MINTFLOW: MINIMAL TRAJECTORY INTERVENTION FOR CONSTRAINED FLOW MATCHING

Yesom Park[1], Kelvin Kan[1], Qifan Chen[1], Thomas Flynn[2], Hayden Schaeffer[1], Xihaier Luo[2]*
[1]Department of Mathematics, University of California, Los Angeles
[2]Brookhaven National Laboratory
{yeisom, kelvin.kan, chenqifan, hayden}@math.ucla.edu
{tflynn, xluo}@bnl.gov

ABSTRACT

Flow matching models excel at generative modeling, and many downstream applications require their samples to satisfy prescribed constraints, such as observed measurements and physical laws. However, existing constrained samplers often face a trade-off: *enforcing constraints can substantially displace samples from the pretrained data distribution*. To address this trade-off, we introduce **MintFlow**, a training-free constrained sampling framework that formulates constraint enforcement as a minimal intervention on the pretrained flow trajectory. MintFlow seeks the minimal perturbation of an intermediate flow state such that its subsequent evolution under the pretrained flow field satisfies the target constraint. By minimally perturbing the flow state while keeping the pretrained flow field unchanged, MintFlow enforces the constraint while minimizing unnecessary deviation from the pretrained distribution. An adjoint formulation yields a closed-form expression for this perturbation, eliminating expensive iterative optimization. Furthermore, MintFlow adaptively selects the intervention time to balance the required perturbation magnitude with its amplification by the remaining flow. Across a range of tasks in generative vision and physical system modeling, MintFlow achieves competitive constraint satisfaction while preserving the pretrained generative distribution substantially better than state-of-the-art constrained methods.

1 INTRODUCTION

Flow matching has emerged as a powerful paradigm for generative modeling, efficiently transporting simple reference distributions into complex data distributions through learned continuous-time dynamics (Lipman et al., 2023; Liu et al., 2026). Yet many real-world applications demand more than plausible generation. Generated samples need to satisfy measurement constraints, structural requirements, physical laws, or other application-specific conditions (Utkarsh et al., 2026; Martin et al., 2025a; Patel et al., 2025). In these settings, the central objective is to enforce specified constraints while preserving the distributional structure already learned by the pretrained model.

Existing methods approach this problem either during training or at inference time. Training-time methods (Li et al., 2026a; Tauberschmidt et al., 2026) incorporate constraints through specialized objectives, regularization, or architectural design, but require costly retraining for each new constraint, which is often impractical. Inference-time methods avoid this by enforcing the constraint during sampling, either by modifying the pretrained dynamics via guidance (Chung et al., 2023a; Feng et al., 2025), flow states through trajectory optimization (Ben-Hamu et al., 2024; Patel et al., 2025), or generated samples using post-hoc projection (Cheng et al., 2025; Utkarsh et al., 2026).[1] However, these approaches face a key challenge: *constraint satisfaction alone does not prevent large deviations to the pretrained generation process, which can push samples away from the pretrained distribution* (Figure 1).

Therefore, we adopt a simple principle: *modify only what is necessary, and only as much as necessary*. As the desired constrained samples remain within the distribution learned by the pretrained

*Corresponding author.
[1]Refer to Appendix C for a detailed discussion of related work.

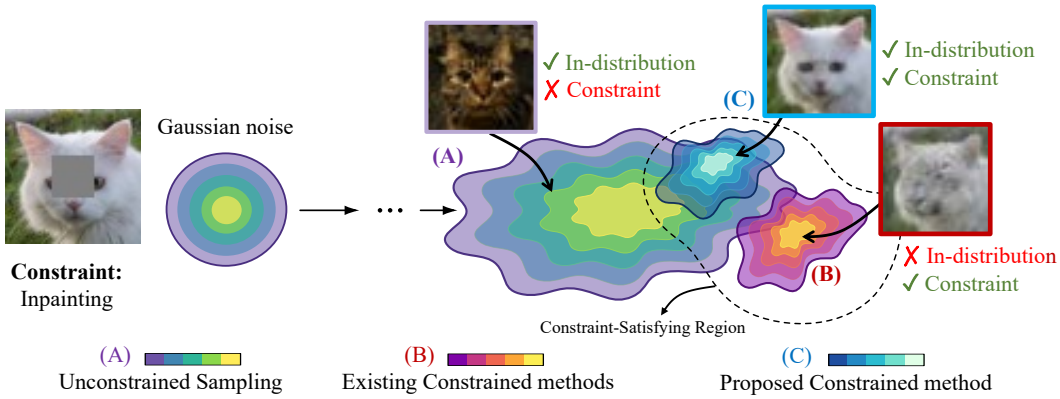


Figure 1: Balancing constraint satisfaction and distributional fidelity. **Unconstrained sampling (A)** generates samples from the pretrained distribution but often violates the constraints, while **prior constrained sampling methods (B)** enforce the constraints at the cost of large distributional shifts. **The proposed method (C)** achieves constraint satisfaction while better preserving the pretrained distribution. Image inpainting examples illustrate this trade-off.

flow, the learned dynamics themselves need not be altered. This suggests preserving the original vector field and intervening only on the trajectory with the smallest correction needed to satisfy the constraint while limiting unnecessary deviation from the pretrained distribution.

Motivated by this principle, we propose **MintFlow**, a training-free framework that formulates constrained generation as *minimal trajectory intervention* (Figure 2). Given a trajectory generated by the pretrained flow, MintFlow finds the smallest perturbation to an intermediate state such that its subsequent evolution under the *unchanged pretrained flow* satisfies the terminal constraint. The minimum-norm intervention is computed directly from the terminal constraint sensitivity via an adjoint equation, avoiding explicit construction of the high-dimensional flow Jacobian and costly iterative optimization through the full generation trajectory. MintFlow further adaptively selects the intervention time to balance the correction magnitude against the remaining flow propagation time.

Under mild regularity conditions, we characterize the constraint satisfaction error and show that the Wasserstein distance between the corrected and pretrained distributions is bounded by the intervention magnitude and the remaining flow propagation time. Experiments on image and physics data with diverse constraints, including image inverse problems, image editing, and physics-informed generation, demonstrate that MintFlow effectively satisfies constraints while preserving the learned distribution across a broad range of applications.

2 Background

Let f_θ be the velocity field of a pretrained flow matching model, which generates samples by integrating the ordinary differential equation (ODE)

$$\frac{du}{dt} = f_\theta(u,t), \qquad t \in [0,T]. \tag{1}$$

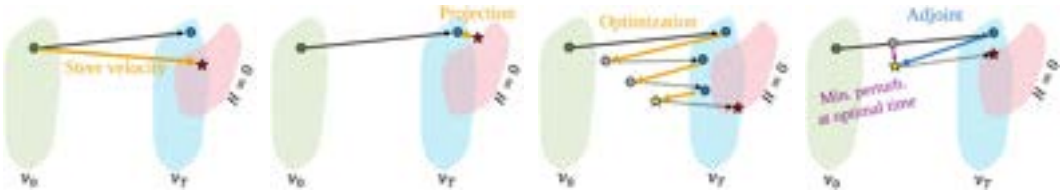
We denote by $\Phi(t,s)$ the associated flow map that moves a state from time s to t along (1), i.e., $\Phi(t,s)(u_s) = u_t$. For a base distribution μ at $t=0$, the induced marginal distribution at time t is

$$\nu_t := \Phi(t,0)_\# \mu,$$

where $(\cdot)_\#$ denotes the pushforward of a distribution through the flow. The terminal distribution $\nu_T = \Phi(T,0)_\# \mu$ is assumed to approximate the target data distribution.

Constrained Generation. Given a pretrained flow matching model with terminal distribution ν_T and a differentiable constraint operator H, we aim to generate samples u satisfying

$$u \sim \nu_T, \qquad H(u) = 0. \tag{2}$$



(a) Steer Velocity Field (b) Terminal Projection (c) Trajectory Optimization (d) Trajectory Intervention (Ours)

Figure 2: Comparison of constrained generation strategies. **Steer Velocity Field** modifies the flow by learning an additional velocity field. **Terminal Projection** projects the generated sample onto the constraint manifold. **Trajectory Optimization** iteratively optimizes states along the flow. In contrast, our **Trajectory Intervention** minimally perturbs the trajectory at an intermediate time then follows the original flow.

As the pretrained velocity field f_θ is learned to transport the base distribution μ toward the unconstrained terminal distribution ν_T, it is not explicitly trained to enforce the constraint $H(u) = 0$. As such, we seek a constrained generation procedure that generates samples from the terminal distribution ν_T that satisfy the prescribed constraint using only the pretrained flow field f_θ and the constraint operator H without additional model training.

The key challenge is to simultaneously satisfy the constraint $H(u) = 0$ while keeping the samples close to the pretrained distribution ν_T. Existing approaches often enforce the constraint at the cost of deviating from ν_T (see Figure 1). To address this, we formulate constrained generation as a *minimal-intervention problem*, seeking the smallest perturbation to the intermediate states of the pretrained flow trajectory necessary to enforce the constraint while maintaining the pretrained distribution.

3 Method

3.1 Trajectory Intervention for Constrained Generation

Rather than applying post-hoc projection or modifying the pretrained flow, we enforce the constraint through a minimal intervention on the generated trajectory. Given a trajectory $\{u_t\}_{t\in[0,T]}$, we seek an intermediate time $s \in [0, T)$ and a perturbation δ such that

$$H\left(\Phi(T, s)(u_s + \delta)\right) = 0.$$

Thus, the perturbed state $u_s + \delta$ is propagated to the terminal time using the original pretrained flow. For a fixed intervention time s, we find the smallest perturbation satisfying the terminal constraint:

$$\min_{\delta} \ \frac{1}{2}\left\|\delta\right\|^2, \qquad \text{subject to } \ F_s(\delta) \coloneqq H\left(\Phi(T, s)(u_s + \delta)\right) = 0. \tag{3}$$

This formulation casts constrained generation as a **minimal trajectory intervention**. The flow dynamics remain unchanged, while only the intermediate state is minimally perturbed to reach the constraint manifold at $t = T$.

3.2 Adjoint-Based Optimal Correction

We solve (3) by computing the minimum-norm perturbation via adjoint sensitivity of the flow map. For a generated trajectory $\{u_t\}_{t\in[0,T]}$, consider a perturbation at an intermediate time s:

$$u'_s = u_s + \delta.$$

For a sufficiently small δ, the terminal constraint can be linearized around the original trajectory as

$$H(u') = F_s(\delta) \approx H(u_T) + \nabla H(u_T) D\Phi(T, s)(u_s)\delta, \quad D\Phi(T, s) \coloneqq \frac{\partial \Phi(T, s)(u)}{\partial u}(u). \tag{4}$$

Then, enforcing $F_s(\delta) = 0$ under the linear approximation gives

$$A_s\delta = -H(u_T), \quad A_s \coloneqq \nabla H(u_T) D\Phi(T, s)(u_s) \tag{5}$$

whose minimum-norm solution is

$$\delta_s^* = -A_s^\dagger H(u_T), \tag{6}$$

where $A_s^\dagger$ denotes the Moore–Penrose pseudoinverse, which can be computed by solving the associated least-squares problem. In particular, when H is scalar-valued, (6) reduces to

$$\delta_s^* = -\frac{A_s^\top}{\|A_s\|^2} H(u_T), \quad A_s \neq 0.$$

Consequently, for each intervention time s, the minimum-norm perturbation is determined under the linear approximation. This correction can also be interpreted as a single *Newton step* for solving the nonlinear equation $F_s(\delta) = 0$. Appendix A.1 also provides an *optimal control* interpretation.

Adjoint Computation. The computation of A_s in in (5) requires the sensitivity of the terminal state with respect to the intermediate state. Directly computing the full Jacobian $D\Phi(T, s)$ is computationally prohibitive for high-dimensional systems. Instead, we compute its vector-Jacobian product through the adjoint equation.

Specifically, we define the adjoint variable $\lambda(t)$ as the solution of the *adjoint equation*

$$\dot{\lambda}(t) = -\left(\frac{\partial f_\theta}{\partial u}(u_t, t)\right)^\top \lambda(t), \quad \lambda(T) = \nabla H(u_T)^\top. \tag{7}$$

Solving this equation backward from $t = T$ to $t = s$ yields exactly the required vector-Jacobian product (Coddington, 1955; Pontryagin, 1987)

$$\lambda(s) = D\Phi(T, s)^\top \nabla H(u_T)^\top = A_s^\top.$$

Thus, A_s is obtained without explicitly forming the flow Jacobian, which enables efficient computation of the optimal correction for high-dimensional flow matching models.

We provide theoretical guarantees for the proposed adjoint-based intermediate correction in terms of both constraint satisfaction and distributional preservation.

Theorem 3.1 (Informal)**.** *Under mild regularity conditions on f_θ and H, the proposed linearized correction in (6) achieves a second-order terminal constraint residual:*

$$\|H\left(\Phi(T, s)(u_s + \delta_s^*)\right)\| \leq C\|\delta_s^*\|^2.$$

Moreover, if the flow field is L_f-Lipschitz in u, the corrected terminal distribution ν_T^* satisfies

$$W_2(\nu_T^*, \nu_T) \leq e^{L_f(T-s)}\|\delta_s^*\|_{L^2(\nu_s)}.$$

The formal statements and proofs are provided in Appendix A.2. The first result quantifies the constraint satisfaction of the proposed correction, while the second shows the corrected terminal distribution deviation from the original flow-matching distribution is bounded by both the magnitude of the intermediate correction and the remaining integration time $T - s$.

3.3 Adaptive Intervention Time Selection

The previous subsection determines the minimum-norm correction for a fixed intervention time s. We next select s by balancing the correction magnitude and the remaining propagation time $T - s$. Theorem 3.1 suggests that two factors should be controlled: the magnitude of the intervention and the remaining flow integration time. Because the Lipschitz constant L_f is generally unknown, we instead consider the following practical surrogate:

$$s^* = \arg\min_{s \in [0,T)} J(s) := \|\delta_s^*\|^2 + \lambda(T - s)^2, \tag{8}$$

where δ_s^* is given by (6) and λ controls the preference for later intervention. The first term favors intervention times that require a small correction. The second term favors later intervention by penalizing the remaining propagation time after the intervention. This reduces both the potential amplification of the perturbation through the remaining flow evolution and the numerical cost required to reach the terminal time. Thus, $J(s)$ provides a practical criterion that balances distributional preservation and computational efficiency.

In practice, we evaluate $J(s)$ on a finite set of candidate times $0 \leq s_1 < \cdots < s_K < T$:

$$s^* = \arg\min_{k=1,\cdots,K} J(s_k). \quad (9)$$

After generating the trajectory once, we integrate the adjoint equation backward to the earliest candidate time s_1, storing adjoint states at $s_k k = 1^K$. We then compute $\delta s_k{}^*$ and $J(s_k)$ in parallel for all k and select s^* via (9). We then apply

Algorithm 1 MintFlow Constrained Generation

Require: $u_0 \sim \mu$, f_θ, H, $\{s_k\}_{k=1}^K$, λ

1: Generate trajectory $\{u_t\}_{t=0}^T$ and compute $H(u_T)$.
2: Solve the adjoint backward from T and obtain sensitivities at $\{s_k\}_{k=1}^K$.
3: In parallel, compute $\delta^*_{s_k}$ via (6) and evaluate $J(s_k)$ for all s_k.
4: Select $s^* = \arg\min_{s_k} J(s_k)$.
5: Re-integrate from $u_{s^*} + \delta^*_{s^*}$ to T.
return $u_T^{\text{corr}} = \Phi(T, s^*)\left(u_{s^*} + \delta^*_{s^*}\right)$

$$u_{s^*} \leftarrow u_{s^*} + \delta^*_{s^*}$$

and reintegrate the original flow from s^* to T to obtain the constrained sample. The complete MintFlow algorithm is summarized in Algorithm 1.

Iterative Refinement. The correction in (6) is a single Newton step for $F_s(\delta) = 0$. If the resulting constraint residual exceeds a prescribed tolerance, we iteratively apply Algorithm 1 on the corrected trajectory until the residual falls below the tolerance. At iteration i, we compute the adjoint sensitivity, select an intervention time $s^{(i)}$, apply $\delta^*_{s^{(i)}}$, and reintegrate the original flow to T. We restrict the intervention times to $s^{(1)} < s^{(2)} < \cdots < T$, so each subsequent iteration starts from the trajectory produced by the previous correction and requires a shorter backward adjoint integration. Algorithm 3 and its convergence analysis are provided in Appendix A.3.

4 Experiments

We evaluate **MintFlow** on three constrained generation settings: *(i)* image inverse problems, *(ii)* image editing, and *(iii)* physical system modeling. The experiments span different flow models, data domains, and constraint types, ranging from local to global and from linear to nonlinear, demonstrating the broad applicability of our approach. We compare MintFlow against existing training-free constrained generation methods, using the same pretrained flow for all methods.[2] We assess each method along three core dimensions, *constraint satisfaction*, *preservation of the pretrained distribution*, and *computational efficiency*, supplemented by task-specific performance metrics. The code is publicly available at `https://github.com/Yebbi/MintFlow`.

4.1 Image Inverse Problems

Setup. Following Patel et al. (2025), we evaluate MintFlow on noise-free image inverse problems, including inpainting, super resolution, and deblurring, using the AFHQ-Cat and FFHQ datasets. We use the same pretrained Rectified Flow++ (RF) models (Lee et al., 2024) across all experiments, and resize all images to 64×64. For each task, given a degraded observation u^{obs}, we define the measurement constraint as $H(u) = A(u) - u^{\text{obs}} = 0$, where A denotes the task-specific degradation operator. We compare MintFlow with existing image-constraint methods—including D-Flow, DPS, Freedom, PNPFlow, and FlowChef—while sharing the same frozen RF model and source noises. In addition to constraint satisfaction, distribution preservation, and efficiency, we report reconstruction quality using PSNR, SSIM, and LPIPS. For further details, please refer to Appendix E.

Quantitative Results. Table 1 summarizes the quantitative results. Across all tasks and datasets, MintFlow consistently provides a substantially better balance between constraint satisfaction and reconstruction quality than existing methods. In particular, it achieves the lowest constraint error and Fréchet Inception Distance (FID) score across all evaluated tasks. This indicates that MintFlow can closely satisfy the measurement constraints without requiring excessive modification of the pretrained generative prior. Importantly, the improved constraint satisfaction does not come at the expense of perceptual or reconstruction quality. MintFlow achieves the best FID, PSNR, SSIM, and LPIPS across the evaluated settings, demonstrating its ability to simultaneously preserve the

[2] All image experiments were conducted on a single NVIDIA H200 GPU, and all physical system experiments were conducted on a single NVIDIA A100 GPU.

Table 1: Quantitative comparison of methods for image inverse problems on AFHQ-Cat and FFHQ. We report constraint error H, FID, PSNR (dB), SSIM, LPIPS, inference time (s), and peak GPU memory (MB). Light green (green) indicates the top performer, while light red (red) represents the second-best performer across all methods for each metric (applies to all tables herein).

	AFHQ-Cat					FFHQ					Efficiency	
Method	$H \downarrow$	**FID** $\downarrow$	**PSNR** $\uparrow$	**SSIM** $\uparrow$	**LPIPS** $\downarrow$	$H \downarrow$	**FID** $\downarrow$	**PSNR** $\uparrow$	**SSIM** $\uparrow$	**LPIPS** $\downarrow$	**Time** $\downarrow$	**Memory** $\downarrow$
Task: Inpainting												
D-Flow	1.775E-2	23.662	20.8597	0.7180	0.1578	1.592E-2	29.692	22.7378	0.7910	0.0915	153.47	3597.02
DPS	2.583E-2	14.532	20.9537	0.6232	0.1806	2.897E-2	33.498	21.3944	0.6913	0.1146	6.61	646.74
Freedom	6.013E-2	19.999	17.8908	0.4377	0.2502	6.789E-2	39.998	18.0828	0.5334	0.1605	4.60	637.36
PNPFlow	1.926E-2	24.063	22.2547	0.6877	0.1897	2.126E-2	63.579	22.3143	0.7388	0.1281	5.56	338.28
FlowChef	2.476E-2	18.292	19.4181	0.6110	0.1824	2.572E-2	17.226	20.5481	0.6934	0.1028	0.21	330.74
MintFlow	5.498E-4	6.506	26.3913	0.9207	0.0376	5.172E-4	6.046	28.7002	0.9417	0.0207	0.19	329.99
Task: Super Resolution												
D-Flow	1.371E-2	27.351	19.9141	0.5397	0.2309	1.347E-2	65.788	20.2276	0.6187	0.1565	188.96	3596.63
DPS	1.723E-2	12.771	19.9188	0.5220	0.1686	1.838E-2	9.608	19.7618	0.5924	0.1173	6.42	646.69
Freedom	4.814E-2	19.048	17.5691	0.3816	0.2452	5.124E-2	25.978	17.3415	0.4694	0.1568	4.53	637.32
PNPFlow	1.193E-2	52.9554	21.6583	0.6187	0.2289	1.342E-2	79.951	21.1512	0.6670	0.1649	4.9	338.24
FlowChef	1.842E-2	23.601	18.9815	0.4435	0.2352	1.950E-2	18.998	18.7612	0.5141	0.1381	0.25	330.69
MintFlow	9.901E-5	8.296	22.9958	0.7147	0.0757	1.248E-4	8.313	22.4535	0.7468	0.0543	0.22	329.86
Task: Deblurring												
D-Flow	9.015E-3	47.8724	18.2846	0.3734	0.3025	9.850E-3	81.818	17.9925	0.4277	0.1528	187.30	3596.97
DPS	1.950E-2	12.5016	20.5182	0.5629	0.1694	2.018E-2	23.919	20.2454	0.6272	0.1145	6.23	646.69
Freedom	5.174E-2	18.962	17.6911	0.3983	0.2384	5.191E-2	28.108	17.5191	0.4886	0.1566	4.11	637.32
PNPFlow	1.211E-2	25.170	22.3216	0.6711	0.2005	1.348E-2	73.589	21.8534	0.7116	0.1463	4.43	338.24
FlowChef	2.365E-2	18.548	19.2998	0.4924	0.2237	2.543E-2	19.195	19.0493	0.5593	0.1334	0.23	330.69
MintFlow	2.877E-4	6.819	28.3760	0.8988	0.03445	2.916E-4	6.417	28.4017	0.9221	0.0237	0.22	329.94

learned data distribution and recover high-quality image details. Furthermore, MintFlow shows computational efficiency of a minimal-intervention approach. In contrast, while competing methods can reduce the constraint error, they often produce reconstructions that deviate from the learned data distribution, resulting in higher FID and inferior reconstruction quality (Figure 3).

Qualitative Comparison. Figure 3 presents representative qualitative results with additional uncurated results provided in Appendix E.4. These results corroborate the quantitative findings featured in Table 1. MintFlow produces natural-looking reconstructions while faithfully preserving the measurement constraints. In contrast, baselines often produce blurry reconstructions or introduce visually implausible structures and artifacts, indicating deviations from the learned image distribution. Overall, the qualitative results demonstrate that MintFlow more effectively satisfies the measurement constraints while remaining faithful to the learned image distribution.

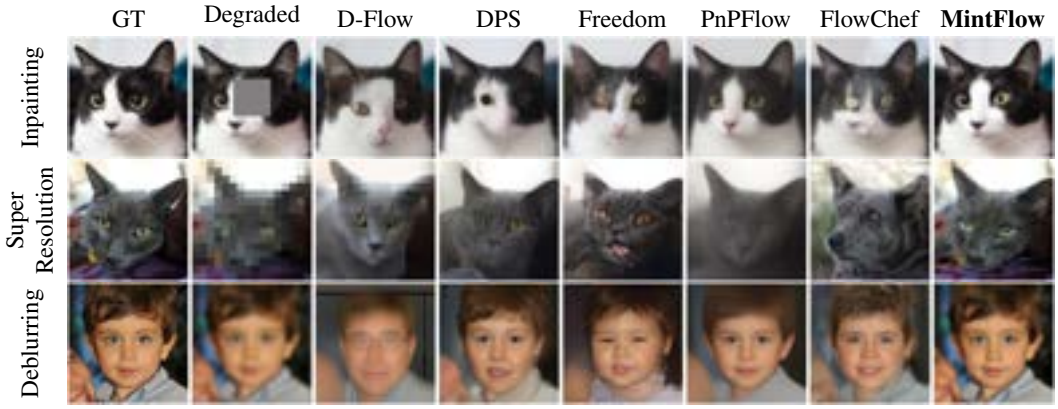


Figure 3: Representative qualitative results for image inverse problems. We compare ground-truth images (GT), degraded inputs, and results from different methods. Inpainting and super resolution are evaluated on AFHQ-Cat, while deblurring is evaluated on FFHQ.

4.2 Image Editing

Setup. Following Patel et al. (2025), we conduct image editing experiments on PIE-Bench (Ju et al., 2024) using InstaFlow (Liu et al., 2023c) as the flow-matching model. Given a reference image, mask, and text guidance, the task requires preserving the unmasked regions while editing the masked region according to the provided text prompt. Thus, the primary challenge is to simultaneously satisfy the preservation constraint and the text-guided editing objective. We compare against the same baselines as in Section 4.1, while utilizing the identical model and source noises. D-Flow is omitted due to its excessive GPU memory requirements. In addition to constraint satisfaction and efficiency, we evaluate prompt adherence and image similarity using CLIP scores, measuring how well the edited image follows the text prompt and remains visually consistent with the reference image, respectively. Additional details are provided in Appendix F.

Quantitative Results. Table 2 summarizes the quantitative comparison on the image editing task. MintFlow demonstrates robust overall performance across the evaluated metrics. In particular, it attains the best constraint satisfaction, indicating it more faithfully preserves the unmasked regions while performing the desired edit. It also achieves the highest prompt adherence, demonstrating that the resulting images better follow the given text guidance. Although Freedom achieves higher image similarity, it comes at the cost of lower prompt adherence, indicating a stronger tendency to preserve the reference image. As shown in Figure 4, semantic elements that conflict with the prompt, such as the dog's ears, tongue, and power-plant smoke, are retained. MintFlow, in contrast, provides a more favorable balance between preserving the reference image and following the editing instruction. MintFlow also is the most computationally efficient method, achieving the lowest inference time while using the least

Table 2: Comparison on image editing task.

Method	H ↓	CLIP-T ↑	CLIP-I ↑	Time ↓	Memory ↓
DPS	0.0115	0.2916	0.7911	9.17	8881.75
Freedom	0.0071	0.2756	0.8466	17.65	8122.47
PNPFlow	0.0503	0.2918	0.6897	42.45	6507.95
FlowChef	0.0136	0.2891	0.7939	9.29	6507.26
MintFlow	0.0069	0.2926	0.8151	8.79	6507.26

Figure 4: Qualitative comparison of image editing results. Given an input image and its corresponding text prompt, we compare the editing results produced by different methods. Strikethrough and red text indicate the original and edited text, respectively.

Table 3: Physical system modeling benchmark results. Performance evaluation across $N = 1000$ generated samples comparing System 1 (Heat Equation) and System 2 (Reaction-Diffusion Equation). Lower values (↓) indicate superior performance.

	System 1: Heat Equation					System 2: Reaction-Diffusion Equation				
Method	**FID ↓**	**MMD ↓**	**W_2 ↓**	**H ↓**	**Sim. Error ↓**	**FID ↓**	**MMD ↓**	**W_2 ↓**	**H ↓**	**Sim. Error ↓**
Vanilla FFM	1.255E−3	2.496E−2	4.872E−2	5.16E−3	5.57E−2	1.74E−4	2.130E−2	2.078E−2	1.907E0	3.55E−2
DiffusionPDE	4.712E−3	5.361E−2	7.431E−2	1.86E−2	8.49E−2	9.34E−4	6.063E−2	3.712E−2	9.56E−1	3.29E−2
D-Flow	1.290E−3	2.554E−2	4.098E−2	4.71E−3	4.34E−2	3.95E−4	3.861E−2	4.085E−2	4.44E−1	3.02E−2
ECI	1.865E−1	2.067E−1	4.502E−1	2.27E−7	4.47E−1	1.829E−2	1.299E−1	1.647E−1	8.02E−6	2.28E−1
PCFM	1.851E−1	2.062E−1	4.485E−1	1.30E−7	4.45E−1	1.831E−2	1.301E−1	1.648E−1	8.06E−6	2.28E−1
MintFlow	2.91E−5	5.44E−3	1.455E−2	3.57E−6	1.23E−2	1.09E−4	1.342E−2	1.870E−2	3.18E−4	2.58E−2

memory. This demonstrates the scalability and efficiency of MintFlow even for text-guided image editing. Overall, these results show that MintFlow can simultaneously satisfy the preservation constraint, adhere closely to the editing prompt, and perform the task efficiently, providing a favorable balance between editing quality and computational cost.

Qualitative Comparison. Representative qualitative results are presented in Figure 4, with additional results in Appendix F.4. As shown, baseline methods often struggle to maintain structural and visual coherence, producing artifacts such as prompt-inconsistent semantic elements retained from the reference image (e.g., untransformed ears), anatomical inconsistencies (e.g., misplaced eyes), and unnatural color shifts or blurring. In contrast, our method performs the desired edits in accordance with the given prompt while preserving visual coherence and image quality. These results demonstrate that our approach effectively satisfies the prompt constraints while preserving the contextual information of the input image without straying outside of the learned data distribution.

4.3 Physical System Modeling

Setup. Following the benchmark protocol of Utkarsh et al. (2026), we evaluate MintFlow on physical-system generation using *Functional Flow Matching* (FFM), which extends flow matching to function-valued data (Kerrigan et al., 2023). We consider two spatiotemporal systems: the periodic heat and a one-dimensional reaction–diffusion (RD1D) equations. Each system uses its own pretrained FFM checkpoint, and all samplers share the same frozen weights and source noises. The constraint operator jointly encodes conservation, initial-condition (IC), and boundary-condition (BC) residuals. Heat uses structural periodicity, while RD1D imposes prescribed Neumann fluxes. We compare MintFlow against Vanilla FFM, DiffusionPDE, D-Flow, ECI, and PCFM without retraining or fine-tuning the pretrained model. In addition to the introduced core criteria, we evaluate physical consistency using the governing-equation residual and the error to a numerical trajectory initialized from each generated sample's own IC. Distributional agreement is measured against a numerical reference set generated under the same FFM-derived physical conditions, so the reported set-level distances quantify agreement with the corresponding conditioned numerical distribution rather than divergence from Vanilla FFM itself. Further details are provided in Appendix G.

Quantitative Results. Table 3 presents the compact main comparison using three distribution metrics, the mean constraint residual, and mean simulation error. Table 10 additionally reports KID and partial differential equation (PDE) residuals together with sample-level standard deviations for the constraint and physical-consistency metrics. MintFlow has the smallest FID, MMD, W_2, and simulation-error point estimates on both systems, and the full Table 3 shows the same ordering for KID and PDE residual. Together, these rankings indicate improved agreement with the conditioned numerical reference and improved dynamical consistency. The constraint metric exposes the complementary trade-off: post-hoc projection methods, ECI and PCFM, obtain the smallest residuals, but their distribution and physical-consistency metrics deteriorate sharply. By contrast, MintFlow reduces the residual by roughly three orders of magnitude relative to Vanilla, DiffusionPDE, and D-Flow while avoiding ECI and PCFM's pronounced distributional shift and dynamical error. The results place MintFlow on a distinct Pareto point rather than establishing uniform dominance. ECI and PCFM remain preferable if minimizing the explicit constraint residual is the sole objective, whereas MintFlow has the lowest reported distribution and physical-consistency point estimates.

Qualitative Comparison. Figure 5 examines dynamical self-consistency at the sample level. For each source-noise index, the corresponding unconstrained FFM trajectory determines the target initial condition and the physical parameters required to define the numerical problem. These quantities are then held fixed across all samplers. For each generated field, we extract its own IC, integrate the corresponding PDE under the fixed sample-indexed physical parameters and BCs, and display the pointwise absolute discrepancy to this numerical rollout. The results show that ECI and PCFM produce broad, spatially coherent errors despite their small explicit constraint residuals. MintFlow remains visibly closer to its numerical rollout, consistent with its lower aggregate simulation error relative to these baselines. Appendix G.6 provides additional fixed-ID examples.

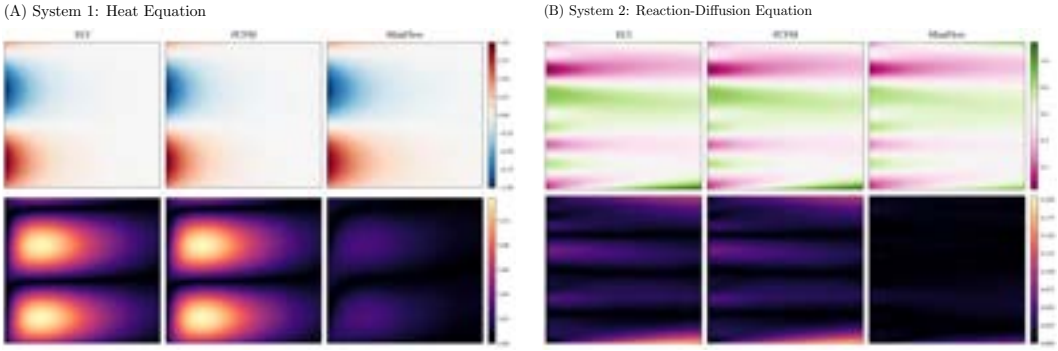


Figure 5: Qualitative physical consistency under physical constraints. The top row shows samples generated by ECI, PCFM, and MintFlow, while the bottom row illustrates the pointwise absolute discrepancy between the generated field and this sample-conditioned numerical rollout.

Model Analysis. Figure 6 ablates the sampling steps and regularization parameter λ in (8) on the Heat dataset. While the benchmark uses 100 Euler sampling steps by default, increasing the sampling resolution improves both distributional fidelity and constraint satisfaction. This is because discretization affects both the forward integration and the backward adjoint computation: finer sampling yields a more accurate trajectory and adjoint, leading to better preservation of the pretrained distribution and more accurate local constraint corrections. Nevertheless, the method remains relatively robust, with noticeable degradation only below 100 sampling steps. Varying λ reveals a trade-off between constraint satisfaction and distribution preservation. Since λ controls the preferred intervention time rather than the intervention magnitude directly, larger λ favors later interventions. When the intervention is too close to the terminal time, the correction approaches a direct projection of the terminal sample toward the constraint manifold, which can better satisfy the constraint but may deviate further from the pretrained distribution ν_T. The method remains stable for a moderate range of λ. Further details are provided in Appendix H.

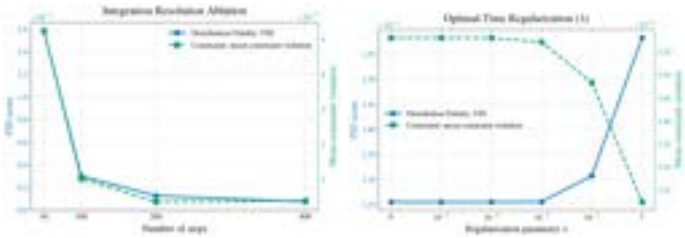


Figure 6: Sensitivity of MintFlow to sampling steps and regularization parameter λ.

5 Conclusion

This work introduces **MintFlow**, a training-free framework for constrained generation in flow matching models. By formulating constrained generation as a minimal trajectory intervention, MintFlow preserves the learned generative dynamics while minimally perturbing the flow state. Leveraging an adjoint-based sensitivity formulation, it efficiently computes the minimum-norm intermediate perturbation with theoretical analysis of constraint satisfaction and distributional stability. Extensive experiments across diverse domains, constraints, and backbone models demonstrate that MintFlow effectively satisfies constraints while preserving the pretrained distribution. While MintFlow relies

on a first-order linearization for high efficiency, it can be seamlessly complemented by iterative refinements or higher-order approximations when dealing with highly nonlinear constraints. Looking forward, extending MintFlow to more challenging nonlinear or non-differentiable constraints, reducing its computational overhead, and broadening the framework to other dynamics-based generative models pose promising directions for future work.

Acknowledgements. This work was supported by the U.S. Department of Energy (DOE), Office of Science, Advanced Scientific Computing Research program under award B&R-KJ0401010 and FWP-CC147 and grant DE-SC0026262.

Reproducibility Statement. We provide the necessary details to facilitate reproducibility of our method and experiments. The complete algorithms for single-step correction and iterative refinement are given in Algorithms 2 and 3. All theoretical assumptions and complete proofs are provided in Appendix A. Detailed experimental settings, including datasets, model configurations, hyperparameters, and evaluation protocols, are documented in Appendices D-G. Source code is provided as anonymous supplementary material.

AI Disclosure. We used generative AI tools solely to identify typographical and grammatical errors and to suggest improvements to the English wording of the manuscript. All suggestions were manually reviewed by the authors, and appropriate changes were incorporated by the authors. We did not use generative AI tools for any of the required-disclosure tasks listed in the ICLR 2027 AI policy. We take responsibility for the final content of this work.

REFERENCES

Heli Ben-Hamu, Omri Puny, Itai Gat, Brian Karrer, Uriel Singer, and Yaron Lipman. D-flow: Differentiating through flows for controlled generation. *International Conference on Machine Learning*, 2024.

Chaoran Cheng, Boran Han, Danielle Maddix, Abdul Fatir Ansari, Andrew Stuart, Michael W Mahoney, and Bernie Wang. Gradient-free generation for hard-constrained systems. In *International Conference on Learning Representations*, volume 2025, pp. 100510–100539, 2025.

Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. 2020.

Jacob K Christopher, Stephen Baek, and Ferdinando Fioretto. Constrained synthesis with projected diffusion models. *Advances in Neural Information Processing Systems*, 37:89307–89333, 2024.

Hyungjin Chung, Byeongsu Sim, Dohoon Ryu, and Jong Chul Ye. Improving diffusion models for inverse problems using manifold constraints. *Advances in neural information processing systems*, 35:25683–25696, 2022.

Hyungjin Chung, Jeongsol Kim, Michael T. Mccann, Marc L. Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. *International Conference on Learning Representations*, 2023a.

Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *The Eleventh International Conference on Learning Representations*, 2023b. URL https://openreview.net/forum?id=OnD9zGAGT0k.

Earl A Coddington. Theory of ordinary differential equations. *McGraw-Hill*, 1955.

Ruiqi Feng, Chenglei Yu, Wenhao Deng, Peiyan Hu, and Tailin Wu. On the guidance of flow matching. *International Conference on Machine Learning*, 2025.

Weiguo Gao, Ming Li, and Qianxiao Li. Terminally constrained flow-based generative models from an optimal control perspective. *arXiv preprint arXiv:2601.09474*, 2026.

Xuan Ju, Ailing Zeng, Yuxuan Bian, Shaoteng Liu, and Qiang Xu. Pnp inversion: Boosting diffusion-based editing with 3 lines of code. In *International Conference on Learning Representations*, volume 2024, pp. 23395–23422, 2024.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

Gavin Kerrigan, Giosue Migliorini, and Padhraic Smyth. Functional flow matching. *arXiv preprint arXiv:2305.17209*, 2023.

Jeongsol Kim, Bryan Sangwoo Kim, and Jong Chul Ye. FlowDPS: Flow-driven posterior sampling for inverse problems. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12328–12337. IEEE, 2025.

Alaina Kolli, Theodoros Xenakis, Utkarsh Utkarsh, Pengfei Cai, Rafael Gomez-Bombarelli, Alan Edelman, and Christopher Vincent Rackauckas. SNAP-FM: Sparse nonlinear accelerated projection for physics-constrained generative modeling. *arXiv preprint arXiv:2607.00095*, 2026.

Vince Kurtz and Alexander Davydov. Constrained flow matching via lagrangian dual flows. *arXiv preprint arXiv:2607.04513*, 2026.

Sangyun Lee, Zinan Lin, and Giulia Fanti. Improving the training of rectified flows. *Advances in Neural Information Processing Systems*, 2024.

Xinpeng Li, Enming Liang, and Minghua Chen. Gauge flow matching: Efficient constrained generative modeling over general convex set and beyond. In *The Fourteenth International Conference on Learning Representations*, 2026a.

Zeyang Li, Kaveh Alim, and Navid Azizan. Hardflow: Hard-constrained sampling for flow-matching models via trajectory optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026b.

Jinhao Liang, Yixuan Sun, Anirban Samaddar, Sandeep Madireddy, and Ferdinando Fioretto. Chance-constrained flow matching for high-fidelity constraint-aware generation. *arXiv preprint arXiv:2509.25157*, 2025.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *International Conference on Learning Representations*, 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. *Advances in neural information processing systems*, 38:40783–40818, 2026.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *International Conference on Learning Representations*, 2023a.

Xingchao Liu, Lemeng Wu, Shujian Zhang, Chengyue Gong, Wei Ping, and Qiang Liu. Flowgrad: Controlling the output of generative odes with gradients. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24335–24344. IEEE, 2023b.

Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, et al. Instaflow: One step is enough for high-quality diffusion-based text-to-image generation. In *The twelfth international conference on learning representations*, 2023c.

Ségolène Martin, Anne Gagneux, Paul Hagemann, and Gabriele Steidl. Pnp-flow: Plug-and-play image restoration with flow matching. In *International Conference on Learning Representations*, volume 2025, pp. 45466–45492, 2025a.

Ségolène Martin, Anne Gagneux, Paul Hagemann, and Gabriele Steidl. Pnp-flow: Plug-and-play image restoration with flow matching. *International Conference on Learning Representations*, 2025b.

Maitreya Patel, Song Wen, Dimitris N. Metaxas, and Yezhou Yang. Steering rectified flow models in the vector field for controlled image generation. *International Conference on Computer Vision*, 2025.

Ashwini Pokle, Matthew J. Muckley, Ricky T. Q. Chen, and Brian Karrer. Training-free linear image inverses via flows. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=PLIt3a4yTm.

Lev Semenovich Pontryagin. *Mathematical theory of optimal processes*. CRC press, 1987.

Mehrsa Pourya, Bassam El Rawas, and Michael Unser. FLOWER: A flow-matching solver for inverse problems. In *International Conference on Learning Representations*, volume 2026, pp. 91318–91343, 2026.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Jiaming Song, Arash Vahdat, Morteza Mardani, and Jan Kautz. Pseudoinverse-guided diffusion models for inverse problems. In *International conference on learning representations*, 2023.

Kaiyu Song, Hanjiang Lai, Yan Pan, Kun Yue, and Jian Yin. Flow-matching posterior sampling: A training-free conditional generation for flow matching. *IEEE Transactions on Image Processing*, 35:5970–5981, 2026. doi: 10.1109/TIP.2026.3698367. URL https://arxiv.org/abs/2411.07625.

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

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

Luran Wang, Chaoran Cheng, Yizhen Liao, Yanru Qu, and Ge Liu. Training free guided flow matching with optimal control. *arXiv preprint arXiv:2410.18070*, 2024.

Yici Yan, Yichi Zhang, Xiangming Meng, and Zhizhen Zhao. FIG: Flow with interpolant guidance for linear inverse problems. In *International Conference on Learning Representations*, volume 2025, pp. 17635–17659, 2025.

Jiwen Yu, Yinhuai Wang, Chen Zhao, Bernard Ghanem, and Jian Zhang. Freedom: Training-free energy-guided conditional diffusion model. *International Conference on Computer Vision*, 2023.

Zichao Yu, Ming Li, Wenyi Zhang, Difan Zou, and Weiguo Gao. ProFlow: Zero-shot physics-consistent sampling via proximal flow guidance. *arXiv preprint arXiv:2601.20227*, 2026.

Yasi Zhang, Peiyu Yu, Yaxuan Zhu, Yingshan Chang, Feng Gao, Ying N Wu, and Oscar Leong. Flow priors for linear inverse problems via iterative corrupted trajectory matching. *Advances in Neural Information Processing Systems*, 37:57389–57417, 2024.

A Theoretical Justification

A.1 Control-Theoretic Interpretation of the Minimum-Norm Correction

The proposed correction strategy admits a natural interpretation from the perspective of optimal control in (3) and constrained endpoint optimization. Introduce a Lagrangian η:

$$\frac{1}{2}|\delta|^2 + \eta^\top F_s(\delta),$$

the first-order optimality conditions are

$$\delta + DF_s(\delta)^\top \eta = 0, \quad F_s(\delta) = 0.$$

Therefore, the exact constrained correction is characterized by the nonlinear Karush–Kuhn–Tucker (KKT) system.

The proposed adjoint correction algorithm can be interpreted as a first-order approximation of the nonlinear KKT system associated with (3). Substituting the first-order Taylor expansion of $DF_s(\delta)$ and $F(\delta)$ into the KKT conditions gives

$$\delta + A_s^\top \eta = 0, \quad A_s\delta + H\left(u_T\right) = 0,$$

the first-order linearization of the nonlinear KKT system around the generated trajectory. Eliminating the Lagrange multiplier η yields the minimum-norm correction in (6). Therefore, the proposed correction is the solution of the linearized KKT system associated with the minimum-perturbation endpoint-constrained optimization problem.

A.2 Validity of the Linearized Correction

We provide the formal statements and proofs corresponding to the informal theoretical guarantees in Theorem 3.1. In particular, Theorem A.1 establishes the existence of the minimum-norm linearized correction and quantifies the resulting constraint satisfaction. Theorem A.2 further shows that the deviation of the corrected terminal distribution from the learned flow-matching distribution is controlled by the magnitude of the intermediate correction and the remaining integration time.

Theorem A.1 (Existence and Accuracy of the Linearized Correction)**.** *Assume that f_θ and H are twice continuously differentiable with bounded derivatives*

$$\|D_u f_\theta(u,t)\|_{\mathrm{op}} \le L_1, \qquad \|D_u^2 f_\theta(u,t)\|_{\mathrm{op}} \le L_2,$$

and

$$\|\nabla H(y)\| \le M_{H,1}, \qquad \|\nabla^2 H(y)\|_{\mathrm{op}} \le M_{H,2}.$$

and $A_s = DF_s(0)$ has full row rank. Then the linearized constraint

$$A_s\delta + H(u_T) = 0$$

admits a solution, and its unique minimum-norm solution is (6). Moreover, if

$$\frac{M_F}{2}\|\delta^*\|^2 < \|H(u_T)\|, \quad M_F := M_{H,2}e^{2L_1(T-s)} + M_{H,1}\frac{L_2}{L_1}\left(e^{2L_1(T-s)} - e^{L_1(T-s)}\right)$$

then

$$\|H(\Phi(T,s)(u_s+\delta^*))\| < \|H(u_T)\|.$$

Proof. For fixed u, define $J(t) := D_u\Phi(t,s)(u)$. The variational equation is

$$\frac{d}{dt}J(t) = D_u f_\theta(\Phi(t,s)(u),t)J(t), \qquad J(s) = I.$$

Hence, by Grönwall's inequality,

$$\|D_u\Phi(T,s)(u)\|_{\mathrm{op}} \le e^{L_1(T-s)}.$$

Differentiating the variational equation once more with respect to u gives

$$\frac{d}{dt}D_u^2\Phi(t,s)(u) = D_u f_\theta(\Phi(t,s)(u),t)D_u^2\Phi(t,s)(u) + D_u^2 f_\theta(\Phi(t,s)(u),t)[D_u\Phi(t,s)(u), D_u\Phi(t,s)(u)],$$

with $D_u^2\Phi(s,s)(u) = 0$. It follows that

$$\frac{d}{dt}\|D_u^2\Phi(t,s)(u)\|_{\mathrm{op}} \le L_1\|D_u^2\Phi(t,s)(u)\|_{\mathrm{op}} + L_2 e^{2L_1(t-s)}.$$

Another application of Grönwall's inequality yields

$$\begin{aligned}\|D_u^2\Phi(T,s)(u)\|_{\mathrm{op}} &\le e^{L_1(T-s)}\int_s^T L_2 e^{L_1(t-s)}\,dt\\ &= \frac{L_2}{L_1}e^{L_1(T-s)}\left(e^{L_1(T-s)}-1\right)\\ &= \frac{L_2}{L_1}\left(e^{2L_1(T-s)}-e^{L_1(T-s)}\right).\end{aligned}$$

Let $\phi(\delta) := \Phi(T,s)(u_s+\delta)$ The second-order chain rule gives, for $v, w \in \mathbb{R}^d$,

$$D^2F_s(\delta)[v,w] = D^2H(\phi(\delta))[D\phi(\delta)v, D\phi(\delta)w] + DH(\phi(\delta))[D^2\phi(\delta)[v,w]].$$

Thus,

$$\begin{aligned}\|D^2F_s(\delta)\|_{\mathrm{op}} &\le \|\nabla^2H(\phi(\delta))\|_{\mathrm{op}}\|D\phi(\delta)\|_{\mathrm{op}}^2 + \|\nabla H(\phi(\delta))\|\|D^2\phi(\delta)\|_{\mathrm{op}}\\ &\le M_{H,2}e^{2L_1(T-s)} + M_{H,1}\frac{L_2}{L_1}\left(e^{2L_1(T-s)}-e^{L_1(T-s)}\right) =: M_F.\end{aligned}$$

Since

$$u_T = \Phi(T,s)(u_s),$$

we have

$$F_s(0) = H(\Phi(T,s)(u_s)) = H(u_T).$$

The first-order Taylor expansion of F_s around $\delta = 0$ gives

$$F_s(\delta) = F_s(0) + DF_s(0)\delta + R(\delta^*), \tag{10}$$

where the above estimation gives

$$\|R(\delta^*)\| \le \frac{M_F}{2}\|\delta^*\|^2.$$

Therefore

$$F_s(\delta) = H(u_T) + A_s\delta + O(\|\delta\|^2).$$

Consider the linearized constraint

$$A_s\delta + H(u_T) = 0.$$

Since $A_s \in \mathbb{R}^{m\times d}$ has full row rank,

$$-H(u_T) \in \mathrm{Range}(A_s).$$

Therefore, the linearized constraint admits a solution.

The minimum-norm solution of

$$A_s\delta = -H(u_T)$$

is given by the Moore–Penrose pseudoinverse:

$$\delta_s^* = -A_s^\dagger H(u_T).$$

Indeed, the general solution can be written as

$$\delta = -A_s^\dagger H(u_T) + \left(I - A_s^\dagger A_s\right)z, \qquad z \in \mathbb{R}^d.$$

The first term belongs to

$$\mathrm{Range}(A_s^\top) = \mathrm{Null}(A_s)^\perp,$$

whereas

$$\left(I - A_s^\dagger A_s\right)z \in \mathrm{Null}(A_s).$$

Thus, the two terms are orthogonal, and

$$\|\delta\|^2 = \left\|A_s^\dagger H(u_T)\right\|^2 + \left\|\left(I - A_s^\dagger A_s\right)z\right\|^2.$$

Therefore, the norm is uniquely minimized when

$$\left(I - A_s^\dagger A_s\right) z = 0,$$

which yields

$$\delta_s^* = -A_s^\dagger H(u_T).$$

Using

$$F_s(0) = H(u_T), \qquad DF_s(0) = A_s,$$

the linearization (10) gives

$$F_s(\delta_s^*) = H(u_T) + A_s\delta_s^* + R_s(\delta_s^*).$$

By construction of δ_s^*,

$$A_s\delta_s^* = -H(u_T).$$

Hence the zeroth- and first-order terms cancel, giving

$$F_s(\delta_s^*) = R_s(\delta_s^*).$$

Consequently,

$$\|F_s(\delta^*)\| \le \frac{M_F}{2}\|\delta^*\|^2.$$

and therefore

$$F_s(\delta_s^*) = O\Big(\|\delta_s^*\|^2\Big).$$

Thus, the minimum-norm solution of the linearized constraint satisfies the original nonlinear constraint up to a second-order residual. □

The assumptions on f_θ and H impose standard smoothness conditions needed for our perturbation analysis. Bounded first- and second-order derivatives are reasonable for smooth neural networks and differentiable constraint operators, at least on the bounded region traversed by the generated trajectories. The full-row-rank condition on $A_s = DF_s(0)$ is a standard local non-degeneracy assumption, ensuring that the constraint directions are locally independent and that the constrained solution is well-defined. However, this condition may not hold for general constraints, particularly for highly nonlinear constraints where the Jacobian can become rank-deficient at certain points. Thus, the full-row-rank assumption should be viewed as a local regularity condition that restricts the analysis to locally regular constraint configurations.

The following theorem states that a small intermediate correction induces a controlled deviation from the terminal distribution ν_T generated by the pretrained flow.

Theorem A.2 (Distributional Stability under Intermediate Correction)**.** *Let $u_s \sim \nu_s$ and define the uncorrected terminal state by*

$$u_T = \Phi(T, s)(u_s),$$

so that $u_T \sim \nu_T$. Let

$$u_s^* = u_s + \delta_s^*(u_s)$$

be a measurable intermediate correction, and define the corrected terminal state by

$$u_T^* = \Phi(T, s)(u_s^*) = \Phi(T, s)\big(u_s + \delta_s^*(u_s)\big).$$

Let ν_T^* denote the distribution of u_T^*.

Assume that the flow field f_θ is globally L_f-Lipschitz with respect to its state variable, uniformly in $t \in [s, T]$, i.e.,

$$\|f_\theta(u, t) - f_\theta(v, t)\| \le L_f\|u - v\|, \qquad \forall u, v, \quad t \in [s, T].$$

Then, for every u_s,

$$\|u_T^* - u_T\| \le e^{L_f(T-s)}\|\delta_s^*(u_s)\|.$$

Consequently, the corrected terminal distribution satisfies

$$W_2(\nu_T^*, \nu_T) \le e^{L_f(T-s)}\|\delta_s^*\|_{L^2(\nu_s)}.$$

Proof. Fix $u_s \in \mathbb{R}^d$ and consider the uncorrected and corrected trajectories

$$u(t) = \Phi(t,s)(u_s), \qquad u^*(t) = \Phi(t,s)\big(u_s + \delta_s^*(u_s)\big), \qquad t \in [s,T].$$

Define the difference by

$$e(t) = u^*(t) - u(t).$$

Since both trajectories evolve according to the same flow field,

$$\frac{d}{dt}e(t) = f_\theta(u^*(t),t) - f_\theta(u(t),t).$$

By the L_f-Lipschitz assumption,

$$\left\| \frac{d}{dt}e(t) \right\| \le L_f \|e(t)\|.$$

Moreover, at the correction time s,

$$e(s) = u_s^* - u_s = \delta_s^*(u_s).$$

Therefore,

$$\frac{d}{dt}\|e(t)\| \le L_f \|e(t)\|$$

for almost every $t \in [s,T]$. Applying Grönwall's inequality gives

$$\|e(T)\| \le e^{L_f(T-s)}\|e(s)\|.$$

Since

$$e(T) = u_T^* - u_T$$

and

$$e(s) = \delta_s^*(u_s),$$

we obtain the pointwise bound

$$\|u_T^* - u_T\| \le e^{L_f(T-s)}\|\delta_s^*(u_s)\|.$$

We next establish the distributional bound. Since $u_s \sim \nu_s$ and $u_T = \Phi(T,s)(u_s)$, the pushforward property of the flow gives

$$u_T \sim \Phi(T,s)_\# \nu_s = \nu_T.$$

Similarly, by definition,

$$u_T^* = \Phi(T,s)\big(u_s + \delta_s^*(u_s)\big) \sim \nu_T^*.$$

Hence the joint random variable (u_T, u_T^*) defines a coupling of ν_T and ν_T^*. By the definition of the Wasserstein-2 distance,

$$W_2^2(\nu_T^*, \nu_T) \le \mathbb{E}\left[\|u_T^* - u_T\|^2\right].$$

Applying the pointwise stability bound gives

$$\begin{aligned} W_2^2(\nu_T^*, \nu_T) &\le \mathbb{E}\left[e^{2L_f(T-s)}\|\delta_s^*(u_s)\|^2\right] \\ &= e^{2L_f(T-s)}\mathbb{E}_{u_s\sim\nu_s}\left[\|\delta_s^*(u_s)\|^2\right]. \end{aligned}$$

Taking the square root yields

$$W_2(\nu_T^*, \nu_T) \le e^{L_f(T-s)}\left(\mathbb{E}_{u_s\sim\nu_s}\left[\|\delta_s^*(u_s)\|^2\right]\right)^{1/2},$$

which is equivalently

$$W_2(\nu_T^*, \nu_T) \le e^{L_f(T-s)}\|\delta_s^*\|_{L^2(\nu_s)}.$$

□

The global Lipschitz assumption on f_θ is a standard regularity condition for neural ODEs, ensuring existence, uniqueness, and stability of the flow trajectories. It is generally reasonable for smooth neural networks on bounded regions containing the generated trajectories. However, global Lipschitzness over the entire state space can be restrictive for practical neural networks. Thus, the assumption can be interpreted as requiring Lipschitz continuity only on the relevant region of the state space explored by the flow.

A.3 Convergence of Iterative Refinement

Theorem A.3 (Convergence of Iterative Refinement)**.** *Let $f_\theta : \mathbb{R}^d \times [s,T] \to \mathbb{R}^d$ be C^2 in u, let $H : \mathbb{R}^d \to \mathbb{R}$ be C^2, and set $\Delta = T - s$. Suppose that, on the relevant reachable set,*

$$\|D_u f_\theta(u,t)\|_{\mathrm{op}} \le L_1, \qquad \|D_u^2 f_\theta(u,t)\|_{\mathrm{op}} \le L_2,$$

and

$$0 < m_H \le \|\nabla H(y)\| \le M_{H,1}, \qquad \|\nabla^2 H(y)\|_{\mathrm{op}} \le M_{H,2}.$$

Then, the correction provided by (6)

$$u^{(n+1)} = u^{(n)} + \delta^{(n)}$$

satisfies

$$|F(u^{(n+1)})| \le K|F(u^{(n)})|^2,$$

where

$$K = \frac{1}{2m_H^2}\left[M_{H,2}e^{4L_1\Delta} + M_{H,1}\frac{L_2}{L_1}\left(e^{4L_1\Delta} - e^{3L_1\Delta}\right)\right].$$

Consequently, if $K|F(u^{(0)})| < 1$, then

$$|F(u^{(n)})| \le \frac{1}{K}\left(K|F(u^{(0)})|\right)^{2^n}.$$

In particular, a sufficient number of iterations for $|F(u^{(n)})| \le \epsilon$ is

$$n \ge \log_2\left(\frac{\log(1/(K\epsilon))}{\log(1/(K|F(u^{(0)})|))}\right).$$

Proof. Let $\phi(u) := \Phi(T,s)(u)$. For fixed u, define $J(t) := D_u\Phi(t,s)(u)$. The variational equation is

$$\frac{d}{dt}J(t) = D_u f_\theta(\Phi(t,s)(u),t)J(t), \qquad J(s) = I.$$

Hence, by Grönwall's inequality,

$$\|D_u\Phi(T,s)(u)\|_{\mathrm{op}} \le e^{L_1\Delta}.$$

Since the flow is a diffeomorphism,

$$D_u\Phi(T,s)(u)^{-1} = D_y\Phi(s,T)(y), \qquad y = \Phi(T,s)(u),$$

and the same estimate applied to the inverse flow gives

$$\|D_u\Phi(T,s)(u)^{-1}\|_{\mathrm{op}} \le e^{L_1\Delta}.$$

Therefore,

$$\sigma_{\min}(D_u\Phi(T,s)(u)) = \frac{1}{\|D_u\Phi(T,s)(u)^{-1}\|_{\mathrm{op}}} \ge e^{-L_1\Delta}.$$

Next, differentiating the variational equation once more with respect to u, we obtain

$$\frac{d}{dt}D_u^2\Phi(t,s)(u) = D_u f_\theta(\Phi(t,s)(u),t)D_u^2\Phi(t,s)(u) + D_u^2 f_\theta(\Phi(t,s)(u),t)[D_u\Phi(t,s)(u), D_u\Phi(t,s)(u)],$$

with $D_u^2\Phi(s,s)(u) = 0$. Using

$$\|D_u\Phi(t,s)(u)\|_{\mathrm{op}} \le e^{L_1(t-s)},$$

we obtain

$$\frac{d}{dt}\|D_u^2\Phi(t,s)(u)\|_{\mathrm{op}} \le L_1\|D_u^2\Phi(t,s)(u)\|_{\mathrm{op}} + L_2 e^{2L_1(t-s)}.$$

Another application of Grönwall's inequality yields

$$\begin{aligned}\|D_u^2\Phi(T,s)(u)\|_{\mathrm{op}} &\le e^{L_1\Delta}\int_s^T L_2 e^{L_1(t-s)}\,dt \\ &= \frac{L_2}{L_1}e^{L_1\Delta}\left(e^{L_1\Delta}-1\right) \\ &= \frac{L_2}{L_1}\left(e^{2L_1\Delta}-e^{L_1\Delta}\right).\end{aligned}$$

By the chain rule gives $\nabla F(u) = D\phi(u)^T\nabla H(\phi(u))$, it follows that

$$\|\nabla F(u)\| \ge \sigma_{\min}(D\phi(u))\|\nabla H(\phi(u))\| \ge m_H e^{-L_1\Delta}.$$

The second-order chain rule gives, for $v, w \in \mathbb{R}^d$,

$$D^2F(u)[v,w] = D^2H(\phi(u))[D\phi(u)v, D\phi(u)w] + DH(\phi(u))[D^2\phi(u)[v,w]].$$

Thus,

$$\begin{aligned}\|D^2F(u)\|_{\mathrm{op}} &\le \|\nabla^2 H(\phi(u))\|_{\mathrm{op}}\|D\phi(u)\|_{\mathrm{op}}^2 + \|\nabla H(\phi(u))\|\|D^2\phi(u)\|_{\mathrm{op}} \\ &\le M_{H,2}e^{2L_1\Delta} + M_{H,1}\frac{L_2}{L_1}\left(e^{2L_1\Delta}-e^{L_1\Delta}\right) =: M_F.\end{aligned}$$

Since the optimal perturbation is given by

$$\delta^{(n)} := u^{(n+1)} - u^{(n)} = -\frac{F(u^{(n)})}{\|\nabla F(u^{(n)})\|^2}\nabla F(u^{(n)}),$$

we have

$$\|\delta^{(n)}\| = \frac{|F(u^{(n)})|}{\|\nabla F(u^{(n)})\|} \le \frac{e^{L_1\Delta}}{m_H}|F(u^{(n)})|.$$

By Taylor's theorem,

$$F(u^{(n)}+\delta^{(n)}) = F(u^{(n)}) + \nabla F(u^{(n)})^T\delta^{(n)} + R^{(n)},$$

where

$$|R^{(n)}| \le \frac{M_F}{2}\|\delta^{(n)}\|^2.$$

Since $\delta^{(n)}$ satisfies

$$\nabla F(u^{(n)})^T\delta^{(n)} = -F(u^{(n)}),$$

it follows that

$$F(u^{(n+1)}) = R^{(n)},$$

and therefore

$$|F(u^{(n+1)})| \le \frac{M_F}{2}\|\delta^{(n)}\|^2 \le \frac{M_F e^{2L_1\Delta}}{2m_H^2}|F(u^{(n)})|^2.$$

Substituting the definition of M_F gives

$$|F(u^{(n+1)})| \le K|F(u^{(n)})|^2,$$

where

$$K = \frac{1}{2m_H^2}\left[M_{H,2}e^{4L_1\Delta} + M_{H,1}\frac{L_2}{L_1}\left(e^{4L_1\Delta}-e^{3L_1\Delta}\right)\right].$$

Finally, let $r^{(n)} := |F(u^{(n)})|$. Then $r^{(n+1)} \le K\left(r^{(n)}\right)^2$. If $Kr^{(0)} < 1$, induction gives

$$r^{(n)} \le \frac{1}{K}(Kr^{(0)})^{2^n}.$$

Indeed, if the bound holds at iteration n, then

$$r^{(n+1)} \le K \left[\frac{1}{K} (K r^{(0)})^{2^n} \right]^2 = \frac{1}{K} (K r^{(0)})^{2^{n+1}}.$$

Thus,

$$|F(u^{(n)})| \le \epsilon$$

whenever

$$(K r^{(0)})^{2^n} \le K\epsilon.$$

Since $K r^{(0)} < 1$, this is equivalent to

$$2^n \ge \frac{\log(1/(K\epsilon))}{\log(1/(K r^{(0)}))},$$

which proves the stated iteration bound and the asymptotic estimate

$$N_\epsilon = O\left(\log\log \frac{1}{\epsilon} \right).$$

□

A.4 Specialization to Rectified Flow

The proposed trajectory intervention admits a particularly simple form when the pretrained flow is a rectified flow (RF) model and sampling is performed using an Euler discretization. Specifically, for a single sampling step from an intermediate time s to the terminal time T, the RF sampler takes the form

$$u_T = u_s + (T - s) v_\theta(u_s, s), \tag{11}$$

where v_θ denotes the learned RF velocity field. After applying an intermediate perturbation δ, the corresponding terminal state becomes

$$u_T^\delta = u_s + \delta + (T - s) v_\theta(u_s + \delta, s). \tag{12}$$

Consequently, the terminal constraint can be written as

$$F_s(\delta) = H\left(u_s + \delta + (T - s) v_\theta(u_s + \delta, s)\right). \tag{13}$$

Unlike the general continuous-time formulation, the sensitivity of the terminal state with respect to the intermediate perturbation can be obtained directly from the local Jacobian of the RF velocity field. In particular,

$$\left. \frac{\partial u_T^\delta}{\partial \delta} \right|_{\delta=0} = I + (T - s) J_v(u_s, s), \qquad J_v(u_s, s) \coloneqq \frac{\partial v_\theta}{\partial u}(u_s, s). \tag{14}$$

Therefore, the constraint Jacobian in (3) becomes

$$A_s = \nabla H(u_T) \left[I + (T - s) J_v(u_s, s) \right]. \tag{15}$$

For a scalar-valued constraint H, define the pulled-back constraint gradient

$$g_s \coloneqq \left[I + (T - s) J_v(u_s, s) \right]^\top \nabla H(u_T)^\top. \tag{16}$$

The minimum-norm linearized correction is then

$$\delta_s^* = -\frac{g_s}{\|g_s\|^2} H(u_T), \qquad g_s \ne 0. \tag{17}$$

Importantly, computing g_s does not require explicitly forming the full Jacobian J_v. It can be obtained by a single vector–Jacobian product through the pretrained velocity network. Thus, for a one-step RF sampler, the continuous adjoint ODE can be replaced by a local discrete sensitivity computation.

More generally, when the RF sampler uses multiple Euler steps,

$$u_{k+1} = u_k + \Delta t_k \, v_\theta(u_k, t_k), \tag{18}$$

the sensitivity from an intervention at step k to the terminal state is given by the product of the corresponding local Jacobians,

$$D\Phi_{\mathrm{RF}}(1, t_k) = \prod_{j=k}^{K-1} \left[I + \Delta t_j J_v(u_j, t_j) \right]. \tag{19}$$

Its transpose can be computed efficiently by a discrete adjoint or standard reverse-mode differentiation through the remaining sampling steps. Hence, the proposed method naturally extends from the continuous-time adjoint formulation to the discrete RF sampler used in practice.

Finally, (17) reveals the relationship between our method and ordinary terminal projection. If the RF velocity field is locally insensitive to the state, such that

$$J_v(u_s, s) \approx 0, \tag{20}$$

then

$$g_s \approx \nabla H \left(u_T \right)^\top, \tag{21}$$

and the proposed correction reduces to the standard minimum-norm projection onto the linearized constraint manifold at the terminal state. In contrast, when $J_v(u_s, s)$ is non-negligible, the factor

$$\left[I + (T - s) J_v(u_s, s) \right]^\top \tag{22}$$

accounts for how an intermediate perturbation is transformed by the RF dynamics before reaching the terminal state. Thus, the proposed correction can be viewed as a *dynamics-aware projection* rather than a purely terminal post-hoc projection.

B ALGORITHM

We summarize the proposed constrained generation algorithm from Section 3. The single-correction procedure is presented in Algorithm 2, while its iterative extension is presented in Algorithm 3.

Algorithm 2 Adjoint-Based Optimal Correction

Require: Initial sample $u_0 \sim \mu$, vector field f_θ, constraint H, candidate times $0 < s_1 < \cdots < s_K < T$, trade-off parameter λ

Forward sampling

Solve

$$\dot{u} = f_\theta(u, t), \qquad u(0) = u_0,$$

from 0 to T, and store $\{u(s_k)\}_{k=1}^{K}$ and the terminal state u_T.

Compute the terminal residual

$$r = H(u_T).$$

Backward adjoint solve

Initialize

$$\lambda(T) = \nabla H(u_T)^\top.$$

Solve the adjoint ODE backward only once, from T to s_1 and store $\lambda(s_k)$ whenever a candidate time is reached:

$$\dot{\lambda} = -\left(\nabla_u f_\theta(u, t)\right)^\top \lambda.$$

Candidate evaluation

for $k = 1, \ldots, K$ **do**

Compute

$$A_{s_k} = \nabla H(u_T) D\Phi(T, s_k).$$

Compute

$$\delta_k^* = -A_{s_k}^\dagger r.$$

Evaluate

$$J_k = \|\delta_k^*\|^2 + \lambda(T - s_k)^2.$$

end for

Select

$$k^* = \arg\min_k J_k.$$

Re-integrate the flow from s_{k^*}:

$$u(s_{k^*}) \leftarrow u(s_{k^*}) + \delta_{k^*}^*,$$

via

$$\dot{u} = f_\theta(u, t), \qquad t \in [s_{k^*}, T].$$

return corrected sample u_T^{corr}.

Algorithm 3 Iterative Adjoint-Based Optimal Correction

Require: Initial sample $u_0 \sim \mu$, vector field f_θ, constraint H, candidate number K, iteration number N

$s_0 \leftarrow 0, \quad u(s_0) \leftarrow u_0$
for $i = 1, \ldots, N$ **do**
  Sample trajectory on $[s_{i-1}, T]$
  Select candidate times

$$s_{i-1} < s_1 < \cdots < s_K < T$$

  Solve the adjoint ODE once from T to s_1
  for $k = 1, \ldots, K$ **do**
    Compute δ_k^* by (6)
    Evaluate $J(s_k)$
  end for
  $s_i \leftarrow \arg\min_{s_k} J(s_k)$
  Re-integrate the flow from $u(s_i) \leftarrow u(s_i) + \delta_i^*$
end for
return corrected sample u_T^{corr}

C Related Work

We consider inference-time constraint enforcement for pretrained flow matching (Lipman et al., 2023) and rectified flow models (Liu et al., 2023a). Existing approaches can be broadly categorized according to how they incorporate constraints during sampling: modifying the velocity field, projecting onto the constraint manifold, or optimizing the source or intermediate states, as illustrated in Figures 2a, 2b, and 2c, respectively.

Velocity modification. A line of work modifies the pretrained velocity field to steer the generated trajectory toward a desired target or constraint. General flow matching guidance derives exact or approximate corrections for energy-tilted laws (Feng et al., 2025). Image inverse solvers correct the path toward the data manifold with gradient-based guidance (Chung et al., 2022; 2023b; Song et al., 2023), adapting the same idea to flow matching via likelihood gradients, endpoint estimates, measurement interpolants, or posterior-score surrogates (Pokle et al., 2024; Song et al., 2026; Yan et al., 2025; Kim et al., 2025). TOCFlow formulates terminal constraints as optimal control with geometry-aware feedback based on the downstream-flow metric (Gao et al., 2026). Lagrangian Dual Flows augment the ODE with dual and slack dynamics that drive equality and inequality residuals to zero near terminal time (Kurtz & Davydov, 2026). These methods directly alter the sampling dynamics and require recomputing constraint-specific velocity corrections for each constraint, whereas our method leaves the pretrained vector field unchanged.

Projection. Projection methods alternate flow integration with corrections that project the state onto a feasible or observation-consistent set (Christopher et al., 2024). ECI extrapolates an intermediate state to the endpoint, applies a constraint correction, and interpolates the result back into the trajectory (Cheng et al., 2025). PCFM combines terminal shooting with a Gauss-Newton projection and optimal-transport-based reinsertion (Utkarsh et al., 2026). ProFlow replaces the terminal correction with a proximal problem over physical and observational constraints, while CCFM projects onto time-dependent chance-feasible sets (Yu et al., 2026; Liang et al., 2025). Image inverse solvers combine data-consistency corrections with flow-path reprojection (Pourya et al., 2026), and SNAP-FM accelerates PCFM-style nonlinear projections through sparse optimization (Kolli et al., 2026). While projection can effectively enforce the constraints, the explicit state corrections may induce deviations from the distribution learned by the pretrained model. In contrast, our method avoids explicit projection and instead seeks to reduce constraint violations through small perturbations of the intermediate states.

Optimization. Another family formulates constraint enforcement as an optimization problem over the generated trajectory. These methods treat the source noise, intermediate states, or distributed controls as decision variables and optimize them to produce samples that satisfy the constraints.

Table 4: Overview of experiments and ablation studies. We evaluate our method across diverse tasks, datasets, and flow matching (FM) models, and further conduct ablation studies. Detailed experimental settings and results are provided in the corresponding sections.

Task	Details	Data	FM Method	Constraint
Image generation	App. D	MNIST, CIFAR-10	Flow Matching	Intensity, fidelity
Image inverse problems	App. E (Sec. 4.1)	AFHQ-Cat, FFHQ	Rectified Flow	Observation consistency
Image editing	App. F (Sec. 4.2)	PIE-Bench	Latent text-conditioned FM	Text-guided generation with region preservation
Physics-informed generation	App. G (Sec. 4.3)	Heat, RD1D PDE	Functional Flow Matching	Boundary condition, conservation laws
Ablation studies: App. H				

D-Flow optimizes the source noise by differentiating a terminal objective through the complete flow solver (Ben-Hamu et al., 2024). However, backpropagating through the entire sampling ODE requires computing gradients across all integration steps, making the optimization computationally expensive. FlowChef improves efficiency by adapting this formulation to rectified flows, which eliminates the need to backpropagate through the ODE integration (Patel et al., 2025). Rather than optimizing the source noise, FlowChef sequentially optimizes intermediate states. HardFlow solves receding-horizon trajectory-optimization problems with hard constraints on an estimated terminal state (Li et al., 2026b), whereas OC-Flow optimizes distributed controls using adjoint costates and a running deviation penalty (Wang et al., 2024). FlowGrad (Liu et al., 2023b) likewise optimizes distributed controls by backpropagating gradients from a terminal guidance loss. Other related works propagate terminal gradients to intermediate controls and optimize corrupted trajectories (Liu et al., 2023b; Zhang et al., 2024). These approaches leave the pretrained velocity field unchanged, allowing them to accommodate a broad range of constraints without altering the underlying sampling dynamics. However, solving the resulting optimization problems can be computationally demanding, and optimizing intermediate states may introduce deviations from the distribution learned by the pretrained model. In contrast, our method uses minimal perturbations of intermediate states and replaces iterative optimization with a single adjoint system, reducing deviations from the learned distribution while enabling more efficient constraint enforcement.

D Application 1: Image Generation

This appendix provides additional experiments on image generation that are not included in the main manuscript, including benchmark definitions, model and sampling configurations, evaluation protocols, and additional diagnostics. We evaluate a standard flow matching model on image generation tasks with two types of constraints: **color intensity constraints** and a **nonlinear fidelity constraint** defined through a pretrained autoencoder. Together with the experiments in the main manuscript, these results further demonstrate that our approach generalizes across different flow matching models and diverse image-domain constraints, including nonlinear constraints, highlighting the versatility of MintFlow.

As in the main experiments, we separately quantify two aspects of constrained generation quality:

- **Constraint satisfaction:** how accurately the generated image satisfies the prescribed constraint.
- **Distributional fidelity:** how well the distribution of generated images is preserved relative to the reference image distribution.

D.1 Datasets and Constraints

We use the standard **MNIST** and **CIFAR-10** datasets provided by `torchvision`. All images are converted to `float32` tensors and scaled to the range $[-1, 1]$.

Color Intensity Constraint. For an image $u \in [-1, 1]^{C \times H \times W}$, we define the channel-wise average intensity vector as

$$\bar{u} := \frac{1}{HW} \sum_{h=1}^{H} \sum_{w=1}^{W} u_{:,h,w} \in \mathbb{R}^C. \tag{23}$$

Given a target intensity vector $\mathbf{c} \in \mathbb{R}^C$, we impose the constraint

$$H(u; \mathbf{c}) \coloneqq \bar{u} - \beta = \frac{1}{HW} \sum_{h=1}^{H} \sum_{w=1}^{W} u_{:,h,w} - \mathbf{c} = 0. \tag{24}$$

For MNIST, where $C = 1$, we impose a scalar intensity constraint with a prescribed target $c \in \mathbb{R}$:

$$H(u) \coloneqq \frac{1}{HW} \sum_{h=1}^{H} \sum_{w=1}^{W} u_{1,h,w} - c = 0. \tag{25}$$

The target intensity c primarily controls the overall stroke thickness of the generated digits. For CIFAR-10, this constraint controls the overall color composition of the generated image through the mean intensity of each RGB channel.

For each dataset, we first compute the empirical distribution of the average intensity values over the original images. We then randomly sample multiple target values from the range between the 20th and 80th percentiles of this distribution. We restrict the target values to this central range to avoid overly rare intensity levels for which only a small fraction of the data naturally satisfy the constraint, thereby ensuring that the constraint remains sufficiently represented in the underlying data distribution.

AE Fidelity Constraint We further consider a fidelity-related constraint defined using an autoencoder (AE) trained on the ground-truth data. Let Enc and Dec denote the encoder and decoder, respectively. We define the reconstruction constraint as

$$H(u) = \mathrm{Dec}(\mathrm{Enc}(u)) - u = 0. \tag{26}$$

For a well-trained AE, the reconstruction error is expected to be small for samples that are consistent with the learned data distribution. Thus, enforcing this constraint encourages generated samples to preserve characteristics captured by the pretrained data manifold. We use this constraint not only to improve the fidelity of generated images, but also to examine the robustness of MintFlow under highly nonlinear and complex constraints defined by a trained neural network.

For MNIST, we train a convolutional AE with a 128-dimensional latent representation for 200 epochs using the Adam optimizer with a learning rate of 10^{-3}. The model is trained with the Smooth L1 reconstruction loss. Input images are normalized to $[-1, 1]$ during training.

For CIFAR-10, we train a deeper convolutional AE with a 256-dimensional latent representation for 200 epochs. The model is optimized using AdamW with a learning rate of 5×10^{-5} and weight decay 10^{-4}, together with a cosine learning-rate schedule. The model is trained using the L1 reconstruction loss. CIFAR-10 images are normalized to $[-1, 1]$ and randomly horizontally flipped during training.

The trained AEs achieve reconstruction losses of 0.0008 on MNIST and 0.00139 on CIFAR-10 on the test sets, indicating accurate reconstruction of the data and providing a reliable basis for the proposed AE fidelity constraint. This experiment assumes that the pretrained AE provides a meaningful representation of the underlying data distribution; its purpose is to evaluate the applicability of MintFlow to nonlinear constraints induced by a pretrained neural network, rather than to claim AE reconstruction error as a universally optimal measure of fidelity.

D.2 Flow Matching Model and Baselines

For image datasets, all flow matching models were implemented based on the official Flow Matching repository.[3] We used a U-Net architecture and Table 5 summarizes the the model configurations. For all experiments, models were optimized using the AdamW optimizer with a learning rate of 1×10^{-4} and momentum coefficients $\beta_1 = 0.9$ and $\beta_2 = 0.95$. The learning rate was kept constant throughout training using a constant learning rate scheduler. At inference time, we solve the flow ODE using 100 integration steps for all experiments.

We consider the same baselines as in Section 4.3. Specifically, DiffusionPDE guides the pretrained FM velocity using the gradient of the constraint, D-Flow optimizes the latent noise to satisfy the

[3] https://github.com/facebookresearch/flow_matching

Table 5: Flow Matching model configurations for image datasets.

Dataset	Channels	ResBlocks	Attention	Dropout	Channel Mult.
MNIST	64	2	$\{2\}$	0.1	$(1, 2, 2)$
CIFAR-10	128	4	$\{2\}$	0.3	$(2, 2, 2)$

constraint, while ECI and PCFM enforce the constraint through a final projection step. Although some of these methods were primarily developed for physics-informed constrained generation, their formulations are not inherently restricted to physics data and can, in principle, be applied to general data domains with differentiable constraints. We therefore include them here to examine their applicability to image-domain constraints and to provide a consistent empirical comparison. In particular, the color intensity constraint considered here shares the same global structure as the conservation constraints addressed by these methods, making it a natural setting for cross-domain evaluation. The AE fidelity constraint further examines how these approaches behave under a more complex and highly nonlinear constraint induced by a pretrained neural network. Our goal is not to suggest that image generation was the original design target of these methods, but rather to assess the generality of constrained generation approaches across different data domains and constraint structures. For our method on image data, we set the number of candidate intervention times to $K = 10$, uniformly spaced across the entire sampling trajectory. All image experiments were conducted on a single NVIDIA H200 GPU.

D.3 Evaluation Protocol

To evaluate the quality of constraint-generated samples, we consider two aspects: constraint satisfaction and preservation of the data distribution.

Constraint Satisfaction. We measure the constraint satisfaction using the average constraint value over generated samples. For each method, including the ground-truth conditional samples, we generate 2000 samples and report the average constraint value.

Conditional Distribution Preservation. To evaluate whether the generated samples preserve the distribution learned by the flow matching model under the imposed constraint, we measure the discrepancy between constraint-generated samples and reference samples. The goal of constraint generation is to obtain samples that satisfy the imposed constraint while remaining within the distribution learned by the pretrained flow matching model. Therefore, the reference distribution used for evaluation is not the ground-truth data distribution, but a conditional distribution constructed from samples generated by the pretrained flow matching model.

Specifically, we first generate samples from the pretrained flow matching model and collect 2,000 samples that satisfy the given constraint. These constraint-satisfying samples are used as reference samples for evaluation. We then measure the discrepancy between the constraint-generated samples and the reference samples to quantify how well the generated samples preserve the flow-matching-induced distribution.

For image data, we evaluate the conditional distribution matching using FID, KID, MMD, and Wasserstein-2 (W_2) distance. For MMD, we use a Gaussian RBF kernel:

$$k(x, y) = \exp\left(-\frac{|x - y|^2}{2\sigma^2}\right),$$

where σ^2 is determined using the median heuristic computed from pairwise distances between generated and reference samples. Since directly estimating the empirical W_2 distance becomes computationally prohibitive in high-dimensional feature spaces, we approximate each feature distribution by a Gaussian distribution using its empirical mean and covariance, and compute the exact W_2 distance between the resulting Gaussian approximations. This provides a computationally tractable measure of distributional discrepancy while capturing differences in both the first- and second-order statistics.

Table 6: Quantitative results on the average-intensity constrained MNIST dataset. Light green (green) indicates the top performer, while light red (red) represents the second-best performer across all methods for each metric.

	Distribution Metrics				Constraint	Efficiency	
Method	FID ↓	KID ↓	MMD ↓	W_2 ↓	H ↓	Time (s) ↓	Memory (GB) ↓
Unconstrained	0.879	0.02215	0.03158	0.938	2.79E-1	5.07	0.10
DiffusionPDE	0.899	0.06860	0.04427	1.064	5.36E-2	4.29	5.10
D-Flow	0.813	0.05342	0.04726	0.902	5.49E-2	703.40	125.27
ECI	1.851	0.09982	0.07863	1.361	3.55E-15	8.89	0.35
PCFM	1.736	0.12379	0.12064	1.317	1.43E-12	18.37	10.12
Ours	0.262	0.01812	0.01289	0.602	4.55E-5	6.66	0.74

For MNIST, feature-space metrics are computed using the penultimate-layer representations of a pretrained MNIST classifier. The classifier is a residual convolutional network trained on the MNIST training set for 200 epochs using the Adam optimizer with a learning rate of 10^{-3}. The extracted features correspond to the 64-dimensional representation before the final classification layer. For CIFAR-10, feature representations are extracted using a pretrained Inception-v3 network.

Efficiency. Runtime is measured as synchronized wall-clock time per generated sample, and memory is reported as peak allocated GPU memory.

D.4 Results

D.4.1 MNIST

Intensity Constraint. We evaluate MintFlow on MNIST under an average-intensity constraint with a target value of -0.6, generating 2,000 samples for each method. All methods generate images using the same pretrained flow-matching (FM) model. To construct a reference distribution for evaluating distributional metrics, we generate samples from the pretrained FM model and retain those that satisfy the target constraint, obtaining 2,000 constraint-satisfying samples. We use these samples as the ground-truth samples from the conditional distribution induced by the pretrained flow model. Table 6 summarizes the quantitative results, while Figure 8 presents qualitative comparisons across methods.

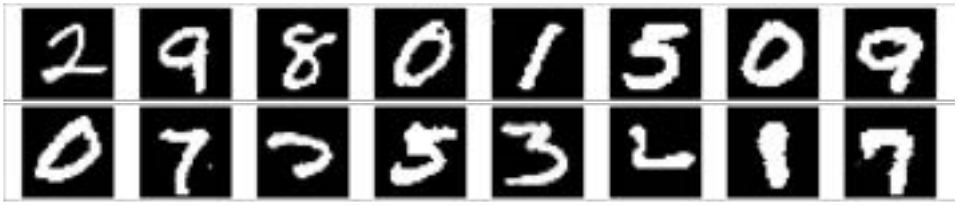
Among existing constrained generation methods, ECI and PCFM first generate samples and then apply a post-hoc projection to enforce the constraint. Consequently, these methods achieve nearly exact constraint satisfaction, as indicated by their extremely small constraint errors. However, their distribution-matching performance deteriorates substantially, suggesting that strict post-hoc constraint enforcement can cause the generated samples to deviate from the distribution learned by the flow model. This behavior is also evident in the qualitative results. In particular, their final projection adjusts the generated samples to satisfy the target average intensity without explicitly preserving the visual structure of the original samples. Since the projection only enforces the global intensity constraint, it can significantly alter the appearance of the image.

D-Flow achieves the strongest distributional performance among the baselines. However, it incurs substantial computational overhead because its optimization requires retaining the computational graph throughout the sampling ODE from the latent noise to the generated sample in order to compute the required gradients. This makes D-Flow considerably more memory-intensive and time-consuming than methods that do not require backpropagation through the entire sampling trajectory.

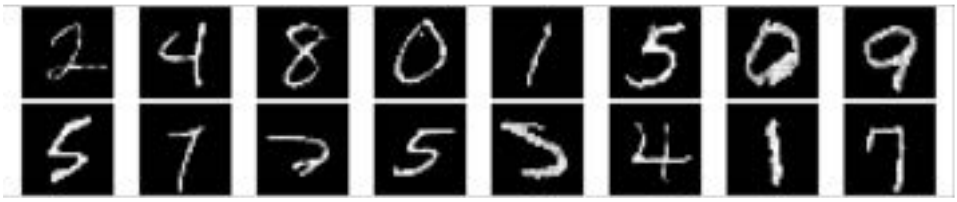
In contrast, MintFlow provides a better balance between constraint satisfaction and distributional fidelity. Although its constraint error is slightly higher than those of PCFM and ECI, it achieves substantially better FID, KID, MMD, and W_2 scores, indicating closer agreement with the conditional distribution induced by the pretrained flow model. MintFlow is also faster than the iterative correction methods, D-Flow and PCFM, despite requiring Jacobian-vector products in its adjoint ODE formulation.

For MNIST, average image intensity is closely related to digit stroke thickness. As shown in Figure 7, varying the target intensity allows MintFlow to produce thicker or thinner strokes while preserving the overall structure of the digits. This demonstrates that the intensity constraint provides direct and interpretable control over a salient visual property. Unlike projection-based methods, which seek a constraint-satisfying point by modifying the final generated sample, MintFlow applies intermediate perturbations along the flow trajectory. Consequently, the corrected samples are not explicitly required to remain close to their unconstrained counterparts, allowing the model to reach the target constraint while better preserving the learned data distribution.

(a) Unconstrained



(b) Higher intensity constraint



(c) Lower intensity constraint

Figure 7: Controlling MNIST digit stroke thickness with MintFlow using average intensity constraints. Starting from unconstrained samples, our method modifies the generated digits according to different target intensity constraints, producing thicker or thinner strokes while preserving distribution.

Data Fidelity Constraint. To evaluate whether our method can improve the fidelity of generated samples, we consider a learned reliability constraint based on a AE. The AE is trained to reconstruct original MNIST images through the encoder-decoder pipeline. Therefore, for a well-trained AE, the reconstruction error in (26) provides a measure of how closely a generated sample lies on the learned data manifold. In this experiment, satisfying the AE constraint encourages generated samples to preserve characteristics of the original MNIST distribution. In this experiment, enforcing the AE constraint is intended to encourage generated samples to remain close to the original MNIST distribution. Accordingly, we use the original MNIST test set as the ground-truth distribution and compute distributional metrics between the generated samples and the ground-truth MNIST samples.

Quantitative results are summarized in Table 7 together with qualitative results in Figure 9. Compared with unconstrained sampling, our correction method consistently improves all distribution matching metrics. This demonstrates that the proposed learned constraint can guide the pretrained flow model toward samples with improved fidelity while remaining close to the original flow-induced distribution. Unlike approaches that improve sample quality through additional training or model refinement, our framework provides a complementary inference-time strategy that enhances pretrained flow matching models without modifying the underlying model parameters.

In contrast, existing constrained generation methods show limited performance under this learned nonlinear constraint. Projection-based correction methods struggle to effectively satisfy the AE-based reconstruction constraint, resulting in even larger constraint errors than unconstrained sampling. In particular, ECI's projection-based approach, which is designed to enforce hard constraints, struggles to handle such nonlinear learned constraints effectively. D-Flow incurs additional computational overhead because its optimization requires backpropagating gradients through both the sampling process and the pretrained AE. DiffusionPDE achieves the strongest constraint satisfaction among the baselines, but its overall distributional and constraint metrics remain worse than those of our method, while requiring substantially more memory.

Meanwhile, our method successfully reduces the reconstruction error while achieving the best distribution-matching performance, demonstrating its ability to incorporate nonlinear learned constraints while preserving the underlying flow distribution. Although our method also requires differentiation through the AE encoder and decoder, it maintains a relatively low memory footprint and remains faster than D-Flow and PCFM. These results demonstrate that our approach can effectively incorporate learned nonlinear constraints without substantially compromising either distributional fidelity or computational efficiency.

Overall, this experiment demonstrates two important aspects of our framework: (1) its versatility in incorporating learned reliability constraints to improve generated sample quality, and (2) its ability to extend constrained generation beyond manually designed operators. We acknowledge that the effectiveness of such learned constraints naturally depends on the quality and generalization capability of the auxiliary model used to define them. Therefore, the purpose of this experiment is not to introduce the AE reconstruction constraint as a universally optimal fidelity metric, but rather to demonstrate the potential of our framework to flexibly integrate learned nonlinear constraints during inference and improve pretrained flow matching models without additional training.

Table 7: Quantitative results on the AE-based fidelity constrained MNIST dataset.

	Distribution Metrics				Constraint	Efficiency	
Method	FID ↓	KID ↓	MMD ↓	W_2 ↓	H ↓	Time (s) ↓	Memory (GB) ↓
Unconstrained	0.261	0.0089	0.0093	0.0093	4.54E-3	4.45	0.12
DiffusionPDE	0.242	0.0074	0.0078	0.0069	3.59E-3	6.98	5.15
D-Flow	0.257	0.0087	0.0092	0.5076	3.91E-3	1674.73	63.76
ECI	0.932	0.0079	0.0164	0.9659	3.43E-2	11.91	0.40
PCFM	10.225	0.3210	0.3336	3.1977	1.02E-1	79.56	10.17
Ours	0.221	0.0030	0.0042	0.0048	2.67E-3	18.06	0.76

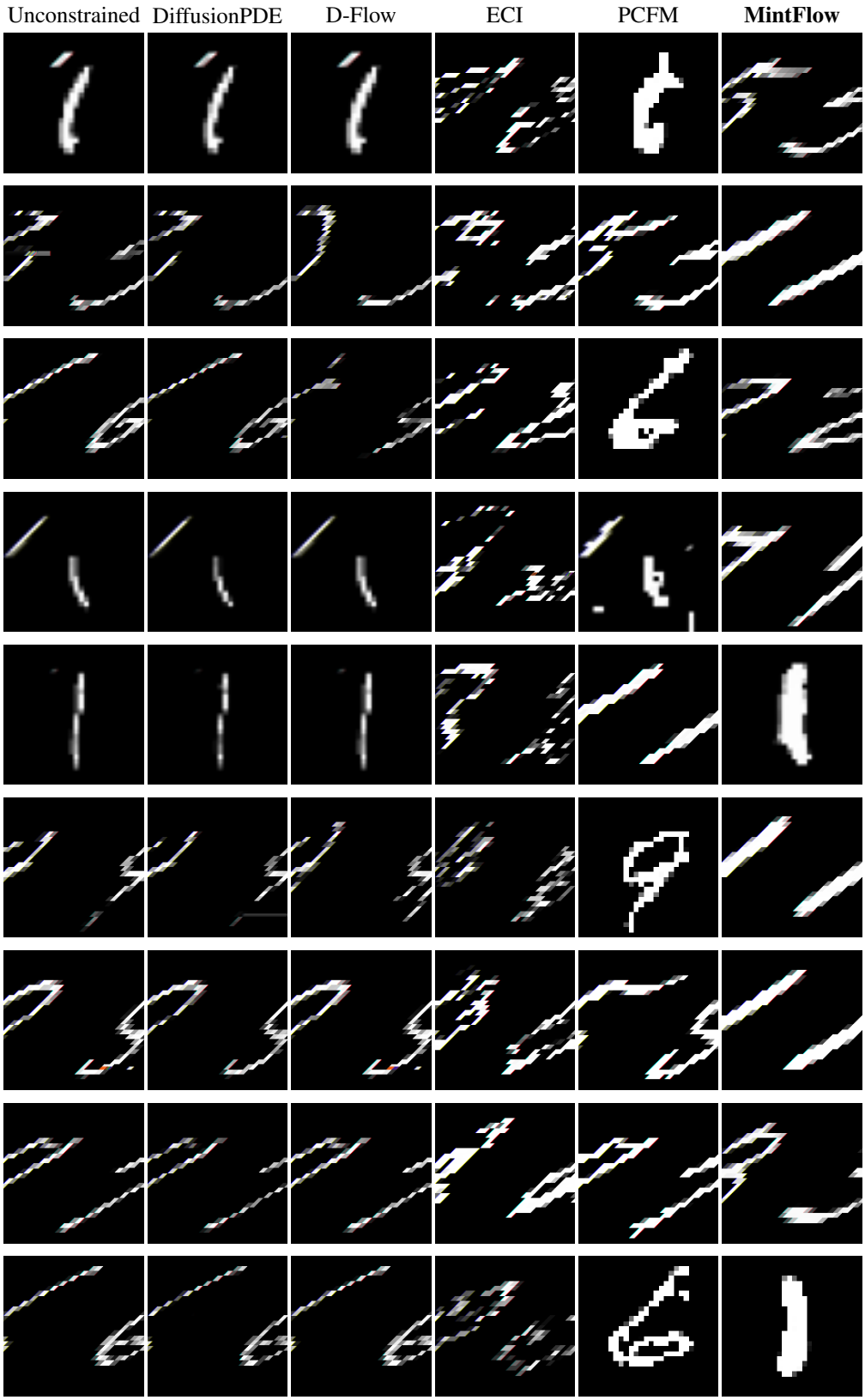


Figure 8: Uncurated results for intensity constrained generation on MNIST. We compare unconstrained samples with constrained samples generated by different methods. All samples in each row are generated from the same initial noise across methods.

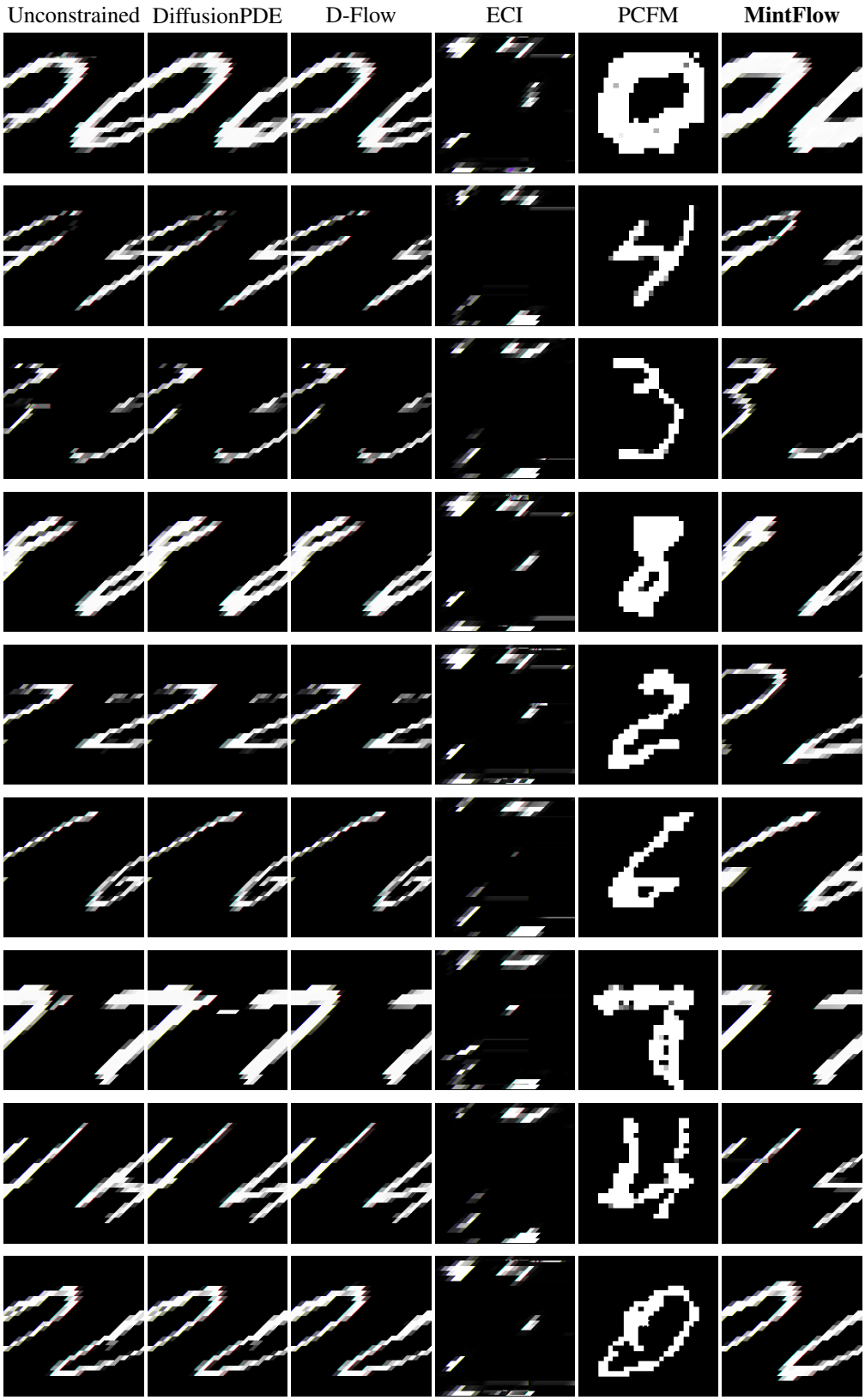


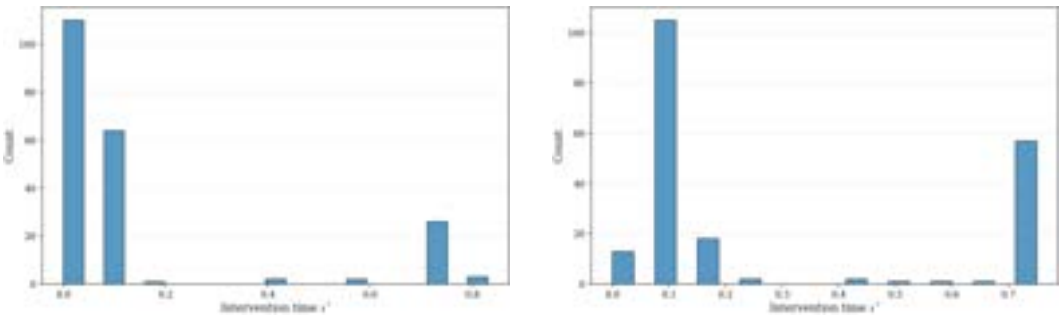
Figure 9: Uncurated results for AE-based fidelity-constrained generation on MNIST. We compare unconstrained samples with constrained samples generated by different methods. All samples in each row are generated from the same initial noise across methods.

Selected Intervention Times. To provide empirical evidence on the intervention times selected by our adaptive criterion in Section 3.3, we visualize the distribution of s^* obtained from (8). Specifically, for each generated sample, we evaluate the objective $J(s)$ over the predefined candidate intervention times and select the minimizing time s^*. Figure 10 shows the resulting histograms for two constraints on MNIST: the intensity constraint and the fidelity constraint.

The selected intervention times exhibit distinct patterns across the two constraints. For the linear intensity constraint, the selected s^* values are concentrated more heavily toward the earlier part of the trajectory. In contrast, the nonlinear fidelity constraint, defined using a trained autoencoder, produces a more dispersed distribution of intervention times, with selections spread across a wider range of the trajectory.

This difference can be understood from the structure of the two constraint functions. The intensity constraint is linear in the terminal state, resulting in a relatively simple and consistent constraint geometry. Consequently, the sensitivity of the constraint to an intermediate perturbation tends to vary more systematically along the trajectory, and the objective in (8) frequently favors earlier intervention times. The fidelity constraint, on the other hand, is induced by a nonlinear trained AE and therefore has a more complex local geometry. The corresponding constraint sensitivity can vary substantially across both samples. As a result, there is no single region of the trajectory that consistently minimizes the trade-off encoded by $J(s)$, leading to a broader distribution of selected intervention times.

Overall, these results suggest that the proposed time-selection criterion is not merely selecting a fixed or universally preferred intervention point. Instead, s^* adapts to the structure of the constraint and to the individual generated trajectory. The contrast between the two constraints provides empirical evidence that the adaptive intervention mechanism responds to the geometry and sensitivity of the constraint rather than imposing a predetermined intervention schedule.



(a) Intensity constraint (b) Fidelity constraint

Figure 10: Distribution of the selected intervention time s^* on MNIST. Each histogram shows the intervention times selected by (8) across generated samples for the corresponding constraint. **(a)** The linear intensity constraint tends to favor earlier intervention times, whereas **(b)** the nonlinear fidelity constraint defined by a trained autoencoder results in a more dispersed distribution of selected intervention times.

D.4.2 CIFAR10

Color Tone Constraint. We evaluate our method on CIFAR-10 using global color-tone constraints. We consider multiple target color tones and generate a total of 10,000 samples for quantitative evaluation. The corresponding quantitative results are reported in Table 8, while qualitative comparisons are shown in Figure 11. Overall, we observe trends similar to those in the MNIST experiments. Both quantitative and qualitative results demonstrate that our method achieves a favorable balance between constraint satisfaction and distribution preservation. Existing methods that rely on explicit projection steps tend to introduce visible artifacts when enforcing the color-tone constraint, leading to degradation in image fidelity. This becomes particularly noticeable for CIFAR-10, where the higher image resolution and more complex visual structure make direct post-hoc modifications more likely to affect semantic and local image features. D-Flow also incurs substantially greater computational overhead than MNIST as the image resolution increases, making it consider-

ably more expensive in this setting. DiffusionPDE achieves relatively strong distributional performance, but exhibits larger constraint errors. In particular, the qualitative results suggest that it does not sufficiently adjust the generated images toward the target color tone, consistent with its relatively large constraint error.

In contrast, our method consistently provides a better trade-off between constraint satisfaction and distributional fidelity. It effectively shifts generated samples toward the desired color-tone appearance while preserving their semantic content and overall visual quality. Furthermore, we demonstrate the flexibility of our approach by applying a diverse set of color-style constraints, showing that the proposed framework can provide controllable changes in global appearance without substantially disrupting the underlying image distribution.

Table 8: Quantitative results on the color-style constrained CIFAR10 dataset.

	Distribution Metrics				Constraint	Efficiency	
Method	FID ↓	KID ↓	MMD ↓	W_2 ↓	H ↓	Time (s) ↓	Memory (GB) ↓
Unconstrained	11.1568	0.0421	0.0518	5.354	1.02E-1	14.17	0.44
DiffusionPDE	8.4464	0.0014	0.0012	2.9062	4.66E-2	12.08	2.25
D-Flow	9.8484	0.0019	0.0015	3.138	5.789E-4	787.30	101.01
ECI	322.500	0.3596	0.29951	17.9582	2.53E-16	19.81	0.34
PCFM	125.262	0.0950	0.0727	11.1920	1.72E-12	27.10	4.26
Ours	5.08	0.0017	0.0014	2.663	2.08E-5	22.67	2.31

Data Fidelity Constraint. Similar to the MNIST experiment, we train a AE on CIFAR10 and use its reconstruction consistency as a learned fidelity constraint to improve the quality of generated samples. Quantitative results and qualitative results are summarized in Table 9 and Figure 12, respectively. Consistent with the MNIST results, applying our constraint correction improves the distribution matching metrics compared with unconstrained flow matching samples, indicating that the proposed framework can guide the pretrained model toward higher-fidelity samples without additional training. In contrast, existing constrained generation methods show limited performance under this nonlinear learned constraint. It is worth noting that the purpose of this experiment is not to optimize unconditional CIFAR10 generation performance, but to evaluate whether our framework can effectively incorporate learned constraints while preserving the distribution of a pretrained flow model.

Table 9: Quantitative results on the AE-based fidelity constrained CIFAR10 dataset.

	Distribution Metrics				Constraint	Efficiency	
Method	FID ↓	KID ↓	MMD ↓	W_2 ↓	H ↓	Time (s) ↓	Memory (GB) ↓
Unconstrained	11.797	0.0053	0.0043	3.4347	2.18E-5	13.91	0.72
DiffusionPDE	10.779	0.0058	0.0042	3.3389	1.06E-5	14.82	2.59
D-Flow	9.984	0.0057	0.0042	3.3438	9.09E-6	1618.61	103.74
ECI	310.353	0.3391	0.3083	17.6227	3.17E-4	30.80	0.68
PCFM	116.953	0.0729	0.0578	10.8794	6.78E-4	49.53	4.60
Ours	9.795	0.0040	0.0033	3.2832	1.04E-5	19.70	2.74

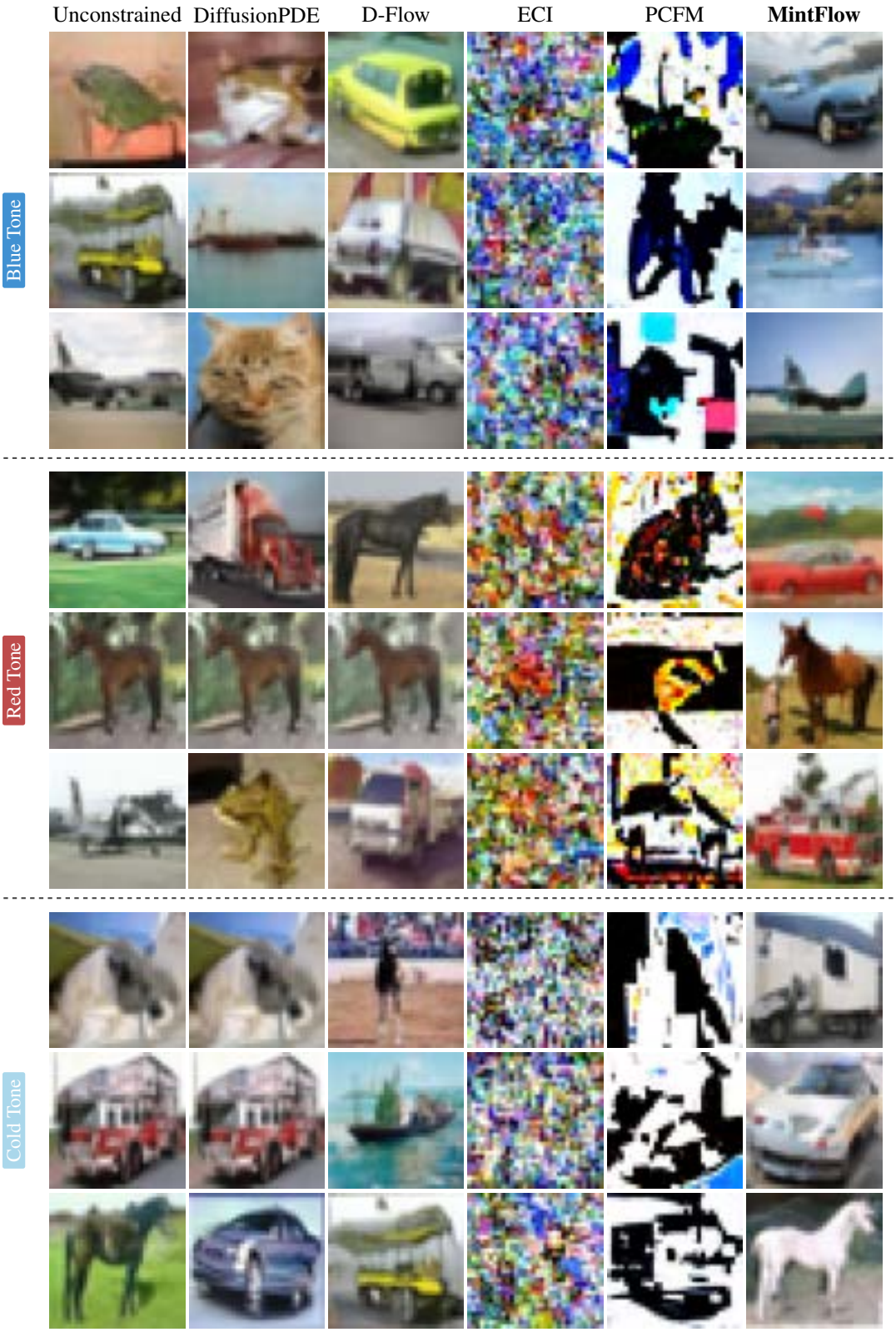


Figure 11: Qualitative comparison on CIFAR-10 under different color constraints. We compare unconstrained samples with constrained samples generated by different methods. All samples in each row are generated from the same initial noise across methods.

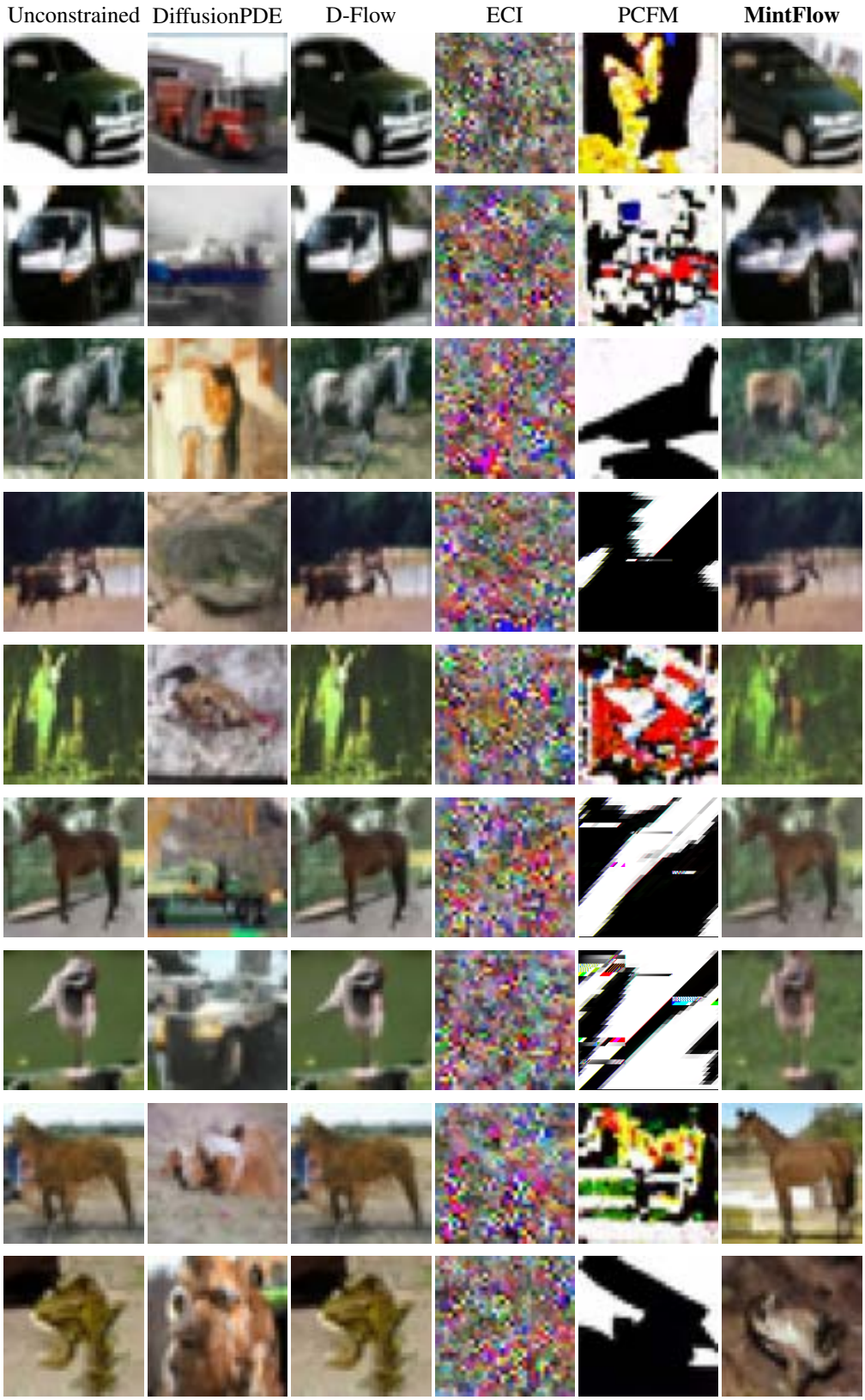


Figure 12: Uncurated results for AE-based fidelity-constrained generation on CIFAR10. We compare unconstrained samples with constrained samples generated by different methods. All samples in each row are generated from the same initial noise across methods.

E Application 2: Image Inverse Problems

This appendix provides the benchmark definitions, model and sampling configurations, evaluation protocol, and additional diagnostics supporting the image inverse problem results in Section 4.1. In particular, we separately quantify three aspects of reconstruction quality:

- **Constraint satisfaction:** how well the reconstructed image satisfies the measurement constraint associated with each inverse problem;
- **Pairwise reconstruction fidelity:** how closely the reconstruction matches the corresponding reference image;
- **Distributional fidelity:** how well the distribution of reconstructed images is preserved relative to the reference image distribution.

E.1 Datasets and Constraints

We evaluate the image inverse problems on two standard image datasets, **AFHQ-Cat** (Choi et al., 2020) and **FFHQ** (Karras et al., 2019). All images are resized to 64×64 before applying the inverse operators.

For each reference image u^{ref}, we construct a measurement from the corresponding forward operator and solve for a reconstruction u that satisfies the resulting measurement constraint.

Inpainting. For inpainting, we consider a centered 20×20 box mask M. The measurement consists of the pixels outside the masked region. The constraint is defined as

$$H(u) = M \odot \left(u - u^{\text{ref}}\right),$$

where $\odot$ denotes element-wise multiplication and M denotes the binary mask selecting the constrained region. Thus, satisfying the constraint requires the reconstruction to agree with the reference image on the observed pixels, while the missing 20×20 region is inferred from the learned image prior.

Super-resolution. For super-resolution, we first downsample the 64×64 reference image by a factor of 4, resulting in a 16×16 observation. Let D denote the downsampling operator, $D : \mathbb{R}^{3\times64\times64} \to \mathbb{R}^{3\times16\times16}$. The measurement constraint is

$$H(u) = D\left(u - u^{\text{ref}}\right).$$

Therefore, a valid reconstruction must reproduce the observed 16×16 image after downsampling, while the high-resolution details are determined by the learned image prior

Deblurring. For deblurring, we apply a Gaussian blur operator B with kernel size 11 and standard deviation $\sigma = 1.0$. The corresponding constraint is

$$H(x) = B\left(u - u^{\text{ref}}\right).$$

The inverse problem therefore consists of recovering a sharp image whose blurred version agrees with the observed measurement.

E.2 Flow-Matching Model and Baselines

Following (Patel et al., 2025), we conduct all experiments using the same pretrained Rectified Flow++ models (Lee et al., 2024). RFM models are trained to make the flow matching trajectories as close to straight lines as possible, resulting in simpler and more direct generative flows. Owing to this property, pretrained rectified flow models have been widely adopted in recent approaches for solving inverse problems with flow-based generative models. For all methods, we use 10-step Rectified Flow sampling to ensure a consistent generation budget across methods.

We compare our method against several existing approaches for solving inverse problems with flow-based generative models. We first consider **FlowChef** (Patel et al., 2025) and **D-Flow** (Ben-Hamu

et al., 2024), which enforce constraints by optimizing either an intermediate state along the generative trajectory or the initial Gaussian noise, respectively. We additionally compare against **PnP-Flow** (Martin et al., 2025b), a plug-and-play approach that incorporates the measurement operator into the flow-based sampling process. Additionally, we include two diffusion-based baselines, DPS (Chung et al., 2023a) and FreeDoM (Yu et al., 2023), using the Rectified Flow-compatible implementations developed in (Patel et al., 2025). For all baselines, we use the implementations provided in the official FlowChef repository[4] without modification. We also use the hyperparameters specified in (Patel et al., 2025) for each baseline. Our method is implemented for RFM following the formulation described in Appendix A.4. Unlike our previous formulation developed for general Flow Matching models, our RFM-based implementation exploits the linearity of the RFM flow path. This property allows us to obtain the optimal correction without explicitly solving the adjoint equation backward in time, as described in (15).

E.3 Evaluation Protocol

For AFHQ-Cat, we use all images in the official test split. For FFHQ, we randomly select $10{,}000$ images from the dataset using a fixed random seed and use the same set of reference images for all inverse-problem evaluations. For each dataset and each inverse-problem task, we evaluate the reconstructions from four complementary perspectives.

Constraint Satisfaction. For each inverse-problem task, we evaluate how well the reconstructed images satisfy the corresponding measurement constraint. Specifically, given a reconstruction $\hat{u}$, we compute the mean squared error of the constraint residual $H(\hat{u})$ over the entire evaluation set. A lower value indicates better satisfaction of the imposed constraint.

Pairwise Reconstruction Fidelity. We measure the reconstruction fidelity by comparing each restored image $\hat{u}$ with its corresponding reference image u^{ref}, i.e., the image before applying the degradation operator. We report PSNR, SSIM, and LPIPS to evaluate pixel-level, structural, and perceptual similarity, respectively. Higher PSNR and SSIM and lower LPIPS indicate better reconstruction fidelity.

Distributional Fidelity. We further evaluate whether the restored images preserve the overall image distribution. To this end, we compute the Fréchet Inception Distance (FID) between the set of restored images and the ground-truth images of AFHQ-Cat and FFHQ. A lower FID indicates that the distribution of the restored images is closer to that of the GT distribution.

Efficiency. Runtime is measured as synchronized wall-clock time per generated sample, and memory is reported as peak allocated GPU memory.

E.4 Additional Qualitative Results

Further qualitative results on the AFHQ-Cat dataset for inpainting, super-resolution, and deblurring are presented in Figures 13, 14, and 15, respectively. Corresponding results on the FFHQ dataset are provided in Figures 16, 17, and 18, respectively.

[4] https://github.com/FlowChef/flowchef

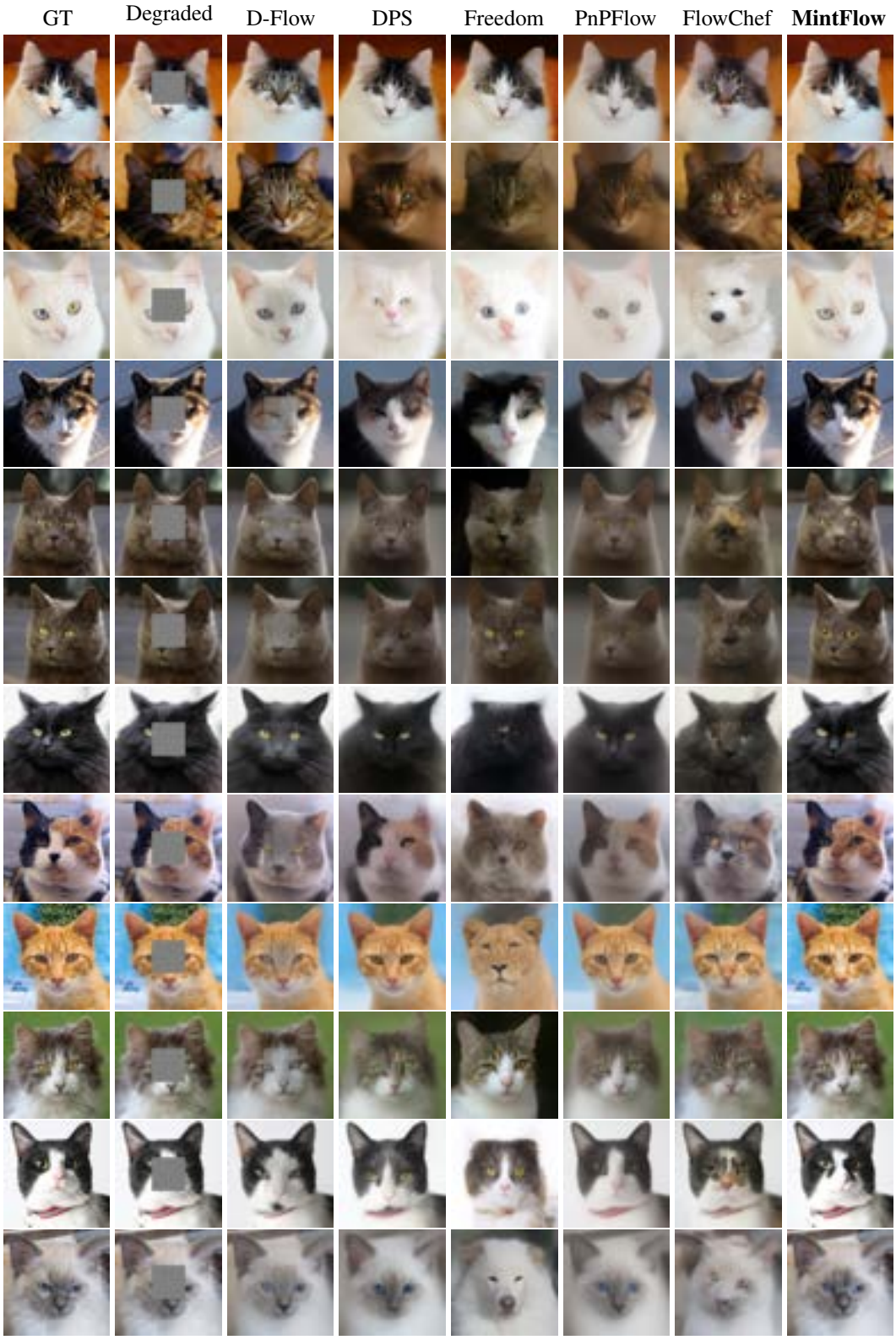


Figure 13: Uncurated inpainting results on the AFHQ-Cat dataset. The figure compares ground-truth images (GT), degraded inputs, and corresponding inpainting results from different methods.

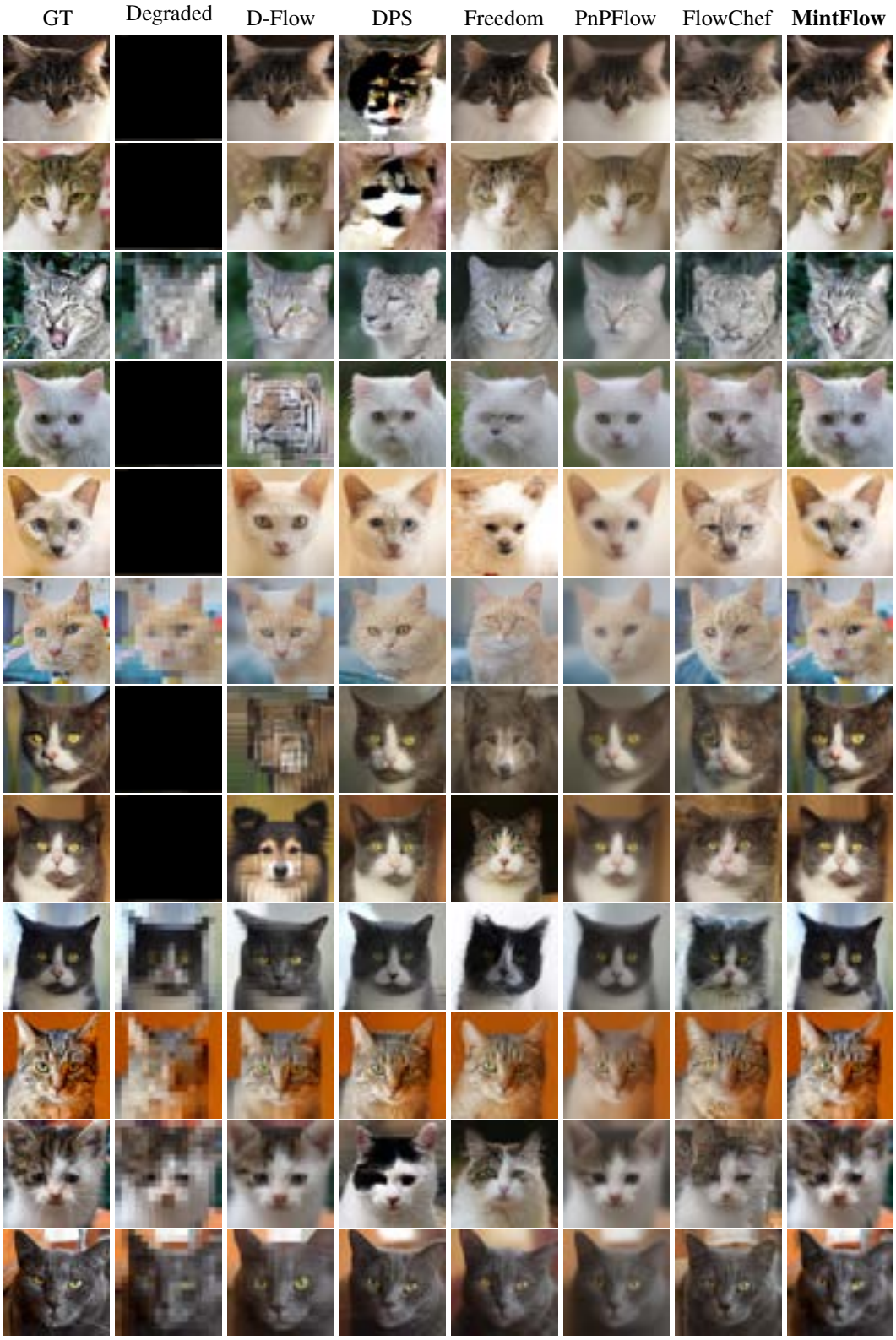


Figure 14: Uncurated super resolution (SR) results on the AFHQ-Cat dataset. The figure compares ground-truth images (GT), degraded inputs, and corresponding SR results from different methods.

Figure 15: Uncurated deblurring results on the AFHQ-Cat dataset. The figure compares ground-truth images (GT), degraded inputs, and corresponding deblurred results from different methods.

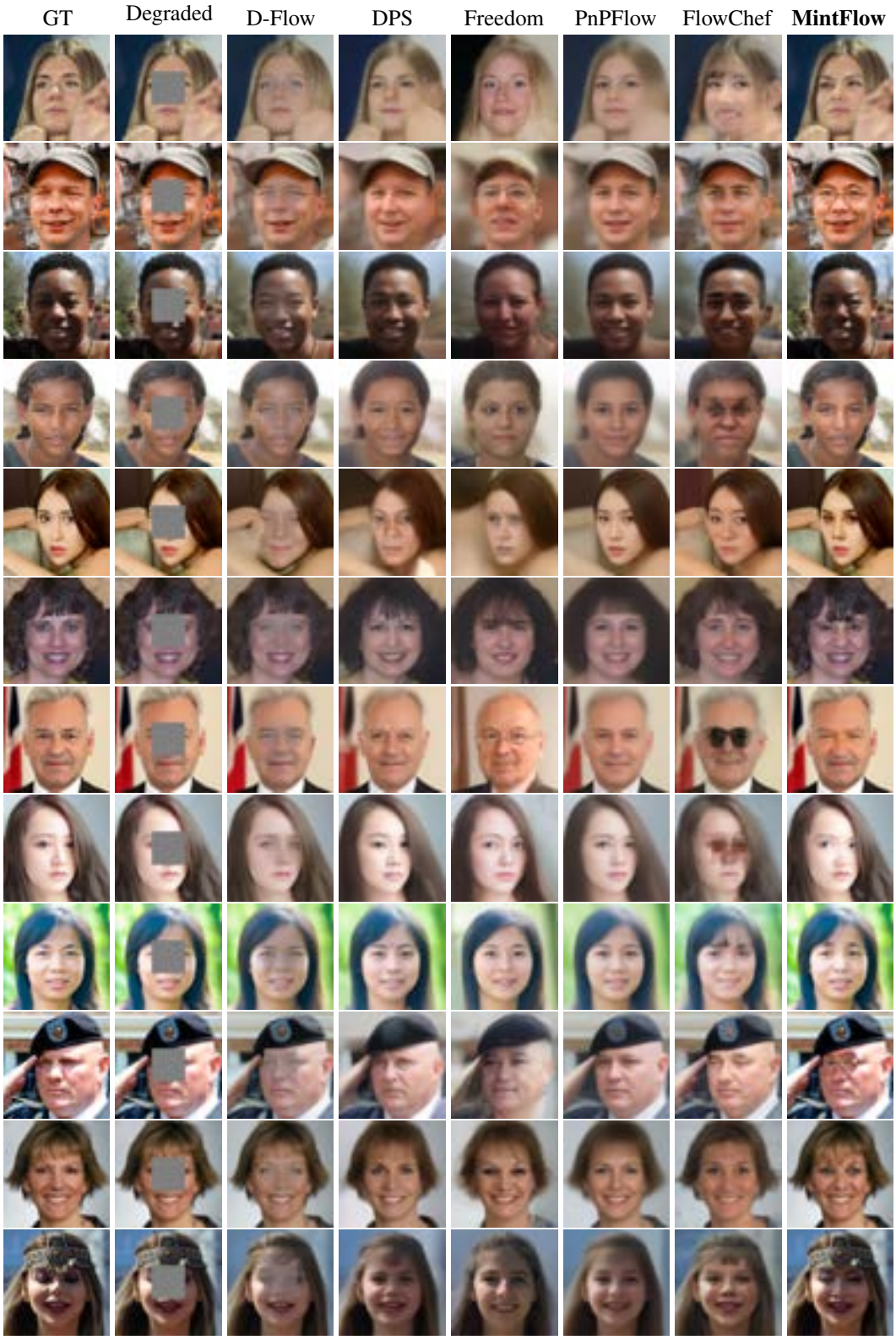


Figure 16: Uncurated inpainting results on the FFHQ dataset. The figure compares ground-truth images (GT), degraded inputs, and corresponding inpainting results from different methods.

Figure 17: Uncurated super resolution (SR) results on the FFHQ dataset. The figure compares ground-truth images (GT), degraded inputs, and corresponding SR results from different methods.

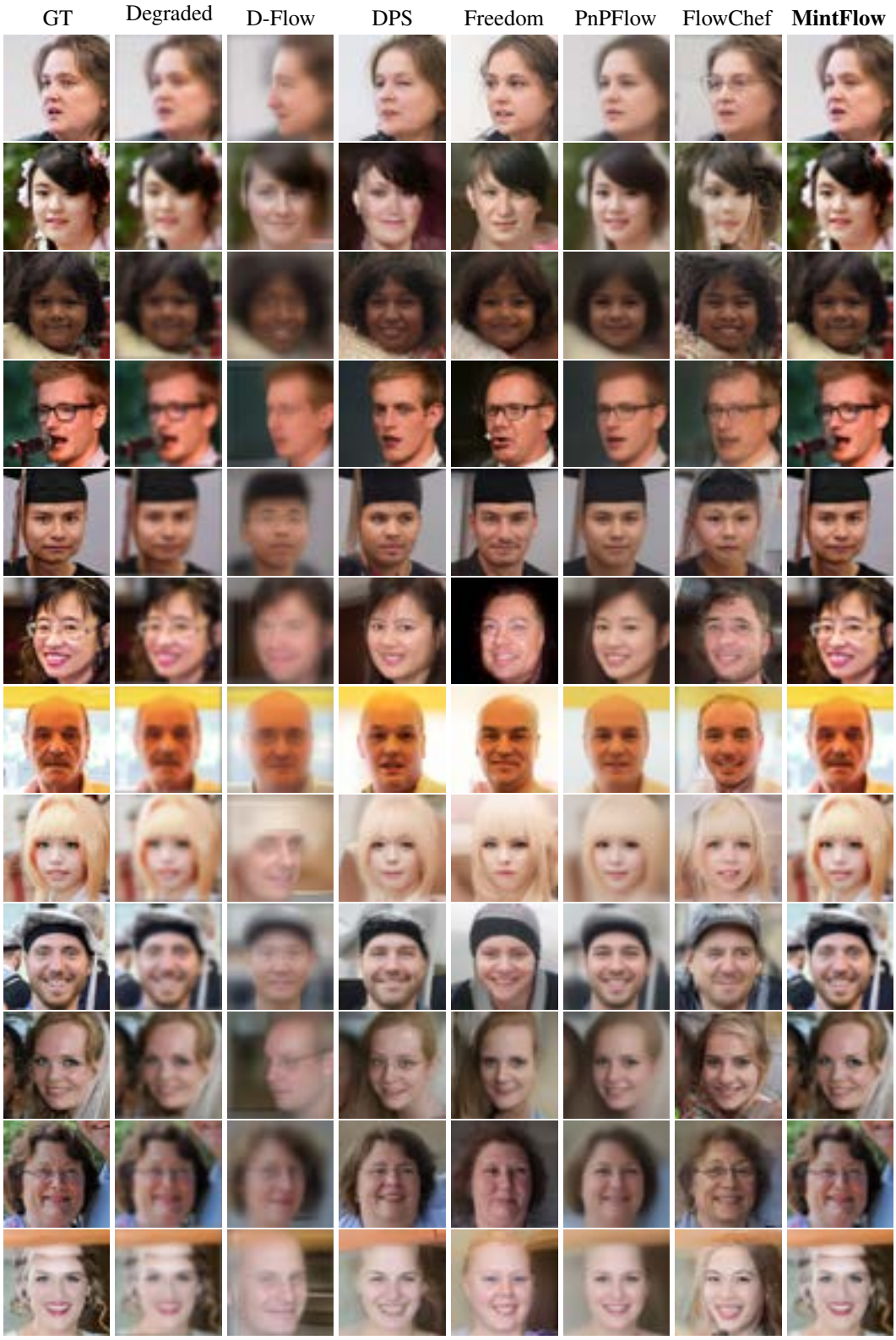


Figure 18: Uncurated deblurring results on the FFHQ dataset. The figure compares ground-truth images (GT), degraded inputs, and corresponding deblurred results from different methods.

F Application 3: Image Editing

This appendix provides the benchmark definitions, model and sampling configurations, evaluation protocol, and additional diagnostics supporting the image editing results in Section 4.2. In particular, we quantify not only how well the edited images satisfy the given constraints and the computational efficiency of each method, but also how well they fulfill the task-specific text guidance.

- **Constraint satisfaction:** how well the edited image satisfies the constraint, i.e., preserving the regions outside the provided mask;
- **Prompt adherence:** how well the edited image follows the given text prompt, measured using CLIP score;
- **Image similarity:** how visually consistent the edited image remains with the reference image, measured using CLIP score;
- **Efficiency:** the computational efficiency of each method in terms of inference speed and memory usage.

F.1 Datasets and Constraints

We perform large-scale image editing experiments on PIE-Bench (Ju et al., 2024). Specifically, we use the images from the `0_random_140` subset available in the official PIE-Bench data repository, along with the provided ground-truth masks. We use the same masks to control all methods for a fair comparison. The editing task is subject to an inpainting-style constraint: regions outside the provided mask should remain identical to the corresponding regions in the reference image. Unlike the aforementioned inverse problem, however, the edited region is additionally required to follow the given text guidance, making it challenging to satisfy both the preservation constraint and the desired semantic edit. Thus, the key challenge of this task is to simultaneously preserve the unmasked regions faithfully while performing the intended text-guided edit within the masked region.

F.2 Flow-Matching Model and Baselines

Following (Patel et al., 2025), we use InstaFlow (Liu et al., 2023c), a Stable Diffusion v1.5 variant, as the flow-matching model. We evaluate the same set of baselines used for the aforementioned inverse problems in Appendix E on InstaFlow. As observed in the previous experiments, D-Flow requires differentiating through the entire forward flow ODE integration, resulting in substantially higher memory consumption. For the image editing task on InstaFlow, this computational overhead exceeds the available memory in our experimental setup. We therefore exclude D-Flow from this experiment.

F.3 Evaluation Protocol

We evaluate the editing results using the following metrics.

Constraint Satisfaction. We measure how faithfully the edited image preserves the reference image outside the provided mask. Specifically, we compute the mean squared error (MSE) between the edited and reference images over the unmasked regions.

Prompt Adherence. We measure how well the edited image follows the given text prompt using CLIP-based similarity (Radford et al., 2021). Specifically, we use the OpenAI-pretrained CLIP ViT-B/32 model implemented in OpenCLIP [5]. We report **CLIP-T**, computed as the cosine similarity between the CLIP image embedding of the edited image and the CLIP text embedding of the corresponding prompt. Higher scores indicate better text-image alignment.

Image Similarity. We measure the visual similarity between the edited image and the reference image using **CLIP-I**. Specifically, CLIP-I is computed as the cosine similarity between their CLIP image embeddings using the same OpenAI-pretrained CLIP ViT-B/32 model. Higher scores indicate greater visual similarity to the reference image.

[5] https://github.com/mlfoundations/open_clip?utm_source=chatgpt.com

Efficiency. We measure the computational cost of constrained sampling in terms of inference time and peak GPU memory allocation. Inference time measures the wall-clock time required to generate an edited image, while peak memory allocation measures the maximum GPU memory allocated during constrained sampling.

F.4 Additional Qualitative Results

Additional qualitative results for image editing are presented in Figures 19 and 20. Overall, we observe that MintFlow successfully modifies the image according to the given prompt while faithfully preserving the original input image. In contrast, baseline methods tend to overly focus on the prompt instructions, resulting in the loss of the input background, artifacts, or blurry outputs. These results demonstrate that our method effectively maintains the image distribution to generate natural in-distribution images, while successfully satisfying constraints to naturally edit the image according to the prompt while retaining the input context.

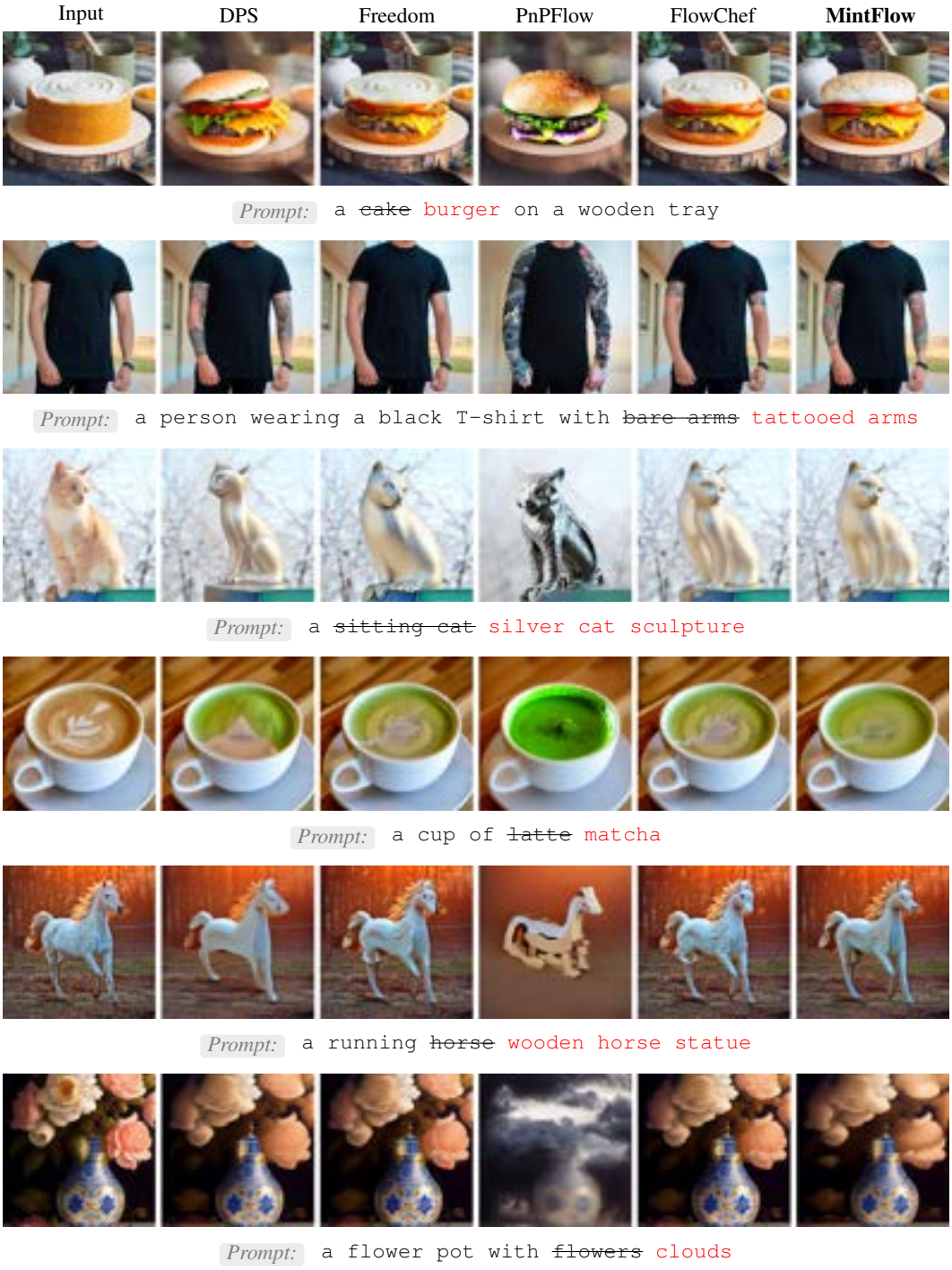


Figure 19: Additional qualitative comparison of image editing results. Given an input image and its corresponding text prompt, we compare the editing results produced by different methods. Strikethrough and red text indicate the original and edited text, respectively.

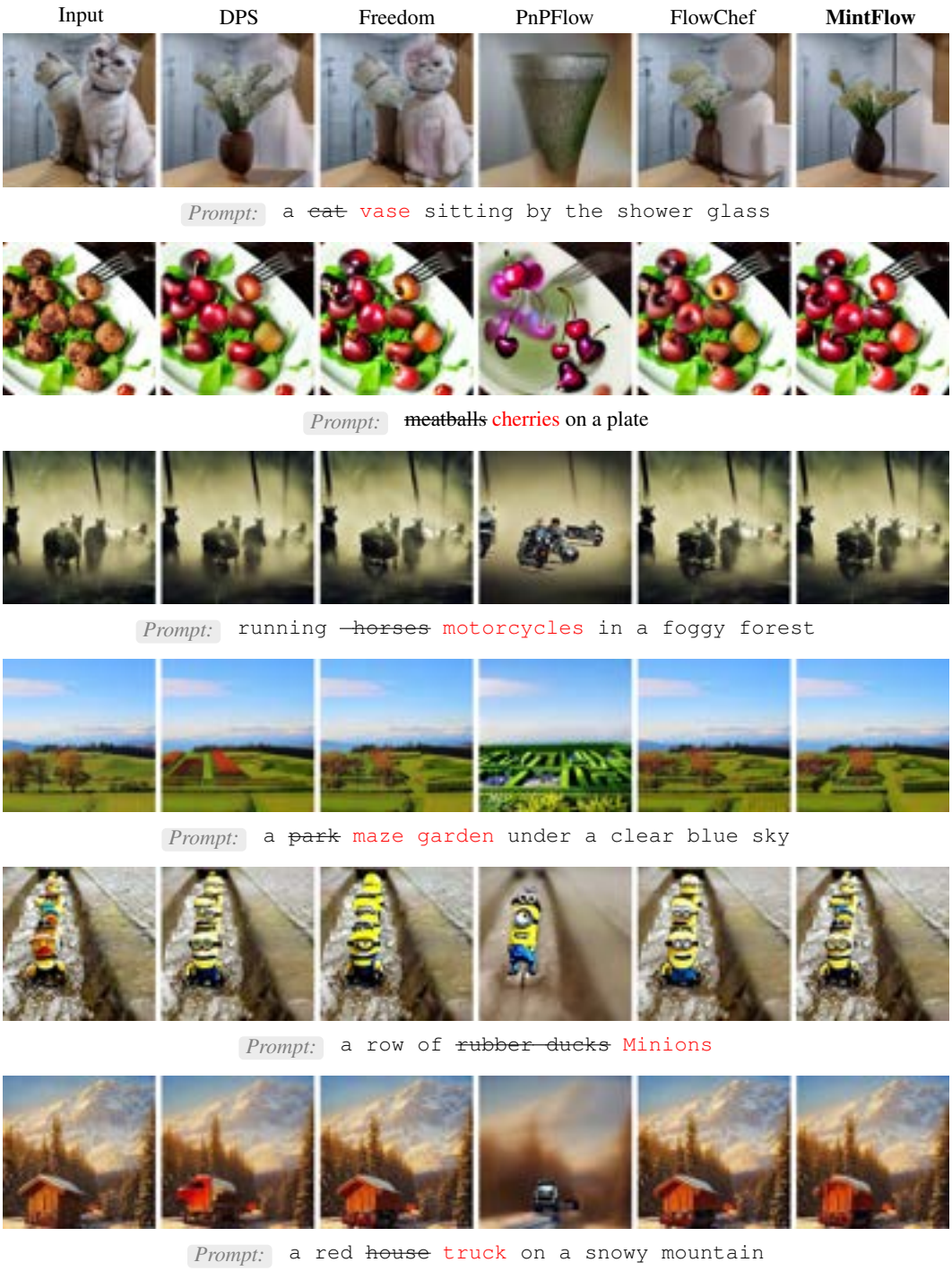


Figure 20: Additional qualitative comparison of image editing results. Given an input image and its corresponding text prompt, we compare the editing results produced by different methods. Strikethrough and red text indicate the original and edited text, respectively.

G Application 4: Physical System Modeling

This appendix provides the benchmark definitions, model and sampling configurations, evaluation protocol, and additional diagnostics supporting the physical-system results in Section 4.3. In particular, we separately quantify

- satisfaction of the joint conservation, initial-condition, and boundary-condition operator;
- set-level agreement with a paired numerical-solution distribution;
- local PDE consistency and agreement with method-specific numerical rollouts; and
- sensitivity to integration resolution and candidate-time selection.

G.1 Physical Systems

To distinguish physical time from the generative-flow time t used in Section 3, we denote physical PDE time by τ throughout this appendix.

Heat Equation. We consider

$$\partial_\tau u = \nu\, \partial_{xx} u, \qquad x \in [0, 2\pi), \qquad \tau \in [0, 1]. \tag{27}$$

The analytic solution family used for data generation is

$$u(x, \tau) = \sin(x + \phi) \exp(-\nu\tau), \qquad \nu \sim \mathcal{U}[1, 5], \qquad \phi \sim \mathcal{U}[0, \pi]. \tag{28}$$

The training loader draws 5,000 trajectories on demand from this family; the configured test split contains 1,000 additional draws. Fields are represented as $(n_x, n_\tau) = (100, 100)$ arrays on an endpoint-excluded periodic spatial grid and an endpoint-inclusive time grid. These analytic training trajectories are distinct from the benchmark reference pool defined below, which solves the PDE from FFM-derived initial conditions.

Reaction-Diffusion Equation. The RD1D benchmark follows

$$\partial_\tau u = \nu\, \partial_{xx} u + \rho u(1 - u), \qquad \nu = 5 \times 10^{-3}, \qquad \rho = 10^{-2}, \tag{29}$$

on

$$x \in [0, 1], \qquad \tau \in [0, 0.99]. \tag{30}$$

Fields are represented on a 128×100 space–time grid. The spatial coordinates are cell centers of the physical domain $[0, 1]$, and the boundary condition is specified by constant left and right Neumann fluxes $g_L \in [0, 0.05]$ and $g_R \in [-0.05, 0]$. The persisted training set is the Cartesian product of 80 initial conditions and 80 flux pairs (6,400 trajectories); the test set uses $16 \times 16 = 256$ trajectories. Both were generated by the finite-volume solver described below. For each benchmark source-noise ID, the IC and fluxes used as sampling targets are derived once from the corresponding Vanilla trajectory and then held fixed across all methods.

G.2 Physical Constraint

Our physical benchmark jointly imposes conservation, initial-condition, and boundary-condition consistency. We write the constraint operator from Section 3 as

$$H(u) = \begin{bmatrix} h_{\text{cons}}(u) \\ h_{\text{IC}}(u) \\ h_{\text{BC}}(u) \end{bmatrix}. \tag{31}$$

MintFlow applies the trajectory intervention of (6) to this joint operator rather than correcting the three components independently. The benchmark reports the sample-level statistic

$$\|H(u)\|_\infty = \max_j |H_j(u)|, \tag{32}$$

after the task-specific normalizations described below.

Initial condition. Let $u_{\mathrm{IC}}(x)$ denote the prescribed initial field associated with the aligned benchmark sample. The IC residual is

$$h_{\mathrm{IC}}(u) = u(\cdot, 0) - u_{\mathrm{IC}}. \tag{33}$$

The target u_{IC} is copied from the aligned Vanilla sample at $\tau = 0$. Residuals are evaluated in the raw, unnormalized field; component scales are applied only when constructing the combined statistic in (32).

Heat conservation. Define the discrete spatial mass

$$M(\tau_k) = \Delta x \sum_{i=1}^{n_x} u(x_i, \tau_k), \qquad \Delta x = \frac{2\pi}{100}. \tag{34}$$

The conservation residual is

$$h_{\mathrm{cons},k}(u) = M(\tau_k) - M(\tau_0), \qquad k = 1, \ldots, n_\tau - 1. \tag{35}$$

This is the periodic rectangle rule on the endpoint-excluded grid. It compares every generated time slice with the generated $\tau = 0$ slice; the separate IC block anchors that slice to u_{IC}. The Heat operator therefore has 99 mass residuals and 100 IC residuals, for a total dimension of 199.

Heat boundary condition. Periodicity is enforced structurally by the Heat representation and therefore does not introduce an independent residual:

$$h_{\mathrm{BC}}(u) \equiv 0. \tag{36}$$

Consequently, the Heat statistic is determined by conservation and IC consistency.

RD1D global balance. Let

$$M(\tau) = \int_\Omega u(x, \tau)\, dx. \tag{37}$$

The global-balance residual at time τ_k is

$$\begin{aligned} h_{\mathrm{cons},k}(u) &= M(\tau_k) - M(\tau_0) \\ &\quad - \int_0^{\tau_k} \left[\rho \int_\Omega u(x,\tau)(1 - u(x,\tau))\, dx + g_L(\tau) - g_R(\tau) \right] d\tau. \end{aligned} \tag{38}$$

Spatial integration uses the cell-centered rectangle rule and temporal accumulation uses the trapezoidal rule. The fluxes in (38) are the prescribed external fluxes, not values re-estimated from the generated field. The nonlinear reaction term makes this component of H nonlinear in u.

RD1D boundary condition. Let $\widehat{g}_L(u, \tau)$ and $\widehat{g}_R(u, \tau)$ denote the configured five-point boundary-flux observables evaluated on the stored cell-centered field. The BC residual is

$$h_{\mathrm{BC}}(u) = \begin{bmatrix} \widehat{g}_L(u,\tau) - g_L(\tau) \\ \widehat{g}_R(u,\tau) - g_R(\tau) \end{bmatrix}_\tau. \tag{39}$$

The five-point one-sided stencil is evaluated for positive-time slices only; the classical corner compatibility condition at $\tau = 0$ is not imposed. The benchmark normalizes the two flux blocks by their physical range, 0.05, before constructing the joint operator:

$$H_{\mathrm{RD1D}}(u) = \begin{bmatrix} h_{\mathrm{cons}}(u) \\ h_{\mathrm{IC}}(u) \\ h_{\mathrm{BC}}(u)/0.05 \end{bmatrix}. \tag{40}$$

The global-balance and IC blocks have unit scale. Thus the RD1D operator has 99 global-balance, 128 IC, and 2×99 BC residuals, for a total dimension of 425. The boundary observable used by the constraint is defined on stored cell centers; it is not identical to the external face-flux variable used internally by the finite-volume reference solver.

G.3 Flow-Matching Model and Baselines

Each PDE has a separately pretrained checkpoint with the same architecture: a four-layer Fourier Neural Operator with 32 Fourier modes per dimension, 64 hidden channels, 256-channel lifting and readout layers, and a 32-dimensional sinusoidal embedding of generative-flow time. The two grid coordinates and time embedding are concatenated with the state, yielding 35 input channels and one output channel. Direct instantiation of the current model gives 17,884,993 trainable parameters.

The generative model is trained for 20,000 optimization iterations using conditional flow matching. Given a data sample u and source sample z, the interpolant and target velocity are

$$u_s = (1-s)z + su, \qquad v^\star = u - z, \qquad s \sim \mathcal{U}[0,1]. \tag{41}$$

The source z is drawn from a Matérn-$\frac{1}{2}$ Gaussian process with length scale 10^{-3} and unit output scale on the normalized two-dimensional grid. Training uses Adam with batch size 256, learning rate 3×10^{-4}, and gradient-norm clipping at 100. Both retained checkpoints are from iteration 20,000.

We compare Vanilla sampling against PCFM, ECI, DiffusionPDE, D-Flow, and MintFlow. All constrained methods act only during generation and introduce no learned parameters. Within an equation, every method uses the same checkpoint, source-noise tensor, and sample-indexed constraint target. The paper reports each approach as a single method; implementation-level MintFlow choices are examined separately in the ablation study.

Vanilla uses adaptive Dopri5 integration with 100 requested evaluation points for Heat and 200 for RD1D. All fixed-step samplers—PCFM, ECI, DiffusionPDE, D-Flow, and MintFlow—use 100 Euler steps on Heat and 200 on RD1D. Thus the integration resolution is aligned across methods within each physical system. DiffusionPDE uses guidance strength 1 on Heat and 10^{-3} on RD1D. D-Flow performs 20 L-BFGS iterations with learning rate 1. Standard ECI uses one mixing step, and standard PCFM uses 10 Newton updates per Heat flow step and one per RD1D flow step.

The MintFlow configuration reported in Tables 3 and 10 uses 100 Euler steps on Heat and 200 on RD1D, with $K = 10$ candidate times in $[0.1, 0.98]$, the end-biased warp $s = s_{\min} + (s_{\max} - s_{\min})[1-(1-q)^2]$, $\lambda = 10^{-2}$, and correction scale $\gamma = 1$. These details are provided for reproducibility and do not denote separate methods in the main comparison.

G.4 Evaluation Protocol

Each benchmark contains $N = 1000$ samples per method. Heat uses source-noise seed 20260811, and RD1D uses seed 20260820. Reported means and standard deviations therefore summarize sample-to-sample variation in one fixed-seed run; they are not uncertainty estimates across training runs or sampling seeds.

Target bank and numerical reference. For each equation, Vanilla is first evaluated on the shared source-noise bank. Its initial slices define the 1,000 IC targets. For Heat, a diffusivity in $[1, 5]$ is fitted once from each complete Vanilla trajectory; for RD1D, constant Neumann fluxes are fitted once and clipped to the training ranges. These latent quantities are held fixed across methods at a given sample index. The numerical reference pool then solves each of these 1,000 physical problems: Heat uses the periodic Fourier semigroup and RD1D uses the data generator's exact-reaction, explicit-midpoint finite-volume scheme. Thus the reference is neither the unconditional training distribution nor an unconstrained FFM pool. It is a solver distribution conditioned on the same finite bank of FFM-derived ICs, BCs, and latent physical parameters used by the constrained samplers. Set-level metrics ignore index pairing after constructing the two sample sets.

Constraint satisfaction. For sample i, $R_\infty(u_i) = \|H(u_i)\|_\infty$ is the largest absolute entry after the RD1D BC normalization above. The main table reports its sample mean, while Table 10 additionally reports the sample standard deviation.

Set-level distribution metrics. We avoid image-specific feature extractors by fitting one frozen, quadrature-weighted, mean-centered proper orthogonal decomposition (POD) to the 1,000 numerical reference trajectories. The transform weights each flattened coordinate by the square root of its

tensor-product physical quadrature weight, subtracts the weighted reference mean, and projects onto the leading right singular vectors. No whitening, nonlinear feature map, clipping, or method-specific refitting is applied. With a 99.9% variance threshold and at most 20 components, the retained dimensions are 5 for Heat (99.9463% variance) and 14 for RD1D (99.9429% variance).

Let a_i and b_j denote generated and reference POD coefficients. All distances are evaluated in float64. FID is the Gaussian Fréchet distance

$$\|\mu_a - \mu_b\|_2^2 + \mathrm{tr}\Big(C_a + C_b - 2(C_a^{1/2} C_b C_a^{1/2})^{1/2}\Big). \tag{42}$$

It is an FID analogue in POD coordinates, not Inception-feature FID. KID is the unbiased two-sample MMD^2 estimator with $k(a,b) = (a^\top b/d + 1)^3$; finite-sample estimates may be slightly negative. MMD is the square root of the nonnegative biased MMD^2 using the average of five RBF kernels. Their bandwidths are $\{0.25, 0.5, 1, 2, 4\}$ times the median pairwise distance fitted once on the reference coefficients. Finally, because both sets have equal size, empirical W_2 is computed exactly as

$$W_2 = \left[\min_{\pi \in S_N} \frac{1}{N} \sum_{i=1}^{N} \|a_i - b_{\pi(i)}\|_2^2\right]^{1/2} \tag{43}$$

using the Hungarian assignment algorithm. These four quantities measure agreement with the conditioned numerical distribution. They do not directly estimate divergence from the unconstrained FFM distribution; Vanilla is reported as an evaluated comparator, not used as the metric reference.

Physical consistency. The PDE residual is the root mean square of a finite-difference operator in the raw field. Heat uses a second-order periodic spatial Laplacian and a second-order numerical temporal derivative; RD1D uses the corresponding second-order interior spatial stencil. For the simulation error, every method uses its own generated initial slice and the physical parameters and BCs fixed for that sample index:

$$E_{\mathrm{sim}}(u_i) = \frac{\|u_i - u_i^{\mathrm{solver}}\|_2}{\|u_i^{\mathrm{solver}}\|_2}. \tag{44}$$

This metric tests dynamical self-consistency and does not by itself test whether the generated IC matches its prescribed target. The main table reports mean simulation error. Table 10 additionally reports the finite-difference PDE residual and sample standard deviations for both physical-consistency metrics.

G.5 Full Quantitative Results

The expanded results reinforce the trade-off visible in the main table. MintFlow has the smallest point estimate for all four distribution metrics and both physical-consistency metrics on both systems, while ECI and PCFM obtain the smallest constraint residuals. Their stronger explicit constraint satisfaction is accompanied by substantially larger distribution distances, PDE residuals, and simulation errors. D-Flow is the strongest competing baseline on physical consistency, but its constraint residual remains much larger than those of MintFlow, ECI, and PCFM. The standard deviations further show that mean performance can conceal heterogeneous behavior: the standard deviation of MintFlow's Heat constraint residual exceeds its mean, while the RD1D PDE-residual standard deviations of ECI and PCFM exceed their corresponding means. By contrast, MintFlow's RD1D PDE residual and simulation error have comparatively small deviations. Finally, the negative KID values are valid finite-sample outcomes of the unbiased estimator and should be interpreted as estimates close to zero, not as negative distances. All distributional conclusions refer to the conditioned numerical reference rather than directly to the unconditional FFM distribution.

G.6 Additional Qualitative Results

Figures 22 and 23 provide fixed-index examples from the retained benchmark runs. For every method and sample, the upper row shows the generated space–time field. We initialize the numerical PDE solver with the initial condition extracted from that field and use the sample-indexed physical parameters and boundary conditions. The lower row reports the pointwise absolute discrepancy from this method-specific numerical trajectory, corresponding to (44) before spatial and temporal

Table 10: Physical System Modeling Benchmark under initial conditions, boundary conditions, and conservation constraints. Performance evaluation across $N = 1000$ generated samples on the Heat Equation (System 1) and Reaction-Diffusion Equation (System 2). The benchmark highlights the trade-off between *Constraint Satisfaction* and *Distribution Preservation*. While strict-constraint baselines (ECI, PCFM) achieve near-zero constraint residuals, they severely degrade distributional quality. In contrast, MintFlow achieves an optimal Pareto frontier, enforcing physical constraints while preserving the pre-trained data distribution. For all metrics, lower values (↓) indicate superior performance.

	Distribution Fidelity†				**Constraint**††	**Physical Consistency**†††	
Method	**FID** ↓	**KID** ↓	**MMD** ↓	$\mathbf{W_2}$ ↓	**H** ↓	**PDE Res.** ↓	**Sim. Error** ↓
System 1: Heat Equation							
Vanilla FFM	1.255E−3	−3.17E−4	2.496E−2	4.872E−2	5.16E−3±2.19E−3	1.103E0±5.08E−1	5.57E−2±6.15E−2
DiffusionPDE	4.712E−3	1.019E−3	5.361E−2	7.431E−2	1.86E−2±2.20E−2	1.044E0±6.98E−1	8.49E−2±2.95E−2
D-Flow	1.290E−3	−2.10E−4	2.554E−2	4.098E−2	4.71E−3±2.08E−3	7.82E−1±3.76E−1	4.34E−2±2.46E−2
ECI	1.865E−1	4.004E−2	2.067E−1	4.502E−1	2.27E−7±6.45E−8	1.554E0±8.78E−1	4.47E−1±2.86E−1
PCFM	1.851E−1	3.965E−2	2.062E−1	4.485E−1	1.30E−7±5.28E−8	1.549E0±8.72E−1	4.45E−1±2.85E−1
MintFlow	2.91E−5	−5.22E−4	5.44E−3	1.455E−2	3.57E−6±3.69E−5	6.77E−1±3.22E−1	1.23E−2±2.09E−2
System 2: Reaction-Diffusion Equation							
Vanilla FFM	1.74E−4	−8.11E−6	2.130E−2	2.078E−2	1.907E0±7.75E−1	2.98E−1±5.9E−2	3.55E−2±1.72E−2
DiffusionPDE	9.34E−4	1.43E−4	6.063E−2	3.712E−2	9.56E−1±2.50E−1	3.43E−1±7.3E−2	3.29E−2±2.05E−2
D-Flow	3.95E−4	1.45E−5	3.861E−2	4.085E−2	4.44E−1±1.40E−1	2.45E−1±4.4E−2	3.02E−2±2.03E−2
ECI	1.829E−2	2.59E−4	1.299E−1	1.647E−1	8.02E−6±2.72E−6	3.250E0±5.866E0	2.28E−1±2.87E−1
PCFM	1.831E−2	2.61E−4	1.301E−1	1.648E−1	8.06E−6±2.81E−6	3.255E0±5.871E0	2.28E−1±2.87E−1
MintFlow	1.09E−4	−1.47E−5	1.342E−2	1.870E−2	3.18E−4±2.16E−4	2.28E−1±3.0E−2	2.58E−2±1.40E−2

† Distribution fidelity evaluates closeness of generated samples to the numerical reference trajectory distribution in a quadrature-weighted, mean-centered POD space. Slightly negative KID values occur near zero due to unbiased polynomial sampling noise.

†† H denotes the combined normalized maximum absolute constraint residual (H) across conservation laws, initial conditions (IC), and boundary conditions (BC), reported as mean ± std. dev.. Heat periodicity is structural, while reaction-diffusion boundary fluxes are normalized by 0.05.

† † † PDE Res. denotes the finite-difference PDE operator residual. Sim. Error measures relative ℓ_2 trajectory error when integrating a numerical solver forward from each sample's generated initial condition.

aggregation. These plots separate explicit constraint satisfaction from dynamical self-consistency: a field can have a small value of $\|H(u)\|_\infty$ while deviating from the PDE solution implied by its own initial condition. ECI and PCFM show broad errors in several selected cases, whereas MintFlow is closer to its corresponding rollout. Because the panels contain selected samples and omit some evaluated variants, they are diagnostic rather than evidence of set-level dominance.

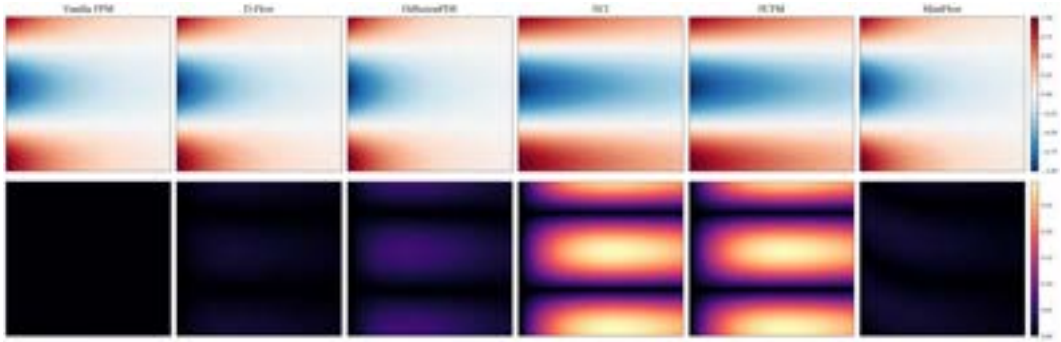


Figure 21: Additional Heat qualitative results. For each example, the first row shows the generated spacetime field for each method. The second row shows the pointwise absolute error to a numerical Heat trajectory initialized from that generated sample's own initial condition.

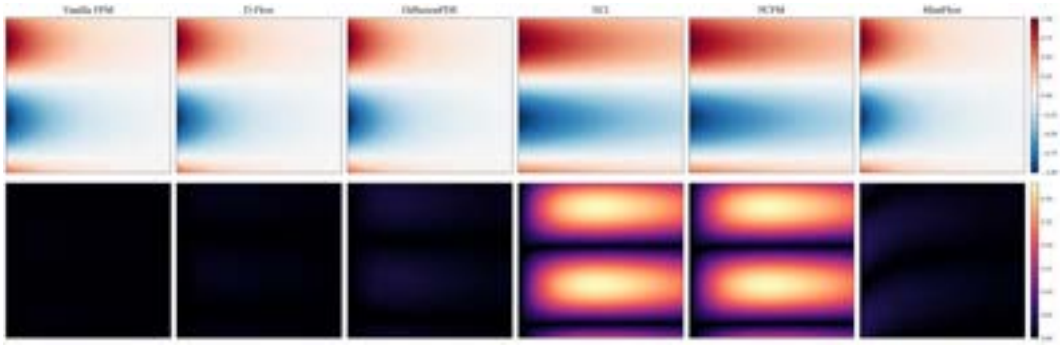


Figure 22: Additional Heat qualitative results.

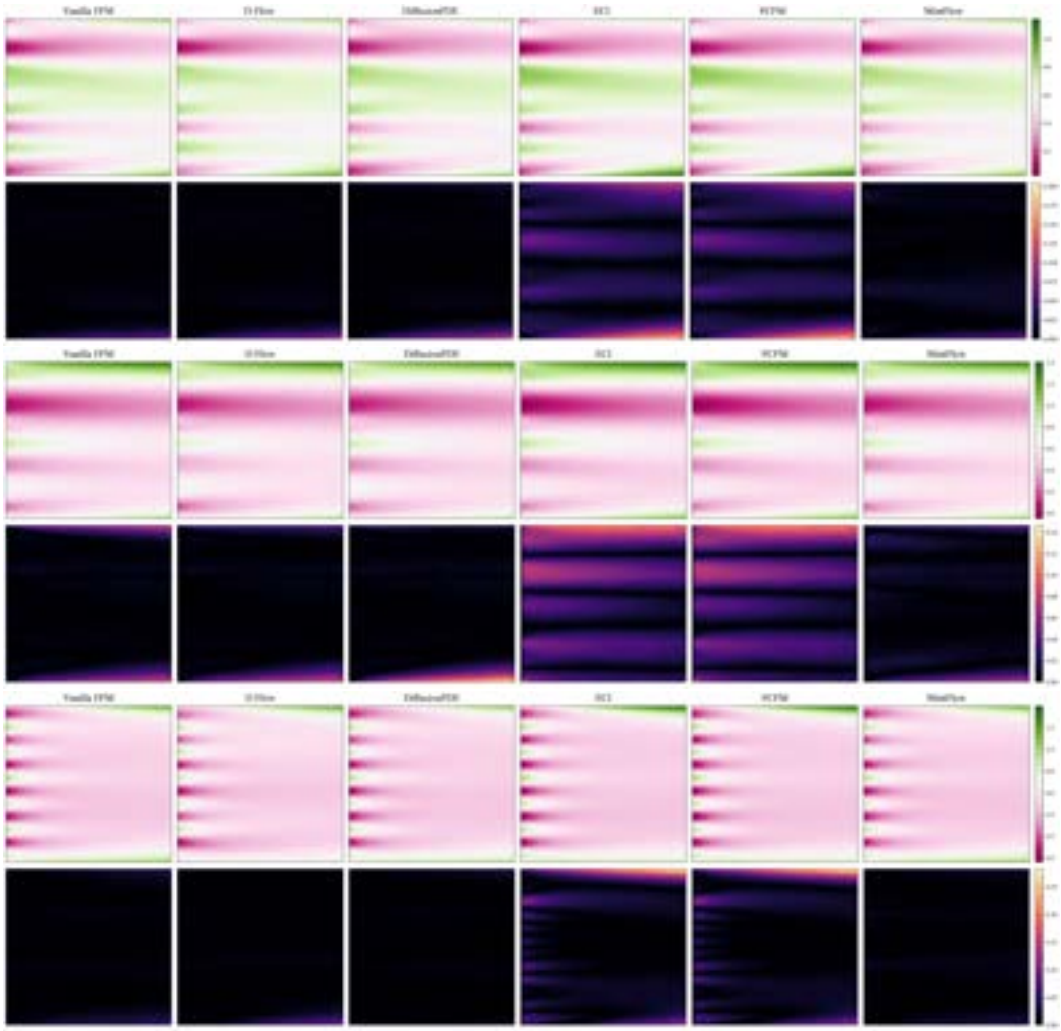


Figure 23: Additional RD1D qualitative results.

H Ablation Studies

H.1 Ablation on Integration Resolution

We ablate MintFlow on the Heat benchmark using the same $N = 1000$ source-noise bank, target bank, numerical reference pool, and frozen POD transform as in the main comparison. We vary only sampling integration resolution, holding all other sampler settings fixed. Consequently, differences across configurations cannot be attributed to resampling or refitting the representation.

As shown in Figure 6, Increasing the number of Euler steps of flow matching model from 50 to 100, 200, and 400 monotonically improves FID, the constraint residual, and same-IC simulation error. Importantly, the discretization of Euler steps affects not only the forward sampling integration error but also the numerical error in solving the backward adjoint equation. A finer discretization therefore yields both a sampling trajectory and a adjoint that more closely approximate their continuous-time counterparts. This leads to a more faithful preservation of the pretrained trajectory and, consequently, improved distributional fidelity, while also providing a more accurate local correction direction for enforcing the terminal constraint. Nevertheless, the framework remains relatively robust across different resolutions, with more noticeable degradation only when fewer than 100 sampling steps are used.

The finite-difference PDE residual improves through 200 steps but rises slightly at 400 steps. The headline Heat comparison nevertheless uses 100 steps to match the integration resolution across all fixed-step methods; the higher-resolution settings quantify numerical sensitivity rather than define the reported operating point. The headline absolute scores remain those in Tables 3 and 10; the ablation is used to assess relative sensitivity to individual design choices.

H.2 Ablation on Candidate Time Selection and optimal-time regularization.

We further ablate the regularization parameter λ in the objective function J in (8) for adaptive intervention time selection. We use the same data and experimental setup as in the preceding section, keeping the source-noise bank, target bank, numerical reference pool, and frozen POD transform fixed. We vary only λ, while holding all other sampler settings fixed. Consequently, differences across configurations cannot be attributed to resampling or refitting the representation.

Reults in Figure 6 shows that varying λ reveals a trade-off between constraint satisfaction and distribution preservation. Since λ does not directly regularize the intervention magnitude but instead determines the preferred intervention time s, a larger λ favors later interventions. When the intervention occurs very close to the terminal time, the correction increasingly resembles a direct perturbation of the terminal sample toward the constraint manifold, leaving less opportunity for the pretrained flow to absorb the intervention. Although such a correction can more strongly enforce the constraint, a large perturbation may move the sample farther from the original terminal distribution ν_T. In practice, the performance remains relatively stable for a moderate range of λ, indicating that the proposed objective provides a robust balance between constraint satisfaction and distributional preservation. The weak sensitivity at small λ indicates that the candidate grid, rather than the regularization term, governs the selected correction times in this regime.

H.3 Ablation on Intervention steps

We conduct an ablation study on the number of intervention steps to investigate how our method responds to repeated trajectory interventions. In image editing, the generated image must simultaneously follow the target prompt and satisfy the constraint, making a single intervention generally insufficient to achieve both objectives. Therefore, in all preceding experiments, we consistently use $N = 20$ intervention steps, following the setting of baseline methods (Patel et al., 2025). In this section, we vary the number of intervention steps N and examine its effect on the editing results.

As shown in Fig. 24, using too few intervention steps often produces images that are aligned with the target prompt but deviate substantially from the input image, indicating that a single or small number of interventions is insufficient to generate an image that satisfies the target prompt while preserving the input constraint. As N increases, the editing results progressively become more consistent with the input image while continuing to satisfy the target prompt. After a certain number of intervention

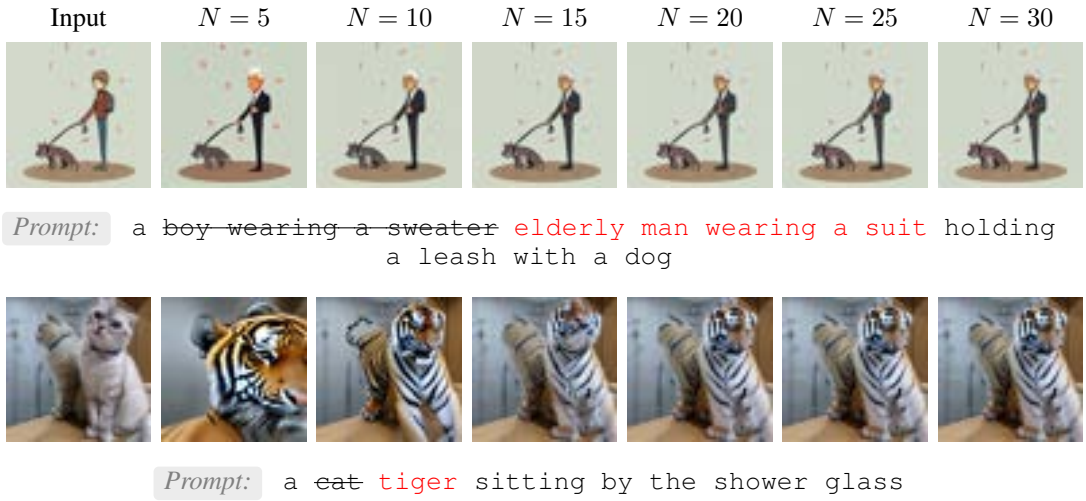


Figure 24: Comparison of image editing results with different numbers of interventions N. When N is too small, the generated images align with the target prompt but fail to satisfy the constraint, resulting in substantial deviations from the input image. As N increases, the method better preserves the background and overall appearance of the input image while naturally modifying only the masked regions to satisfy the target prompt. Beyond a certain N, the results remain largely unchanged, suggesting that a moderate number of interventions is sufficient, while even with more interventions, our method remains within the learned distribution, demonstrating its stability.

steps, the results converge and remain largely unchanged despite further increasing N, suggesting that a moderate number of interventions is sufficient to achieve the desired editing effect.

Importantly, even when a large number of interventions are applied, our method continues to generate plausible images without departing from the learned distribution. This indicates that the proposed minimal-intervention strategy does not introduce excessive perturbations that could push the generated samples out of the learned data distribution. Overall, these observations demonstrate the effectiveness and stability of our intervention strategy, while also providing practical guidance on the number of interventions required for image editing.