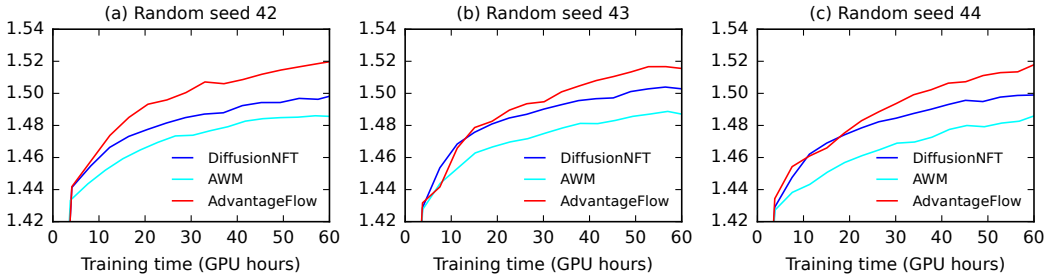

Figure 8: Evaluation on multi-objective optimization with random seeds 42, 43, and 44.

B Fisher–Rao Derivation

We prove Proposition 1, restated below for convenience.

Proposition 1 (restated). Fix a prompt c and let $F(p) = \mathbb{E}_p[r(\cdot, c)]$. Under the Fisher–Rao metric, the natural-gradient direction of F at $p = p_{\text{old}}(\cdot \mid c)$ is

$$\delta p(x_0) = A(x_0, c)\, p_{\text{old}}(x_0 \mid c), \qquad A(x_0, c) = r(x_0, c) - \mathbb{E}_{p_{\text{old}}(\cdot|c)}[r(\cdot, c)].$$

A first-order step of size η yields the additive tilt $q_\eta(\cdot \mid c) = (1 + \eta A(\cdot, c))\, p_{\text{old}}(\cdot \mid c)$ from Equation (9).

Proof. Let

$$F(p) = \mathbb{E}_p[r]. \tag{14}$$

For a perturbation δp of a density p, the probability constraint requires

$$\int \delta p(x)\ \mathrm{d}x = 0. \tag{15}$$

The first variation of F is

$$\delta F = \int r(x)\, \delta p(x)\ \mathrm{d}x. \tag{16}$$

Under the Fisher–Rao metric, tangent vectors are written as $\delta p(x) = u(x)p(x)$ with $\mathbb{E}_p[u] = 0$, and the inner product is

$$\langle u, v \rangle_p = \mathbb{E}_p[u(x)v(x)]. \tag{17}$$

The gradient direction u^* must satisfy

$$\mathbb{E}_p[u^*(x)u(x)] = \mathbb{E}_p[r(x)u(x)] \tag{18}$$

for every zero-mean tangent function u. Therefore

$$u^*(x) = r(x) - \mathbb{E}_p[r]. \tag{19}$$

At $p = p_{\text{old}}$, this gives the Fisher–Rao natural-gradient direction

$$\delta p(x) = A(x)\, p_{\text{old}}(x), \qquad A(x) = r(x) - \mathbb{E}_{p_{\text{old}}}[r]. \tag{20}$$

A first-order step of size η gives

$$q_\eta(x) = p_{\text{old}}(x) + \eta A(x) p_{\text{old}}(x) = \big(1 + \eta A(x)\big) p_{\text{old}}(x). \tag{21}$$

□

C Two-Anchor Equivalence and Variance Reduction

We prove Proposition 2, restated below for convenience.

Proposition 2 (restated). Fix a prompt c and suppose $f_{\text{old}}(x_t, t, c) = \mathbb{E}_{x_0 \sim p_{\text{old}}(\cdot|c),\, \epsilon}[x_0 \mid x_t, t, c]$. Up to a θ-independent constant,

$$\mathbb{E}_{x_0 \sim p_{\text{old}},\, \epsilon,\, t}\big[\|x_0 - f_\theta(x_t, t, c)\|_2^2\big] = \mathbb{E}_{x_0 \sim p_{\text{old}},\, \epsilon,\, t}\big[\|f_{\text{old}}(x_t, t, c) - f_\theta(x_t, t, c)\|_2^2\big].$$

Moreover, the single-sample gradient of the right-hand side has no larger variance than that of the left-hand side.

Proof. Fix a prompt c and suppress it for brevity. Fitting the additive target gives

$$\mathcal{L}_{q_\eta}(\theta) = \mathbb{E}_{p_{\text{old}},\, \epsilon,\, t}\big[(1 + \eta A(x_0))\ \|x_0 - f_\theta(x_t, t)\|_2^2\big]. \tag{22}$$

Expanding,

$$\begin{aligned}\mathcal{L}_{q_\eta}(\theta) = \mathbb{E}_{p_{\text{old}},\, \epsilon,\, t}\big[\|x_0 - f_\theta(x_t, t)\|_2^2\big] \\ + \eta\, \mathbb{E}_{p_{\text{old}},\, \epsilon,\, t}\big[A(x_0)\ \|x_0 - f_\theta(x_t, t)\|_2^2\big].\end{aligned} \tag{23}$$

Assume

$$f_{\text{old}}(x_t, t) = \mathbb{E}_{p_{\text{old}}}[x_0 \mid x_t, t]. \tag{24}$$

Then

$$x_0 - f_\theta = (x_0 - f_{\text{old}}) + (f_{\text{old}} - f_\theta). \tag{25}$$

Taking squared norms and expectations,

$$\begin{aligned}\mathbb{E}\,\|x_0 - f_\theta\|_2^2 = \mathbb{E}\,\|x_0 - f_{\text{old}}\|_2^2 + \mathbb{E}\,\|f_{\text{old}} - f_\theta\|_2^2 \\ + 2\,\mathbb{E}\,\langle x_0 - f_{\text{old}}, f_{\text{old}} - f_\theta\rangle\,.\end{aligned} \tag{26}$$

The cross term vanishes because

$$\mathbb{E}[x_0 - f_{\text{old}} \mid x_t, t] = 0. \tag{27}$$

Therefore

$$\mathbb{E}_{p_{\text{old}},\, \epsilon,\, t}\big[\|x_0 - f_\theta(x_t, t)\|_2^2\big] = \mathbb{E}_{p_{\text{old}},\, \epsilon,\, t}\big[\|f_{\text{old}}(x_t, t) - f_\theta(x_t, t)\|_2^2\big] + C, \tag{28}$$

where C is independent of θ. Substituting and rescaling by $1/\eta$ gives

$$\mathcal{L}_{\text{AFM}}(\theta) = \mathbb{E}_{p_{\text{old}},\, \epsilon,\, t}\left[A(x_0)\ \|x_0 - f_\theta(x_t, t)\|_2^2 + \frac{1}{\eta}\ \|f_\theta(x_t, t) - f_{\text{old}}(x_t, t)\|_2^2\right], \tag{29}$$

up to a θ-independent constant.

Variance reduction. Let

$$g_{\text{sample}} = \nabla_\theta \, \|x_0 - f_\theta(x_t, t)\|_2^2, \qquad g_{\text{rollout}} = \nabla_\theta \, \|f_{\text{old}}(x_t, t) - f_\theta(x_t, t)\|_2^2$$

be the single-sample gradients of the two terms. Since $f_{\text{old}}(x_t, t) = \mathbb{E}[x_0 \mid x_t, t]$,

$$g_{\text{rollout}} = \mathbb{E}[g_{\text{sample}} \mid x_t, t]. \tag{30}$$

The Rao–Blackwell inequality gives

$$\text{Var}(g_{\text{rollout}}) \le \text{Var}(g_{\text{sample}}). \tag{31}$$

□

D DiffusionNFT Algebra

We prove the DiffusionNFT connection stated in Section 6.

Proposition 3 (DiffusionNFT as a special case of AdvantageFlow)**.** *Let $r(x_0, c) \in [0, 1]$ be a normalized optimality score and $A(x_0, c) = 2r(x_0, c) - 1$. The DiffusionNFT branch loss, after replacing velocity losses with prediction losses and dropping θ-independent constants, equals*

$$\beta A(x_0, c) \, \|x_0 - f_\theta(x_t, t, c)\|_2^2 + \beta(\beta - A(x_0, c)) \, \|f_\theta(x_t, t, c) - f_{\text{old}}(x_t, t, c)\|_2^2,$$

corresponding to the AdvantageFlow objective (5) with $\lambda = 0$ and rollout regularization schedule $\gamma_{\text{NFT}}(A) = \beta(\beta - A(x_0, c))$.

Proof. We work in velocity space, since that is how DiffusionNFT states its objective.

Let v be the forward-process velocity target, let v_{old} be the rollout model, and let v_θ be the trainable model. DiffusionNFT defines positive and negative branches

$$v_\theta^+ = (1 - \beta)v_{\text{old}} + \beta v_\theta, \qquad v_\theta^- = (1 + \beta)v_{\text{old}} - \beta v_\theta. \tag{32}$$

Let $r \in [0, 1]$ be the normalized optimality score and define $A(x_0, c) = 2r(x_0, c) - 1$. The branch loss is

$$\ell_{\text{NFT}} = r \, \left\|v_\theta^+ - v\right\|_2^2 + (1 - r) \, \left\|v_\theta^- - v\right\|_2^2. \tag{33}$$

Set $d = v_\theta - v_{\text{old}}$ and $e = v_{\text{old}} - v$. Then

$$v_\theta^+ - v = e + \beta d, \qquad v_\theta^- - v = e - \beta d. \tag{34}$$

Expanding,

$$\ell_{\text{NFT}} = \|e\|_2^2 + \beta^2 \, \|d\|_2^2 + 2\beta A(x_0, c) \, \langle e, d \rangle. \tag{35}$$

Now consider the two-anchor expression

$$A(x_0, c) \, \|v_\theta - v\|_2^2 + (\beta - A(x_0, c)) \, \|v_\theta - v_{\text{old}}\|_2^2. \tag{36}$$

Since $v_\theta - v = e + d$, this equals

$$A(x_0, c) \, \|e\|_2^2 + \beta \, \|d\|_2^2 + 2A(x_0, c) \, \langle e, d \rangle. \tag{37}$$

Multiplying by β gives

$$\beta A(x_0, c) \, \|e\|_2^2 + \beta^2 \, \|d\|_2^2 + 2\beta A(x_0, c) \, \langle e, d \rangle. \tag{38}$$

This differs from Equation (35) only by $(1 - \beta A(x_0, c)) \, \|e\|_2^2$, which is independent of v_θ. Therefore, after dropping constants,

$$\ell_{\text{NFT}} \equiv \beta A(x_0, c) \, \|v_\theta - v\|_2^2 + \beta(\beta - A(x_0, c)) \, \|v_\theta - v_{\text{old}}\|_2^2. \tag{39}$$

To pass from velocity space to prediction space, use

$$f_\theta(x_t, t, c) = x_t - t v_\theta(x_t, t, c), \qquad f_{\text{old}}(x_t, t, c) = x_t - t v_{\text{old}}(x_t, t, c),$$

and let $v = \epsilon - x_0$ be the forward-process velocity target. Then

$$x_0 - f_\theta(x_t, t, c) = t(v_\theta - v), \qquad f_\theta(x_t, t, c) - f_{\text{old}}(x_t, t, c) = -t(v_\theta - v_{\text{old}}).$$

Thus, after moving to the prediction-loss by multiplying the velocity-space objective by t^2, we have

$$t^2 \, \|v_\theta - v\|_2^2 = \|x_0 - f_\theta(x_t, t, c)\|_2^2, \qquad t^2 \, \|v_\theta - v_{\text{old}}\|_2^2 = \|f_\theta(x_t, t, c) - f_{\text{old}}(x_t, t, c)\|_2^2.$$

Substituting these identities into the velocity-space expression gives the prediction-space form of Equation (39). Thus DiffusionNFT corresponds to the advantage-dependent schedule

$$\gamma_{\text{NFT}}(A) = \beta(\beta - A(x_0, c)). \tag{40}$$

For $\beta = 1$, this becomes $\gamma_{\text{NFT}}(A) = 1 - A(x_0, c)$. □