LYAPUFLOW: CONTROLLING GENERATIVE FLOWS WITH LYAPUNOV FEEDBACK FOR INVERSE PROBLEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Pretrained flow models are now widely used as generative priors in science and vision, where inference-time guidance enables test-time constraints without retraining. Existing methods use projection, posterior sampling, or iterative optimization of the generative trajectory. We propose LyapuFlow, an alternative based on Lyapunov feedback control. At each sampling step, LyapuFlow predicts the terminal sample towards which the current flow is evolving, and evaluates the constraint violation on this prediction. Then, we compute the minimum-norm control that satisfies a prescribed Lyapunov decrease condition. The resulting control remains inactive when the uncontrolled dynamics already reduce the constraint violation at the prescribed rate. Otherwise, it provides a corrective update within a flow-relative trust region that prevents the control from dominating the pretrained dynamics. We demonstrate LyapuFlow in both data and latent spaces, outperforming alternatives spanning different mechanisms for test-time constraint enforcement in scientific machine learning and image inverse problems.

1 INTRODUCTION

Flow matching has emerged as a flexible framework for modeling complex data distributions (Lipman et al., 2023). In many downstream applications, the quality of generated samples critically depends on whether they satisfy task-specific constraints. In scientific domains, for example, generated solutions to partial differential equations (PDEs) need to be consistent with hard physical requirements, including initial condition and conservation law. In image inverse problems, generated samples must reproduce measurements degraded by a forward operator. Conditional generative models can incorporate constraints available during training (Tong et al., 2024; Keegan & Ruthotto, 2026), but constraints specified at inference time motivate *test-time constrained generation*, where the sampling trajectory of a pretrained model is steered to satisfy the specified constraints.

Existing test-time constrained generation methods modify the sampling trajectory through iterative optimization or online guidance (Daras et al., 2024). Iterative trajectory optimization repeatedly integrates the generative dynamics to evaluate outcomes and refine the trajectory (Ben-Hamu et al., 2024; Liu et al., 2023; Wang et al., 2024), whereas online guidance applies instantaneous corrections during a single sampling rollout. Specifically, projection-based methods map intermediate states or predicted endpoints toward a constraint-defined feasible set (Christopher et al., 2024; Cheng et al., 2025; Utkarsh et al., 2025). These projections strongly enforce feasibility, but can overly prioritize constraint satisfaction over sample fidelity. Posterior guidance methods bias the trajectory toward samples with higher posterior probability of satisfying the constraints (Chung et al., 2023; Patel et al., 2025; Kim et al., 2025), typically relying on task- and model-dependent guidance strengths. Control-theoretic approaches for generative policies in robotics (Yang et al., 2025; Huang et al., 2025) construct feedback directly from constraints on the evolving state, leaving feedback based on explicit terminal predictions unexplored.

In this work, we propose an alternative guidance using Lyapunov feedback control (Khalil & Grizzle, 2002) on constraint satisfaction of predicted terminal samples. Given a pretrained flow $\boldsymbol{v}_\theta(\boldsymbol{x}_t, t)$ transporting $\boldsymbol{x}_0 \sim p_0$ to $\boldsymbol{x}_1 \sim p_1$, we augment its dynamics with an inference-time control $\boldsymbol{u}(\boldsymbol{x}_t, t)$:

$$\dot{\boldsymbol{x}}_t = \boldsymbol{v}_\theta(\boldsymbol{x}_t, t) + \boldsymbol{u}(\boldsymbol{x}_t, t), \qquad \text{with test-time constraint} \qquad \boldsymbol{h}(\boldsymbol{x}_1) = \boldsymbol{0}, \tag{1}$$

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

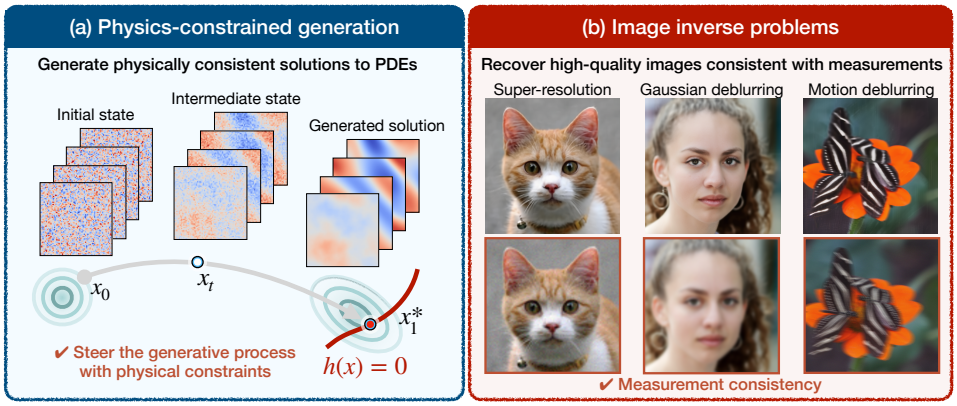

Figure 1: Test-time constraints for scientific and image inverse problems. (a) Physical constraints guide PDE solution generation. (b) Measurements guide image super-resolution and deblurring.

where p_0 and p_1 denote the base and target distributions. The difficulty is that the control acts at intermediate states $\boldsymbol{x}_t$, while the constraint is defined on the terminal sample $\boldsymbol{x}_1$. We propose LyapuFlow to address this by predicting the terminal sample toward which the current flow is evolving and evaluating the constraint on this prediction. The resulting constraint energy defines a terminal-aware Lyapunov function. At each integration step, LyapuFlow computes a closed-form minimum-norm feedback from a prescribed Lyapunov decrease condition. The feedback vanishes when the pretrained flow already satisfies this condition, allowing the sample to follow the original generative trajectory. Otherwise, it provides a corrective update within a flow-relative trust region that prevents excessive intervention, without iterative optimization. Our main contributions are as follows:

- We introduce LyapuFlow, a test-time feedback controller for pretrained flow models. LyapuFlow predicts the terminal sample from each intermediate state and computes a closed-form minimum-norm feedback from a prescribed Lyapunov decrease condition (see Section 3).
- We characterize how the continuous-time feedback behaves under finite-step integration and introduce a feedback trust region to limit overly large corrections (see Section 3.1). We further extend LyapuFlow to latent-space models by propagating constraint gradients through the decoder and optionally restoring the velocity-Jacobian contribution for finer-scale refinement (see Section 3.2).
- We evaluate LyapuFlow on physics-constrained generation in data space (see Section 4.1) and image inverse problems in latent space (see Section 4.2). Across both settings, LyapuFlow improves sample fidelity while maintaining competitive constraint satisfaction.

2 Background and Related Work

In this section, we review existing approaches for enforcing a general equality constraint $\boldsymbol{h}(\boldsymbol{x}) = 0$. As illustrated in Fig. 1, $\boldsymbol{h}$ varies across domains, ranging from physical constraints in PDEs to measurement-consistency constraints in image inverse problems. Below, we examine how existing methods for steering generative trajectories toward these constraints differ in their mechanisms.

Projection-based enforcement. Projection methods explicitly correct generated samples toward the constraint set (Hansen et al., 2023; Christopher et al., 2024). For a differentiable residual $\boldsymbol{h} : \mathbb{R}^d \to \mathbb{R}^m$, consider the local linearization of the constraint $\boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{J}_{\boldsymbol{h}}(\boldsymbol{x})\Delta\boldsymbol{x} = \boldsymbol{0}$, where $\boldsymbol{J}_{\boldsymbol{h}} = \partial\boldsymbol{h}/\partial\boldsymbol{x}$. Assuming $\boldsymbol{J}_{\boldsymbol{h}}$ has full row rank (typically with $d > m$), the minimum-norm correction is

$$\boldsymbol{x}^+ = \boldsymbol{x} - \boldsymbol{J}_{\boldsymbol{h}}(\boldsymbol{x})^\top \left[\boldsymbol{J}_{\boldsymbol{h}}(\boldsymbol{x})\boldsymbol{J}_{\boldsymbol{h}}(\boldsymbol{x})^\top\right]^{-1} \boldsymbol{h}(\boldsymbol{x}). \tag{2}$$

ECI sampling applies constraint corrections to an extrapolated endpoint using a problem-specific correction operator and interpolates the corrected endpoint with x_0 for the next sampling step (Cheng et al., 2025). PCFM extends projection-based correction to differentiable nonlinear equality constraints using Jacobian-based Gauss–Newton updates (Utkarsh et al., 2026; 2025). For nonlinear constraints, Eq. (2) requires solving an $m \times m$ linear system at each projection, which severely limits scalability.

Posterior and gradient-based guidance. Another method class guides generation using likelihood or task-objective gradients. For a measurement $\boldsymbol{y} \in \mathbb{R}^m$, posterior sampling targets

$$p(\boldsymbol{x}_1 \mid \boldsymbol{y}) \propto p_1(\boldsymbol{x}_1)p(\boldsymbol{y} \mid \boldsymbol{x}_1), \tag{3}$$

where p_1 denotes the pretrained generative distribution and $p(\boldsymbol{y} \mid \boldsymbol{x}_1)$ denotes the measurement likelihood. DPS approximates sampling from Eq. (3) by estimating the measurement-likelihood gradient to guide the reverse diffusion process (Chung et al., 2023). FlowChef applies efficient task-objective guidance to rectified flow models, while avoiding backpropagation through the velocity model (Patel et al., 2025). FlowDPS extends posterior guidance to flow models using measurement-likelihood gradients at estimated clean states, likewise bypassing the velocity-model Jacobian (Kim et al., 2025). ProFlow similarly refines terminal estimates to improve physical consistency in PDE problems (Yu et al., 2026). These methods steer the trajectory without explicit projection onto the constraint set, but suitable guidance scales can vary widely across tasks.

Control and trajectory optimization. Control theory provides another approach to modifying pretrained generative dynamics. FlowGrad and OC-Flow iteratively refine a complete sequence of control inputs to improve the terminal objective (Liu et al., 2023; Wang et al., 2024). OC-Flow formulates this refinement as an optimal control problem and solves it using Pontryagin's Maximum Principle (Pontryagin, 1987), balancing terminal loss and control cost:

$$\boldsymbol{u}^*_{[0,1]} \in \arg\min_{\boldsymbol{u}_{[0,1]}} \mathcal{L}(\boldsymbol{x}_1) + \lambda \int_0^1 \|\boldsymbol{u}_t\|^2 \, dt, \tag{4}$$

where $\lambda > 0$ weights the control cost. SAD-Flower and UniConFlow compute instantaneous $\boldsymbol{u}_t$ through quadratic programs based on Lyapunov and barrier conditions for robotic planning (Huang et al., 2025; Yang et al., 2025). These conditions measure constraint violations on partially denoised samples. Preserving fidelity to the generative prior remains a challenge.

3 LyapuFlow: Lyapunov Feedback for Constrained Generation

In this section, we develop LyapuFlow for the controlled generative dynamics introduced in Eq. (1). Given a pretrained velocity field $\boldsymbol{v}_\theta$ and a terminal constraint $\boldsymbol{h}(\boldsymbol{x}_1) = \boldsymbol{0}$, our goal is to construct an inference-time feedback control $\boldsymbol{u}(\boldsymbol{x}_t, t)$ that reduces terminal constraint violation while avoiding excessively large corrections to the generative dynamics. The main difficulty is that the control acts on intermediate states $\boldsymbol{x}_t$, whereas the constraint is defined only on the terminal sample $\boldsymbol{x}_1$. Figure 2 provides an overview of the LyapuFlow sampling, illustrating how terminal constraints are propagated to intermediate states while the feedback remains tied to the pretrained generative dynamics.

The control is applied at intermediate states $\boldsymbol{x}_t$, whereas the constraint is defined on the terminal sample $\boldsymbol{x}_1$. To obtain a quantity that can drive local feedback, we predict the terminal sample $\hat{\boldsymbol{x}}_{1|t}$ using a first-order extrapolation over the remaining integration interval

$$\hat{\boldsymbol{x}}_{1|t} = \boldsymbol{x}_t + (1-t)\boldsymbol{v}_\theta(\boldsymbol{x}_t, t). \tag{5}$$

To quantify violation of the terminal constraint, we use the quadratic residual loss $\mathcal{L}(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{h}(\boldsymbol{x})\|_2^2$, which is nonnegative and vanishes when the constraint is satisfied. Evaluating this loss at the predicted terminal sample defines

$$V(\boldsymbol{x}_t, t) = \mathcal{L}\left(\hat{\boldsymbol{x}}_{1|t}\right) = \frac{1}{2}\left\|\boldsymbol{h}\left(\hat{\boldsymbol{x}}_{1|t}\right)\right\|_2^2. \tag{6}$$

Thus, V can be interpreted as the constraint energy of the predicted terminal sample. It provides a terminal-aware quantity at each intermediate state, while $\hat{\boldsymbol{x}}_{1|t} = \boldsymbol{x}_1$ ensures that $V(\boldsymbol{x}_1, 1) = \mathcal{L}(\boldsymbol{x}_1)$ at the end of generation. We use V as a Lyapunov function to determine the feedback control. Along the controlled dynamics $\dot{\boldsymbol{x}}_t = \boldsymbol{v}_\theta(\boldsymbol{x}_t, t) + \boldsymbol{u}_t$, its total derivative is

$$\dot{V} = \partial_t V + \nabla_{\boldsymbol{x}_t} V^\top \boldsymbol{v}_\theta(\boldsymbol{x}_t, t) + \nabla_{\boldsymbol{x}_t} V^\top \boldsymbol{u}_t. \tag{7}$$

We choose the control so that the predicted terminal constraint violation decreases as

$$\dot{V} \le -\alpha V, \qquad \alpha \ge 0, \tag{8}$$

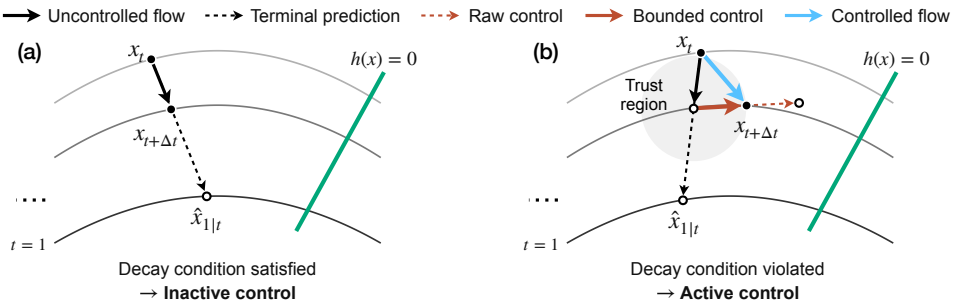

Figure 2: Illustration of LyapuFlow. At each integration step, LyapuFlow evaluates the pretrained velocity, predicts the terminal sample, computes the minimum-norm feedback from the Lyapunov decrease condition, bounds the feedback using the trust region, and advances the controlled flow. (a) When the pretrained dynamics already satisfy the Lyapunov decrease condition, no feedback is applied. (b) Otherwise, LyapuFlow applies the bounded control.

where α specifies the desired rate of decrease. Substituting the controlled dynamics into Eq. (8) gives a linear constraint on the instantaneous control

$$\nabla_{\boldsymbol{x}_t} V^\top \boldsymbol{u}_t \leq -\partial_t V - \nabla_{\boldsymbol{x}_t} V^\top \boldsymbol{v}_\theta(\boldsymbol{x}_t, t) - \alpha V. \tag{9}$$

Following the classical pointwise minimum-norm construction in Freeman & Kokotovic (2008), we select the smallest instantaneous correction that satisfies this condition

$$\text{minimize}_{\boldsymbol{u}_t} \quad \frac{1}{2}\|\boldsymbol{u}_t\|_2^2 \qquad \text{s.t.} \quad \nabla_{\boldsymbol{x}_t} V^\top \boldsymbol{u}_t + \partial_t V + \nabla_{\boldsymbol{x}_t} V^\top \boldsymbol{v}_\theta(\boldsymbol{x}_t, t) + \alpha V \leq 0. \tag{10}$$

The constraint in Eq. (10) defines a half-space in the control space, with minimum-norm solution (see Appendix A.1 for derivation).

$$\boldsymbol{u}_t^\star = -\eta_t \nabla_{\boldsymbol{x}_t} V, \qquad \eta_t = \frac{\left[\partial_t V + \nabla_{\boldsymbol{x}_t} V^\top \boldsymbol{v}_\theta(\boldsymbol{x}_t, t) + \alpha V\right]_+}{\|\nabla_{\boldsymbol{x}_t} V\|_2^2}. \tag{11}$$

The scalar η_t is an adaptive feedback gain determined by the current state of the generative flow. When the term inside $[\cdot]_+$ is nonpositive, the pretrained flow already reduces V at the requested rate and $\eta_t = 0$. Otherwise, η_t is the smallest gain along $-\nabla_{\boldsymbol{x}_t} V$ that satisfies the Lyapunov decrease condition. Unlike fixed-scale gradient guidance, the strength of the correction therefore adapts to the nominal flow, the current constraint violation, and the local gradient of V.

Flow-aware feedback. The gradient $\nabla_{\boldsymbol{x}_t} V$ depends on both the terminal constraint and the local sensitivity of the pretrained flow. Differentiating Eq. (6) through the terminal prediction gives

$$\nabla_{\boldsymbol{x}_t} V = \nabla_{\hat{\boldsymbol{x}}_{1|t}} \mathcal{L} + (1-t) \boldsymbol{J}_{\boldsymbol{v}}(\boldsymbol{x}_t, t)^\top \nabla_{\hat{\boldsymbol{x}}_{1|t}} \mathcal{L}, \tag{12}$$

where $\boldsymbol{J}_{\boldsymbol{v}}(\boldsymbol{x}_t, t) = \frac{\partial \boldsymbol{v}_\theta(\boldsymbol{x}_t, t)}{\partial \boldsymbol{x}_t}$. The first term on the right-hand side captures the direct effect of perturbing the current state on the terminal prediction. The second term captures the indirect effect through the state dependence of the pretrained velocity field. The factor $(1-t)$ reduces the influence of $\boldsymbol{J}_{\boldsymbol{v}}$ near the terminal time, with $\nabla_{\boldsymbol{x}_t} V$ approaching $\nabla_{\hat{\boldsymbol{x}}_{1|t}} \mathcal{L}$.

Computing the first term requires only the constraint-loss gradient at the terminal prediction. The second term additionally requires differentiating through the pretrained velocity model, increasing the cost of the feedback. We only need to compute the vector–Jacobian product $\boldsymbol{J}_{\boldsymbol{v}}^\top \nabla_{\hat{\boldsymbol{x}}_{1|t}} \mathcal{L}$ using reverse-mode automatic differentiation (Baydin et al., 2018). In the latent-space extension, we further consider omitting this Jacobian-dependent term for efficiency, as discussed in Section 3.2. See Appendices B.4 and C.4 for efficiency analyses.

The controller also depends on the explicit time derivative $\partial_t V$. Differentiating the terminal prediction with respect to t and combining the result with Eq. (12) yields

$$\partial_t V + \nabla_{\boldsymbol{x}_t} V^\top \boldsymbol{v}_\theta = (1-t) \nabla_{\hat{\boldsymbol{x}}_{1|t}} \mathcal{L}^\top \left[\partial_t \boldsymbol{v}_\theta + \boldsymbol{J}_{\boldsymbol{v}} \boldsymbol{v}_\theta\right]. \tag{13}$$

The term in brackets is the rate of change of the pretrained velocity along its own trajectory. The uncontrolled part of $\dot{V}$ measures how the predicted terminal constraint changes because the learned velocity field varies along the flow.

Algorithm 1 LyapuFlow (Data space)

Require: $\boldsymbol{v}_\theta$, decay rate α, trust region factor c_{tr}

$\boldsymbol{x}_0 \sim p_0$
for $k = 0, \dots, K-1$ **do**
  $\boldsymbol{v}_k \leftarrow \boldsymbol{v}_\theta(\boldsymbol{x}_k, t_k)$
  $\hat{\boldsymbol{x}}_{1|k} \leftarrow \boldsymbol{x}_k + (1 - t_k)\boldsymbol{v}_k$
  $V_k \leftarrow \mathcal{L}(\hat{\boldsymbol{x}}_{1|k})$
  $\boldsymbol{g}_k \leftarrow \nabla_{\boldsymbol{x}_k} V_k, \quad q_k \leftarrow \partial_t V(\boldsymbol{x}_k, t_k)$
  $\boldsymbol{u}_k^\star \leftarrow -\dfrac{[q_k + \boldsymbol{g}_k^\top \boldsymbol{v}_k + \alpha V_k]_+}{\|\boldsymbol{g}_k\|_2^2}\, \boldsymbol{g}_k$
  $\tilde{\boldsymbol{u}}_k \leftarrow \min\left(1, \dfrac{c_{\mathrm{tr}}\|\boldsymbol{v}_k\|_2}{\|\boldsymbol{u}_k^\star\|_2}\right)\boldsymbol{u}_k^\star$
  $\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{x}_k + \Delta t_k(\boldsymbol{v}_k + \tilde{\boldsymbol{u}}_k)$
end for
return $\boldsymbol{x}_K$

Algorithm 2 LyapuFlow (Latent space)

Require: $\boldsymbol{v}_\theta$, α, c_{tr}, decoder $\mathcal{D}$, gradient switch step k_{sg}

$\mathbf{z}_0 \sim p_0$
for $k = 0, \dots, K-1$ **do**
  $\bar{\boldsymbol{v}}_k \leftarrow \begin{cases} \mathrm{stopgrad}\left(\boldsymbol{v}_\theta(\mathbf{z}_k, t_k)\right), & k < k_{\mathrm{sg}}, \\ \boldsymbol{v}_\theta(\mathbf{z}_k, t_k), & \text{otherwise} \end{cases}$
  $\hat{\mathbf{z}}_{1|k} \leftarrow \mathbf{z}_k + (1 - t_k)\bar{\boldsymbol{v}}_k$
  $V_k \leftarrow \mathcal{L}\left(\mathcal{D}(\hat{\mathbf{z}}_{1|k})\right)$
  $\boldsymbol{g}_k \leftarrow \nabla_{\mathbf{z}_k} V_k, \quad q_k \leftarrow \partial_t V(\mathbf{z}_k, t_k)$
  $\boldsymbol{u}_k^\star \leftarrow -\dfrac{[q_k + \boldsymbol{g}_k^\top \bar{\boldsymbol{v}}_k + \alpha V_k]_+}{\|\boldsymbol{g}_k\|_2^2}\, \boldsymbol{g}_k$
  $\tilde{\boldsymbol{u}}_k \leftarrow \min\left(1, \dfrac{c_{\mathrm{tr}}\|\bar{\boldsymbol{v}}_k\|_2}{\|\boldsymbol{u}_k^\star\|_2}\right)\boldsymbol{u}_k^\star$
  $\mathbf{z}_{k+1} \leftarrow \mathbf{z}_k + \Delta t_k(\bar{\boldsymbol{v}}_k + \tilde{\boldsymbol{u}}_k)$
end for
return $\mathcal{D}(\mathbf{z}_K)$

3.1 Discretization and Feedback Trust Region

We derive Eq. (11) from a continuous-time Lyapunov decrease condition. The sampler applies this feedback on a discretized time grid $\{t_k\}_{k=0}^K$, where K denotes the number of function evaluations (NFEs). Let $\boldsymbol{x}_k$ denote the state at integration step k and $V_k := V(\boldsymbol{x}_k, t_k)$. Under Euler integration, the dynamics controlled by $\boldsymbol{u}_k^\star = -\eta_k \nabla_{\boldsymbol{x}_k} V_k$ is

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta t_k\, \boldsymbol{v}_\theta(\boldsymbol{x}_k, t_k) - \Delta t_k\, \eta_k \nabla_{\boldsymbol{x}_k} V_k. \tag{14}$$

After this discretization, a feedback magnitude chosen to satisfy the instantaneous condition in Eq. (8) can be too large for a finite-step update, moving the sample beyond the region in which the local decrease was evaluated (Taylor et al., 2021). These large corrections are also undesirable for generative flows, where deviations from the learned transport path can degrade sample fidelity (Saini et al., 2026). We introduce a feedback trust region and characterize the resulting trade-off.

Let $\tilde{\boldsymbol{u}}_k$ denote the actual control applied at integration step k. We quantify the resulting violation of the continuous-time Lyapunov condition by

$$\xi_k = \left[\partial_t V_k + \nabla_{\boldsymbol{x}_k} V_k^\top (\boldsymbol{v}_k + \tilde{\boldsymbol{u}}_k) + \alpha V_k\right]_+. \tag{15}$$

If the applied feedback is the closed-form control $\tilde{\boldsymbol{u}}_k = \boldsymbol{u}_k^\star$ from Eq. (11), then the Lyapunov condition is satisfied and $\xi_k = 0$. When the feedback is *bounded*, $\tilde{\boldsymbol{u}}_k$ may differ from $\boldsymbol{u}_k^\star$, and ξ_k measures the resulting loss in the requested Lyapunov decrease.

Proposition 1 (Discrete-time Lyapunov bound) *Suppose that $V(\boldsymbol{x}, t)$ has an L_V-Lipschitz gradient with respect to $(\boldsymbol{x}, t)$. Under an Euler step with step size Δt_k and applied feedback $\tilde{\boldsymbol{u}}_k$,*

$$V_{k+1} - V_k \le -\alpha \Delta t_k V_k + \Delta t_k \xi_k + \frac{L_V}{2} \Delta t_k^2 \left(1 + \|\boldsymbol{v}_k + \tilde{\boldsymbol{u}}_k\|_2^2\right). \tag{16}$$

The second term on the right-hand side measures the loss in Lyapunov decrease when the feedback is bounded. The final term is the discretization error, which grows with the magnitude of the controlled velocity; hence, bounding the control can limit this contribution. Therefore, Proposition 1 reveals a trade-off between *preserving the desired Lyapunov decrease* and *limiting the discretization error*, which can be balanced through an appropriate choice of the feedback bound.

To this end, we define the *feedback trust region* to bound the feedback displacement, $\|\Delta t_k\, \tilde{\boldsymbol{u}}_k\|_2 \le \Delta t_k \bar{u}_k$. While explicitly optimizing $\bar{u}_k$ at every step would incur additional computation, we instead use a simple adaptive rule based on the flow magnitude

$$\bar{u}_k = c_{\mathrm{tr}} \|\boldsymbol{v}_k\|_2, \tag{17}$$

for a positive constant c_{tr}.

The resulting control applied at integration step k is

$$\tilde{\boldsymbol{u}}_k = \frac{\bar{u}_k}{\max\left(\bar{u}_k, \|\boldsymbol{u}_k^\star\|_2\right)} \boldsymbol{u}_k^\star. \tag{18}$$

Algorithm 1 summarizes our sampler. See Appendix A.2 for the proofs and additional analysis.

3.2 Practical Extension to Latent-Space Models

The above formulations directly apply when the flow and constraint are defined in the same data space. In domains with established latent representations, generative models often operate in latent space while constraints remain in data space (Rombach et al., 2022; Rout et al., 2023). Let $\boldsymbol{z}_t$ denote the latent state and $\mathcal{D}$ the decoder. The terminal prediction then follows the computation path

$$\boldsymbol{z}_t \xrightarrow{\boldsymbol{v}_\theta} \hat{\boldsymbol{z}}_{1|t} = \boldsymbol{z}_t + (1-t)\boldsymbol{v}_\theta(\boldsymbol{z}_t, t) \xrightarrow{\mathcal{D}} \hat{\boldsymbol{x}}_{1|t} = \mathcal{D}(\hat{\boldsymbol{z}}_{1|t}), \tag{19}$$

and we define the Lyapunov function using the decoded terminal prediction $V(\boldsymbol{z}_t, t) = \mathcal{L}\left(\mathcal{D}(\hat{\boldsymbol{z}}_{1|t})\right)$. Consequently, the constraint gradient with respect to the latent state becomes

$$\nabla_{\boldsymbol{z}_t} V = \boldsymbol{J}_{\mathcal{D}}^\top \nabla_{\hat{\boldsymbol{x}}_{1|t}} \mathcal{L} + (1-t)\boldsymbol{J}_{\boldsymbol{v}}^\top \boldsymbol{J}_{\mathcal{D}}^\top \nabla_{\hat{\boldsymbol{x}}_{1|t}} \mathcal{L}, \tag{20}$$

where $\boldsymbol{J}_{\mathcal{D}} = \partial \mathcal{D}(\hat{\boldsymbol{z}}_{1|t})/\partial \hat{\boldsymbol{z}}_{1|t}$ and $\boldsymbol{J}_{\boldsymbol{v}} = \partial \boldsymbol{v}_\theta(\boldsymbol{z}_t, t)/\partial \boldsymbol{z}_t$. This is the latent-space counterpart of Eq. (12). The first term propagates the constraint gradient through the decoder alone, whereas the second term accounts for how perturbations of the current latent state alter the predicted endpoint through the velocity field. In modern latent generative models, the velocity network is often larger than the decoder (Chen et al., 2024; Fang et al., 2024). We therefore consider stopping gradients through the velocity prediction, $\bar{\boldsymbol{v}}_t = \mathrm{stopgrad}(\boldsymbol{v}_\theta(\boldsymbol{z}_t, t))$, and use the resulting surrogate gradient

$$\tilde{\boldsymbol{g}}_t = \boldsymbol{J}_{\mathcal{D}}^\top \nabla_{\hat{\boldsymbol{x}}_{1|t}} \mathcal{L}. \tag{21}$$

The $\mathrm{stopgrad}$ operation removes the dependence of the velocity prediction on both the latent state and time, thus simplifying the effective form of the unbounded feedback to

$$\boldsymbol{u}_t^\star = -\frac{\alpha V}{\|\tilde{\boldsymbol{g}}_t\|_2^2} \tilde{\boldsymbol{g}}_t. \tag{22}$$

These surrogate derivatives need not coincide with the true derivatives of V, so this latent approximation does not in general guarantee decrease of the true Lyapunov function. In most of our latent-space experiments, this approximation provides sufficient constraint guidance throughout sampling. For settings requiring high-frequency correction, restoring the $\boldsymbol{J}_{\boldsymbol{v}}$ contribution can be beneficial (Lai et al., 2026). Empirically, the decoder-propagated gradient primarily corrects lower-frequency components, while the flow-aware term in Eq. (20) improves higher-frequency structure. However, including the velocity-Jacobian term from the beginning can overemphasize high-frequency corrections and lead to overly aggressive refinement. Thus, we introduce a gradient switch step k_{sg} that omits the velocity- Jacobian contribution at early steps while enabling the full gradient at later steps:

$$\bar{\boldsymbol{v}}_k = \begin{cases} \mathrm{stopgrad}\left(\boldsymbol{v}_\theta(\boldsymbol{z}_k, t_k)\right), & k < k_{\mathrm{sg}}, \\ \boldsymbol{v}_\theta(\boldsymbol{z}_k, t_k), & k \geq k_{\mathrm{sg}}. \end{cases} \tag{23}$$

Algorithm 2 summarizes the LyapuFlow sampling procedure in latent space.

4 Experimental Results

We evaluate whether LyapuFlow across two diverse applications: physics-constrained PDE solution generation in data space and image inverse problems in latent space. For comparison, we consider several representative baselines for test-time constrained generation, including DPS (Chung et al., 2023), SAD-Flower (Huang et al., 2025), FlowDPS (Kim et al., 2025), ProFlow (Yu et al., 2026), PCFM (Utkarsh et al., 2025), and OC-Flow (Wang et al., 2024). DPS denotes its flow-based variant, which differentiates through the velocity model. We also compare to efficient low-iteration (analysis is provided in Appendix C.4) latent-space-oriented baselines (FlowChef (Patel et al., 2025) and FlowDPS (Kim et al., 2025)). See Appendix B for experimental details on the PDE experiments and Appendix C for the image experiments, including datasets, pretrained models, constraint definitions, evaluation metrics, baseline configurations, and hyperparameters.

4.1 Physics-Constrained Generation

We first consider PDE solution generation, where a pretrained flow is guided to satisfy hard physical requirements not explicitly imposed during training. In this setting, high likelihood under the

Table 1: Quantitative evaluation for physics-constrained generation. Bold: best; underline: second-best. All methods apply the same number of final projections (Burgers: 3; other tasks: 1).

Dataset	Metric ↓	LyapuFlow	DPS	SAD-Flower	FlowDPS	ProFlow	PCFM	OC-Flow	FFM w/ Proj.
1D Heat	MMSE/10^{-2}	**0.012**	0.221	3.463	2.127	0.485	0.115	0.134	5.206
	SMSE/10^{-2}	0.004	**0.002**	1.835	1.234	0.335	0.700	0.339	2.809
	FPD	**0.006**	0.093	1.064	0.826	0.377	0.639	0.353	1.481
	CE(IC)/10^{-2}	0	0	0	0	0	0	0	0
	CE(CL)/10^{-2}	0	0	0.003	0.001	0.002	0	0.001	0.001
1D Burgers	CMMSE/10^{-2}	**0.163**	0.254	6.874	7.578	9.326	4.303	4.172	8.266
	CSMSE/10^{-2}	**0.212**	0.377	5.708	6.234	6.896	2.171	6.150	6.715
	FPD	**0.204**	0.649	3.967	2.911	3.307	7.072	1.020	1.414
	CE(IC)/10^{-2}	0	0	0.002	0.003	0.003	0	0.002	0.003
	CE(ND)/10^{-2}	0.010	0.015	10	18	27	0	13	20
	CE(CL)/10^{-2}	0.005	0.007	21	116	152	0	82	124
1D RD	CMMSE/10^{-2}	**0.019**	0.090	3.379	2.070	1.086	0.386	0.083	3.931
	CSMSE/10^{-2}	**0.028**	0.053	1.903	1.138	0.819	1.225	0.194	2.196
	FPD	**0.002**	0.008	12	24	4.613	3.712	1.928	7.808
	CE(IC)/10^{-2}	0	0	0	0	0	0	0	0
	CE(NCL)/10^{-2}	0	0	0	0.002	0.001	0	0	0.008
2D NS	CMMSE/10^{-2}	**5.457**	5.871	23	22	22	23	13	23
	CSMSE/10^{-2}	**2.480**	2.834	6.193	6.131	6.227	13	5.876	6.270
	FPD	0.568	0.763	0.663	0.638	0.366	0.314	**0.263**	0.653
	CE(IC)/10^{-2}	0	0	0	0	0	0	0	0
	CE(CL)/10^{-2}	0	0	0	0	0	0	0	0

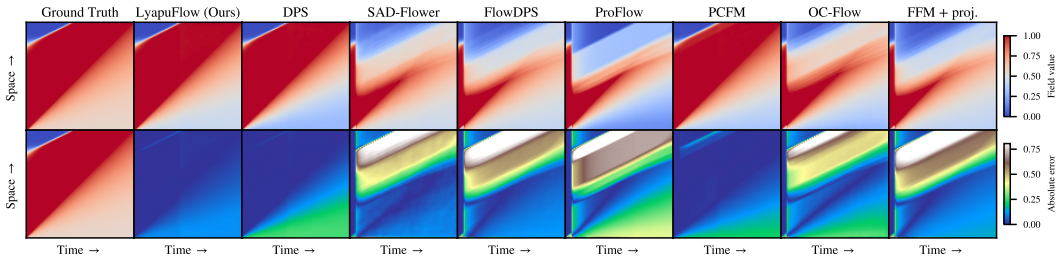

Figure 3: Average 1D Burgers solutions within a fixed-initial-condition group (top) and pointwise absolute error to the ground truth (bottom), subject to the IC, ND constraints over the first five time steps, and CL constraints. See Appendix B.7 for results on other PDE datasets.

learned data distribution does not by itself guarantee physical validity. Following ECI (Cheng et al., 2025) and PCFM (Utkarsh et al., 2025), we test on four PDE benchmarks: 1D Heat, 1D Burgers, 1D Reaction–Diffusion (RD), and 2D Navier–Stokes (NS). The benchmarks span smooth diffusive dynamics to evolution with sharp discontinuities. The evaluated constraints include initial conditions (IC), linear conservation laws (CL), nonlinear conservation laws (NCL), and nonlinear dynamics (ND) based on the Godunov update. Performance is assessed in terms of both constraint error (CE) and fidelity of the resulting conditional solution distribution (See Appendix B.3 for metrics.).

Table 1 shows that LyapuFlow achieves the best MMSE/CMMSE across all four tasks and the best SMSE/CSMSE and FPD on three tasks. LyapuFlow achieves zero or near-zero CE across all tasks, including Burgers after three final projection steps, while the baselines still exhibit non-negligible constraint error under the same projection budget. The main distinction is in sample fidelity, where LyapuFlow more accurately recovers the distribution associated with the specified physical conditions. Figure 3 illustrates this behavior for Burgers: among the baselines, only LyapuFlow and DPS recover the localized shock structure induced by the nonlinear dynamics. We provide a detailed comparison with DPS in Section 4.3.

4.2 Image Inverse Problems

We next consider image inverse problems with measurement-consistency constraints. Given a degradation operator $\mathcal{A}$ and noisy measurement $\boldsymbol{y}$, we define $\boldsymbol{h}(\boldsymbol{x}) = \mathcal{A}(\boldsymbol{x}) - \boldsymbol{y}$, so that satisfying $\boldsymbol{h}(\boldsymbol{x}) = \mathbf{0}$ corresponds to measurement consistency. We test $\times 4$ super-resolution, Gaussian deblurring, and motion deblurring across AFHQ, FFHQ, and DIV2K datasets using a pretrained latent flow

Table 2: Quantitative evaluation for image inverse problems. All experiments use Stable Diffusion 3.5-Medium with 28 NFEs. Bold: best; Underline: second best.

AFHQ 1k (768 × 768)												
	Super-resolution x4 (Bicubic)				Deblurring (Gauss)				Deblurring (Motion)			
Method	PSNR↑	SSIM↑	FID↓	LPIPS↓	PSNR↑	SSIM↑	FID↓	LPIPS↓	PSNR↑	SSIM↑	FID↓	LPIPS↓
FlowChef	29.43	0.798	27.30	0.095	25.22	0.717	40.81	0.274	26.33	0.739	40.83	0.230
FlowDPS	28.24	0.746	16.05	0.122	25.41	0.680	26.87	0.244	25.95	0.693	22.78	0.212
LyapuFlow (Ours)	**30.05**	**0.829**	**9.68**	**0.045**	**26.88**	**0.723**	**20.21**	**0.172**	**27.03**	**0.740**	**20.47**	**0.166**
FFHQ 1k (768 × 768)												
	Super-resolution x4 (Bicubic)				Deblurring (Gauss)				Deblurring (Motion)			
Method	PSNR↑	SSIM↑	FID↓	LPIPS↓	PSNR↑	SSIM↑	FID↓	LPIPS↓	PSNR↑	SSIM↑	FID↓	LPIPS↓
FlowChef	**30.02**	0.831	67.74	0.074	25.67	0.763	108.32	0.232	26.97	0.783	94.39	0.193
FlowDPS	28.99	0.756	**27.36**	0.098	26.03	0.716	46.25	0.210	26.74	0.723	**38.52**	0.177
LyapuFlow (Ours)	29.97	**0.836**	27.91	**0.046**	**27.56**	**0.761**	**41.83**	**0.144**	**27.81**	**0.778**	40.92	**0.138**
DIV2K 0.8k (768 × 768)												
	Super-resolution x4 (Bicubic)				Deblurring (Gauss)				Deblurring (Motion)			
Method	PSNR↑	SSIM↑	FID↓	LPIPS↓	PSNR↑	SSIM↑	FID↓	LPIPS↓	PSNR↑	SSIM↑	FID↓	LPIPS↓
FlowChef	24.47	0.671	51.53	0.111	20.72	0.533	98.34	0.343	21.77	0.575	86.97	0.285
FlowDPS	23.74	0.612	33.68	0.117	20.55	0.482	95.47	0.346	21.35	0.519	76.59	0.281
LyapuFlow (Ours)	**24.78**	**0.710**	**25.54**	**0.051**	**21.79**	**0.547**	**75.67**	**0.252**	**22.28**	**0.589**	**68.05**	**0.230**

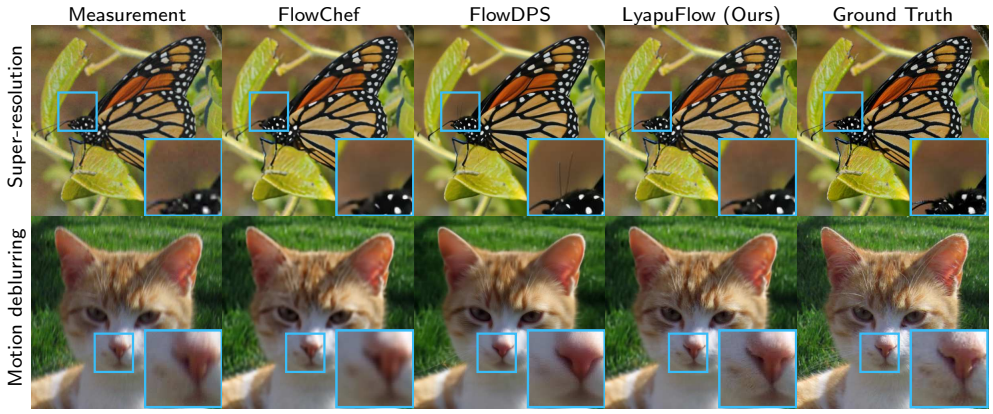

Figure 4: Qualitative comparison of reconstruction fidelity. LyapuFlow avoids hallucinated structures (top) while better preserving fine details (bottom).

model, Stable Diffusion 3.5-Medium (Esser et al., 2024). This setting tests whether LyapuFlow can impose data consistency while preserving the instance-specific structure of the underlying image.

Table 2 compares reconstruction quality across the three inverse problems under a matched 28-NFE budget. LyapuFlow achieves the best or second-best performance across metrics, including the best SSIM and LPIPS throughout. This is achieved with a single feedback update per flow step, without iterative inner optimization. Figure 4 shows the same trend qualitatively. In super-resolution, FlowDPS introduces antenna-like structures on the butterfly that are absent from the target image, while in motion deblurring it produces a hazier reconstruction and removes distinctive markings around the cat's mouth. These examples suggest that measurement guidance can produce plausible reconstructions while losing instance-specific details. LyapuFlow better preserves these structures while maintaining measurement consistency. See Appendix C.5 for additional results.

4.3 Ablation Studies

Role of LyapuFlow Components. We isolate two components of the feedback construction: the explicit time derivative and the feedback trust region. Table 3 shows that removing the time-derivative term consistently degrades fidelity on Burgers and NS (with full tables in Appendix B.5), consistent with its role in tracking how the predicted terminal constraint changes along the flow. Empirically, removing the trust region generally degrades performance, consistent with its role in limiting large finite-step corrections. See Appendix C.5 for ablations on the gradient switch step k_{sg}.

Hyperparameter Sensitivity. We compare LyapuFlow and DPS through their principal guidance parameters. DPS directly tunes a step size η_{DPS}, whereas LyapuFlow uses the decay rate α to

Table 3: Ablations on feedback trust region and time derivative.

Method	1D Burgers CMMSE 10^{-2}	1D Burgers CSMSE 10^{-2}	1D Burgers FPD	1D Burgers CE(IC) 10^{-2}	1D Burgers CE(ND) 10^{-2}	1D Burgers CE(CL) 10^{-2}	2D NS CMMSE 10^{-2}	2D NS CSMSE 10^{-2}	2D NS FPD	2D NS CE(IC) 10^{-2}	2D NS CE(CL) 10^{-2}
LyapuFlow (Ours)	**0.165**	**0.213**	**0.200**	0	1.532	0.978	**5.457**	2.480	**0.568**	0	0
w/o feedback trust region	0.285	0.237	0.474	0	1.303	0.896	5.497	**2.473**	0.576	0	0
w/o time derivative	0.574	0.401	1.890	0	4.757	3.067	6.038	2.712	0.761	0	0

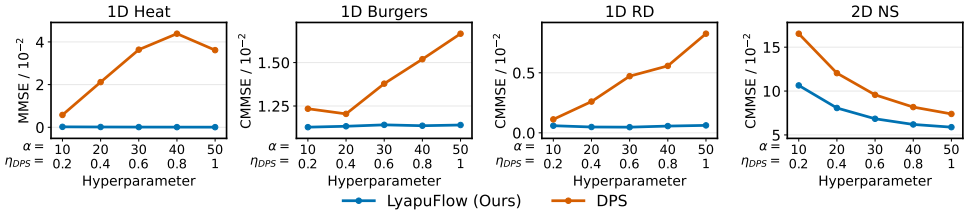

Figure 5: Task adaptability of DPS and LyapuFlow to base step size and decay rate, respectively.

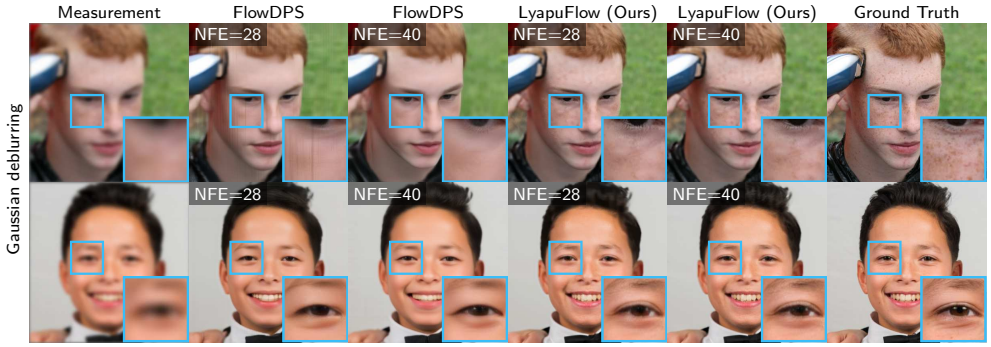

Figure 6: Qualitative comparison of reconstruction fidelity at different NFE budgets.

determine the required feedback. Figure 5 shows that the preferred DPS step size varies substantially across PDEs, with larger values helping NS but degrading Heat, Burgers, and RD. LyapuFlow is more stable across the tested values of α, suggesting lower sensitivity to task-specific dynamics. Extended results for all samplers are provided in Appendix B.6.

NFE Budget. We first test whether the reconstruction gains persist as the integration budget increases. Figure 6 compares FlowDPS and LyapuFlow at 28 and 40 NFEs. Increasing the budget removes some FlowDPS artifacts, but does not recover instance-specific details lost during reconstruction. In the first row, facial freckles remain substantially weakened at both budgets, while in the second row the eyelid crease remains absent. LyapuFlow preserves these structures at both budgets. The additional evaluations at 40 NFEs in Appendix C.5 show the same overall trend, indicating that the difference is not explained only by the number of integration steps.

5 Conclusion

We propose LyapuFlow, a test-time control framework for enforcing constraints on pretrained generative flows through Lyapunov feedback. By evaluating constraint violations on predicted terminal samples, LyapuFlow computes state-dependent corrections that are activated only when the pretrained dynamics fail to provide sufficient Lyapunov decrease. A feedback trust region further limits the control magnitude to reduce excessive finite-step corrections and the resulting discretization error. We demonstrate the approach in both data and latent spaces across scientific machine learning and image inverse problems. LyapuFlow consistently achieves strong constraint satisfaction and high sample fidelity relative to alternative test-time constraint-enforcement strategies.

Limitation. A current limitation is that LyapuFlow relies on differentiable constraint functions to construct the feedback signal. This restricts its direct applicability to constraints for which gradients with respect to the generated sample can be obtained. Extending LyapuFlow to non-differentiable, discrete, or black-box constraints is an interesting direction for future work.

AI USE STATEMENT

In this work, we used generative AI tools for helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refining hypotheses, implementing methods, and assisting with translation. We have not used generative AI tools for supporting qualitative and thematic data analysis, designing or providing feedback on research methodology or experiments, and interpreting results, and generating synthetic data sets and cleaning and reformatting dataset are not applicable to this work. Additionally, we used generative AI tools for summarizing or analyzing existing literature, creating or edit software code, creating or modify scientific figures or images, and proposing a title or keywords for a research paper. We have reviewed all AI-assisted writing and code implementations. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work advances fundamental methodologies for flexible and efficient generative modeling. This study does not involve human subjects, personally identifiable information, sensitive data, or deployment in high-stakes applications. We do not anticipate substantial ethical concerns beyond those generally associated with general-purpose generative models.

REPRODUCIBILITY STATEMENT

Appendix B provides details on how to construct the training dataset, pre-training velocity model, test dataset and constraints, hyperparameters, and evaluation metrics to reproduce the physics-constrained generation experiments in the paper. Appendix C provides details on how to construct the test dataset and text prompts, pre-trained latent-space velocity model, hyperparameters, and evaluation metrics to reproduce the image inverse problem experiments in the paper. We also provide the source code in the supplementary material for reproducibility.

REFERENCES

Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *Journal of machine learning research*, 18(153):1–43, 2018.

Heli Ben-Hamu, Omri Puny, Itai Gat, Brian Karrer, Uriel Singer, and Yaron Lipman. D-flow: Differentiating through flows for controlled generation. *arXiv preprint arXiv:2402.14017*, 2024.

Junsong Chen, Jincheng Yu, Chongjian Ge, Lewei Yao, Enze Xie, Zhongdao Wang, James Kwok, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart-α: Fast training of diffusion transformer for photorealistic text-to-image synthesis. In *International conference on learning representations*, volume 2024, pp. 57611–57640, 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.

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, 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*, 2023.

Giannis Daras, Hyungjin Chung, Chieh-Hsin Lai, Yuki Mitsufuji, Jong Chul Ye, Peyman Milanfar, Alexandros G Dimakis, and Mauricio Delbracio. A survey on diffusion models for inverse problems. *arXiv preprint arXiv:2410.00083*, 2024.

Julius Erbach, Dominik Narnhofer, Andreas Dombos, Bernt Schiele, Jan Eric Lenssen, and Konrad Schindler. Solving inverse problems with flair. *Advances in Neural Information Processing Systems*, 38:136667–136704, 2026.

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

Jiarui Fang, Jinzhe Pan, Xibo Sun, Aoyu Li, and Jiannan Wang. xdit: an inference engine for diffusion transformers (dits) with massive parallelism. *arXiv preprint arXiv:2411.01738*, 2024.

Randy Freeman and Petar V Kokotovic. *Robust nonlinear control design: state-space and Lyapunov techniques*. Springer Science & Business Media, 2008.

Derek Hansen, Danielle C Maddix, Shima Alizadeh, Gaurav Gupta, and Michael W Mahoney. Learning physical models that can respect conservation laws. In *International Conference on Machine Learning*, pp. 12469–12510. PMLR, 2023.

Maximilian Herde, Bogdan Raonić, Tobias Rohner, Roger Käppeli, Roberto Molinaro, Emmanuel De Bezenac, and Siddhartha Mishra. Poseidon: Efficient foundation models for pdes. *Advances in Neural Information Processing Systems*, 37:72525–72624, 2024.

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

Tzu-Yuan Huang, Armin Lederer, Dai-Jie Wu, Xiaobing Dai, Sihua Zhang, Hsiu-Chin Lin, Shao-Hua Sun, Stefan Sosnowski, and Sandra Hirche. SAD-Flower: Flow matching for safe, admissible, and dynamically consistent planning. *arXiv preprint arXiv:2511.05355*, 2025.

Katherine Keegan and Lars Ruthotto. Manifold-aware perturbations for constrained generative modeling. *arXiv preprint arXiv:2601.23151*, 2026.

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

Hassan K Khalil and Jessy W Grizzle. *Nonlinear systems*, volume 3. Prentice hall Upper Saddle River, NJ, 2002.

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.

Bolin Lai, Xudong Wang, Saketh Rambhatla, James M Rehg, Zsolt Kira, Rohit Girdhar, and Ishan Misra. Toward diffusible high-dimensional latent spaces: A frequency perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 43450–43460, 2026.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

Yuandan Lin, Eduardo D Sontag, and Yuan Wang. A smooth converse lyapunov theorem for robust stability. *SIAM Journal on Control and Optimization*, 34(1):124–160, 1996.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

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, 2023.

D Nešić and Andrew R Teel. Stabilization of sampled-data nonlinear systems via backstepping on their euler approximate model. *Automatica*, 42(10):1801–1808, 2006.

Dragan Nešić and Lars Grüne. Lyapunov-based continuous-time nonlinear controller redesign for sampled-data implementation. *Automatica*, 41(7):1143–1156, 2005.

Jonghyun Park and Jong Chul Ye. Flowlps: Langevin-proximal sampling for flow-based inverse problem solvers. *arXiv preprint arXiv:2512.07150*, 2025.

Maitreya Patel, Song Wen, Dimitris N Metaxas, and Yezhou Yang. FlowChef: Steering of rectified flow models for controlled generations. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15308–15318. IEEE, 2025.

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

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 10674–10685. ieee, 2022.

Litu Rout, Negin Raoof, Giannis Daras, Constantine Caramanis, Alex Dimakis, and Sanjay Shakkottai. Solving linear inverse problems provably via posterior sampling with latent diffusion models. *Advances in Neural Information Processing Systems*, 36:49960–49990, 2023.

Shreshth Saini, Shashank Gupta, and Alan Bovik. Rectified cfg++ for flow based models. *Advances in Neural Information Processing Systems*, 38:149034–149074, 2026.

Eduardo D Sontag. A 'universal'construction of artstein's theorem on nonlinear stabilization. *Systems & control letters*, 13(2):117–123, 1989.

Andrew J Taylor, Victor D Dorobantu, Yisong Yue, Paulo Tabuada, and Aaron D Ames. Sampled-data stabilization with control lyapunov functions via quadratically constrained quadratic programs. *IEEE Control Systems Letters*, 6:680–685, 2021.

Alexander Tong, Kilian FATRAS, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

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, 2025.

Utkarsh Utkarsh, Danielle C. Maddix, Ruijun Ma, Michael W. Mahoney, and Yuyang Wang. End-to-end probabilistic framework for learning with hard constraints. In *International Conference on Learning Representations*, volume 2026, pp. 14291–14333, 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.

Rongyuan Wu, Tao Yang, Lingchen Sun, Zhengqiang Zhang, Shuai Li, and Lei Zhang. Seesr: Towards semantics-aware real-world image super-resolution. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 25456–25467. IEEE, 2024.

Zewen Yang, Xiaobing Dai, Dian Yu, Zhijun Li, Majid Khadiv, Sandra Hirche, and Sami Haddadin. Uniconflow: A unified constrained flow-matching framework for certified motion planning. *arXiv preprint arXiv:2506.02955*, 2025.

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.

Bingliang Zhang, Wenda Chu, Julius Berner, Chenlin Meng, Anima Anandkumar, and Yang Song. Improving diffusion inverse problem solving with decoupled noise annealing. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20895–20905. IEEE, 2025.

APPENDIX CONTENTS

A Additional Analysis and Proofs **14**

A.1 Derivation of the Minimum-Norm Lyapunov Control 14

A.2 Discretization and Feedback Trust Region 15

B Physics-Constrained Generation **18**

B.1 Experimental Setup 18

B.2 Baselines and Evaluation Protocol 19

B.3 Evaluation Metrics 20

B.4 Sampling Efficiency 21

B.5 Component Ablations 22

B.6 Hyperparameter Sensitivity 22

B.7 Qualitative Results 23

C Image Inverse Problems **27**

C.1 Experimental Setup 27

C.2 Baselines and Evaluation Protocol 27

C.3 Evaluation Metrics 27

C.4 Sampling Efficiency 28

C.5 Additional Results and Ablations 29

C.6 Frequency Analysis 30

C.7 Qualitative Results 31

A Additional Analysis and Proofs

This appendix provides the derivations and additional conditions in Section 3. We adapt standard tools from robust Lyapunov analysis and sampled-data nonlinear control to address the approximation introduced by LyapuFlow: the finite-step implementation of the feedback Sontag (1989); Lin et al. (1996); Nešić & Grüne (2005); Nešić & Teel (2006).

Throughout, all regularity and boundedness assumptions are required only on the region traversed by the trajectories under consideration. We assume that the pretrained velocity field $\boldsymbol{v}_\theta(\boldsymbol{x}, t)$ is sufficiently smooth for the corresponding flow map and derivatives below to exist, and that the constraint loss $\mathcal{L}$ is continuously differentiable. The discrete-time result requires the stronger smoothness condition stated separately in Appendix A.2.

A.1 Derivation of the Minimum-Norm Lyapunov Control

We derive the minimum-norm control in Eq. (11), which is the optimal solution of

$$\begin{aligned} \text{minimize} \ & \frac{1}{2}\|\boldsymbol{u}_t\|_2^2 \\ \text{s.t.} \ & \nabla_{\boldsymbol{x}_t} V^\top \boldsymbol{u}_t + \partial_t V + \nabla_{\boldsymbol{x}_t} V^\top \boldsymbol{v}_\theta(\boldsymbol{x}_t, t) + \alpha V \le 0. \end{aligned} \tag{24}$$

For brevity, define

$$\boldsymbol{g} = \nabla_{\boldsymbol{x}_t} V, \qquad b = \partial_t V + \nabla_{\boldsymbol{x}_t} V^\top \boldsymbol{v}_\theta(\boldsymbol{x}_t, t) + \alpha V, \tag{25}$$

so that the constraint becomes

$$\boldsymbol{g}^\top \boldsymbol{u}_t + b \le 0. \tag{26}$$

For the corresponding Lagrangian

$$\mathcal{J}(\boldsymbol{u}_t, \lambda) = \frac{1}{2}\|\boldsymbol{u}_t\|_2^2 + \lambda(\boldsymbol{g}^\top \boldsymbol{u}_t + b), \qquad \lambda \ge 0, \tag{27}$$

the KKT conditions are

$$\boldsymbol{u}_t^\star + \lambda^\star \boldsymbol{g} = 0, \qquad \text{(stationarity)} \tag{28}$$

$$\boldsymbol{g}^\top \boldsymbol{u}_t^\star + b \le 0, \qquad \text{(primal feasibility)} \tag{29}$$

$$\lambda^\star \ge 0, \qquad \text{(dual feasibility)} \tag{30}$$

$$\lambda^\star(\boldsymbol{g}^\top \boldsymbol{u}_t^\star + b) = 0, \qquad \text{(complementary slackness).} \tag{31}$$

From stationarity,

$$\boldsymbol{u}_t^\star = -\lambda^\star \boldsymbol{g}. \tag{32}$$

If $b \le 0$, an inactive control $\boldsymbol{u}_t^\star = 0$ is feasible and optimal. If $b > 0$, assuming $\|\boldsymbol{g}\|_2 > 0$ the constraint is active at the optimum, giving

$$-\lambda^\star \|\boldsymbol{g}\|_2^2 + b = 0, \tag{33}$$

and hence

$$\lambda^\star = \frac{b}{\|\boldsymbol{g}\|_2^2}. \tag{34}$$

Combining both cases,

$$\lambda^\star = \frac{[b]_+}{\|\boldsymbol{g}\|_2^2}. \tag{35}$$

By Eq. (32),

$$\boldsymbol{u}_t^\star = -\frac{[b]_+}{\|\boldsymbol{g}\|_2^2}\boldsymbol{g} = -\frac{\left[\partial_t V + \nabla_{\boldsymbol{x}_t} V^\top \boldsymbol{v}_\theta(\boldsymbol{x}_t, t) + \alpha V\right]_+}{\|\nabla_{\boldsymbol{x}_t} V\|_2^2}\nabla_{\boldsymbol{x}_t} V, \tag{36}$$

which proves Eq. (11).

A.2 Discretization and Feedback Trust Region

We now consider the finite-step implementation of the continuous-time LyapuFlow sampler. Let

$$V_k = V(\boldsymbol{x}_k, t_k), \qquad \boldsymbol{v}_k = \boldsymbol{v}_\theta(\boldsymbol{x}_k, t_k), \qquad \Delta t_k = t_{k+1} - t_k > 0, \tag{37}$$

and recall that the closed-form feedback can be written as

$$\boldsymbol{u}_k^\star = -\eta_k \nabla_{\boldsymbol{x}_k} V_k, \tag{38}$$

where η_k is the adaptive feedback gain defined in Eq. (11). Let $\tilde{\boldsymbol{u}}_k$ denote the feedback actually applied by the numerical sampler. The Euler update is

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta t_k \left(\boldsymbol{v}_k + \tilde{\boldsymbol{u}}_k\right). \tag{39}$$

The feedback therefore contributes the finite displacement $\Delta t_k \tilde{\boldsymbol{u}}_k$. Before any feedback bound is applied, its magnitude is

$$\left\|\Delta t_k \boldsymbol{u}_k^\star\right\|_2 = \Delta t_k \eta_k \left\|\nabla_{\boldsymbol{x}_k} V_k\right\|_2. \tag{40}$$

Thus, even though η_k is chosen to satisfy an instantaneous continuous-time condition, the actual correction made by the numerical sampler also depends on the integration step and the local gradient magnitude.

As in the main text, we measure any shortfall in the requested continuous-time Lyapunov decrease by

$$\xi_k = \left[\partial_t V_k + \nabla_{\boldsymbol{x}_k} V_k^\top \left(\boldsymbol{v}_k + \tilde{\boldsymbol{u}}_k\right) + \alpha V_k\right]_+. \tag{41}$$

Equivalently,

$$\partial_t V_k + \nabla_{\boldsymbol{x}_k} V_k^\top \left(\boldsymbol{v}_k + \tilde{\boldsymbol{u}}_k\right) \le -\alpha V_k + \xi_k. \tag{42}$$

If $\tilde{\boldsymbol{u}}_k = \boldsymbol{u}_k^\star$, the closed-form controller satisfies the Lyapunov condition and $\xi_k = 0$. Bounding the feedback can change the applied control and introduce a positive shortfall.

Proof of proposition 1. Define the augmented state

$$\boldsymbol{z}_k = \begin{bmatrix} \boldsymbol{x}_k \\ t_k \end{bmatrix}, \qquad \Delta \boldsymbol{z}_k = \Delta t_k \begin{bmatrix} \boldsymbol{v}_k + \tilde{\boldsymbol{u}}_k \\ 1 \end{bmatrix}. \tag{43}$$

Assume that the gradient of V with respect to the augmented variable $(\boldsymbol{x}, t)$ is L_V-Lipschitz on a neighborhood containing the line segment between $\boldsymbol{z}_k$ and $\boldsymbol{z}_{k+1}$. The descent lemma gives

$$V_{k+1} - V_k \le \Delta t_k \left[\partial_t V_k + \nabla_{\boldsymbol{x}_k} V_k^\top \left(\boldsymbol{v}_k + \tilde{\boldsymbol{u}}_k\right)\right] + \frac{L_V}{2} \|\Delta \boldsymbol{z}_k\|_2^2. \tag{44}$$

Since

$$\|\Delta \boldsymbol{z}_k\|_2^2 = \Delta t_k^2 \left(1 + \|\boldsymbol{v}_k + \tilde{\boldsymbol{u}}_k\|_2^2\right), \tag{45}$$

substituting Eq. (42) yields

$$V_{k+1} - V_k \le -\alpha \Delta t_k V_k + \Delta t_k \xi_k + \frac{L_V}{2} \Delta t_k^2 \left(1 + \|\boldsymbol{v}_k + \tilde{\boldsymbol{u}}_k\|_2^2\right), \tag{46}$$

which proves proposition 1.

The bound separates two effects of modifying the feedback. Restricting the control can increase ξ_k and therefore weaken the instantaneous Lyapunov decrease. Allowing a large control, on the other hand, increases the second-order term through the magnitude of the controlled velocity. This is the finite-step trade-off underlying the feedback trust region.

One-step contraction. The same bound gives a sufficient condition for preserving contraction after an Euler step. For any $\gamma \in (0, 1)$, suppose

$$\xi_k + \frac{L_V}{2} \Delta t_k \left(1 + \|\boldsymbol{v}_k + \tilde{\boldsymbol{u}}_k\|_2^2\right) \le (1 - \gamma) \alpha V_k. \tag{47}$$

Then

$$V_{k+1} \le \left(1 - \gamma \alpha \Delta t_k\right) V_k. \tag{48}$$

For $0 < \gamma \alpha \Delta t_k < 1$ and $V_k > 0$, the Euler step therefore retains a strict contraction.

This condition also illustrates why aggressive feedback can be problematic near a nearly satisfied constraint. Even when $\xi_k = 0$, the finite-step error increases with $\Delta t_k \|\boldsymbol{v}_k + \tilde{\boldsymbol{u}}_k\|_2$, whereas the right-hand side of Eq. (47) decreases with V_k. A feedback law that is valid instantaneously can therefore fail to reduce V after a coarse finite step.

Accumulated discretization error. Consider a uniform grid with $\Delta t = 1/K$. Suppose

$$\xi_k \le \bar{\xi}, \qquad \|\boldsymbol{v}_k\|_2 \le B_v, \qquad \|\tilde{\boldsymbol{u}}_k\|_2 \le B_u \tag{49}$$

for all k, and assume $0 < \alpha\Delta t < 1$. Then proposition 1 gives

$$V_{k+1} \le (1 - \alpha\Delta t)V_k + \Delta t\,\bar{\xi} + \frac{L_V}{2}\Delta t^2\left(1 + (B_v + B_u)^2\right). \tag{50}$$

Iterating the recurrence gives

$$\begin{aligned} V_K \le\ & (1 - \alpha\Delta t)^K V_0 \\ & + \left[\frac{\bar{\xi}}{\alpha} + \frac{L_V\Delta t}{2\alpha}\left(1 + (B_v + B_u)^2\right)\right]\left[1 - (1 - \alpha\Delta t)^K\right]. \end{aligned} \tag{51}$$

When $\bar{\xi} = 0$, the additional error introduced by Euler integration is $\mathcal{O}(\Delta t) = \mathcal{O}(1/K)$. Its constant grows with the magnitude of the applied feedback. Increasing the number of integration steps and increasing the allowed feedback magnitude therefore have coupled effects on the discrete sampler.

If $\bar{\xi} > 0$, grid refinement still removes the Euler discretization term, but the contribution $\bar{\xi}/\alpha$ need not vanish. This distinction is important: reducing Δt controls numerical integration error, whereas persistent relaxation of the Lyapunov condition introduces a separate source of error.

Effect of the feedback trust region. The feedback trust region in Eq. (18) bounds the finite displacement caused specifically by the feedback,

$$\|\Delta t_k \tilde{\boldsymbol{u}}_k\|_2 \le \rho_k. \tag{52}$$

We choose

$$\rho_k = \Delta t_k \bar{u}_k, \tag{53}$$

which is equivalent to $\|\tilde{\boldsymbol{u}}_k\|_2 \le \bar{u}_k$. With fixed $\bar{u}_k$, the feedback displacement therefore vanishes as the integration grid is refined. In contrast, keeping ρ_k fixed independently of Δt_k would permit $\|\tilde{\boldsymbol{u}}_k\|_2 = \mathcal{O}(1/\Delta t_k)$, so the finite-step error in proposition 1 would not necessarily vanish under grid refinement.

To make the effect of clipping explicit, write

$$\tilde{\boldsymbol{u}}_k = \beta_k \boldsymbol{u}_k^\star, \qquad \tilde{\eta}_k = \beta_k \eta_k, \tag{54}$$

where

$$\beta_k = \begin{cases} 1, & \|\boldsymbol{u}_k^\star\|_2 \le \bar{u}_k, \\ \dfrac{\bar{u}_k}{\|\boldsymbol{u}_k^\star\|_2}, & \|\boldsymbol{u}_k^\star\|_2 > \bar{u}_k. \end{cases} \tag{55}$$

Hence

$$\tilde{\boldsymbol{u}}_k = -\tilde{\eta}_k \nabla_{\boldsymbol{x}_k} V_k, \tag{56}$$

and the trust region acts by reducing the adaptive feedback gain while preserving its direction.

The corresponding Lyapunov shortfall can be computed exactly. Let $\boldsymbol{g}_k = \nabla_{\boldsymbol{x}_k} V_k$. From the definition of η_k,

$$\eta_k \|\boldsymbol{g}_k\|_2^2 = \left[\partial_t V_k + \boldsymbol{g}_k^\top \boldsymbol{v}_k + \alpha V_k\right]_+. \tag{57}$$

Substituting $\tilde{\boldsymbol{u}}_k = -\beta_k \eta_k \boldsymbol{g}_k$ into Eq. (41) gives

$$\xi_k = (1 - \beta_k)\eta_k \|\boldsymbol{g}_k\|_2^2 = (\eta_k - \tilde{\eta}_k)\|\boldsymbol{g}_k\|_2^2. \tag{58}$$

Thus, when the trust region is inactive, $\beta_k = 1$ and the requested instantaneous decrease is preserved. When clipping is active, the trust region reduces the finite feedback displacement

$$\Delta t_k \|\tilde{\boldsymbol{u}}_k\|_2 = \Delta t_k \tilde{\eta}_k \|\boldsymbol{g}_k\|_2 \tag{59}$$

at the cost of the explicitly characterized shortfall in Eq. (58). This makes the role of the feedback trust region precise: it moderates the effective finite-step correction rather than changing the direction selected by the Lyapunov controller.

Degenerate gradients. The closed-form controller assumes that the gradient of V provides a direction through which the control can change the Lyapunov derivative. If $\nabla_{\boldsymbol{x}} V = \mathbf{0}$ while

$$\partial_t V + \alpha V > 0, \tag{60}$$

then the control does not affect $\dot{V}$ to first order and the requested Lyapunov decrease cannot be achieved by any finite instantaneous control. Our analysis therefore assumes $\nabla_{\boldsymbol{x}} V \neq \mathbf{0}$ whenever a nonzero correction is required. If $\nabla_{\boldsymbol{x}} V = \mathbf{0}$ and the Lyapunov condition is already satisfied, we take $\boldsymbol{u}^\star = \mathbf{0}$.

B PHYSICS-CONSTRAINED GENERATION

This section provides the complete experimental setup and additional results for physics-constrained generation. We first describe the PDE tasks, pretrained generative backbones, test-time constraints, and evaluation protocol. We then provide additional analyses of computational cost, hyperparameter sensitivity, and the components of LyapuFlow, followed by expanded qualitative results.

B.1 EXPERIMENTAL SETUP

B.1.1 TASKS AND DATASETS

Following ECI and PCFM (Cheng et al., 2025; Utkarsh et al., 2025), we consider four PDE benchmarks spanning different dynamics and constraint structures: 1D Heat, 1D Burgers, 1D Reaction–Diffusion (RD), and 2D Navier–Stokes (NS). We use s for spatial coordinates and τ for physical time, and $\mathcal{U}$ denotes the uniform distribution. Table 4 summarizes the quantitative data information.

Table 4: Summary of PDE datasets.

Task	Spatial dimension	Temporal dimension	d	Train samples
1D Heat	100	100	10,000	5,000
1D Burgers	101	101	10,201	6,400
1D RD	128	100	12,800	6,400
2D NS	64×64	50	204,800	10,000

1D Heat. On the spatial domain $s \in [0, 2\pi)$ and time interval $\tau \in [0, 1]$, the dataset is generated analytically as

$$x(s, \tau) = \sin(s + \phi) \exp(-\nu\tau), \qquad \phi \sim \mathcal{U}(0, \pi), \quad \nu \sim \mathcal{U}(1, 5). \tag{61}$$

This is an exact solution of $x_\tau = \nu x_{ss}$ with periodic boundary conditions. The spatial and temporal grids contain 100 points each, and each trajectory is generated from a newly sampled (ϕ, ν) pair.

1D Burgers. The inviscid Burgers equation on $s, \tau \in [0, 1]$ is solved with a Godunov flux,

$$x_\tau + \partial_s \left(\tfrac{1}{2}x^2\right) = 0, \qquad x(s, 0) = \frac{1}{1 + \exp((s - p)/0.02)}, \qquad p \sim \mathcal{U}(0.2, 0.8). \tag{62}$$

One hundred spatial and temporal intervals yield a 101×101 stored field. The parameter p determines the initial field used as the test-time initial-condition target.

1D Reaction–Diffusion. The dataset is generated from

$$x_\tau = \nu x_{ss} + \rho x(1 - x), \qquad \rho = 0.01, \quad \nu = 0.005. \tag{63}$$

The spatial domain contains 128 cell-centered points in $[0, 1]$, with 100 stored times from 0 to 0.99. The left and right diffusive fluxes are sampled as $g_L \in [0, 0.05]$ and $g_R \in [-0.05, 0]$. Initial fields are normalized random combinations of up to three Fourier modes, with occasional sign, absolute-value, and compact-support transformations.

2D Navier–Stokes. On the unit two-torus, let $\mathbf{s} = (s_1, s_2)$ denote the spatial coordinate. The vorticity equation is

$$\partial_\tau \omega + \mathbf{v} \cdot \nabla_{\mathbf{s}} \omega = \mu \Delta_{\mathbf{s}} \omega + f, \qquad \Delta_{\mathbf{s}} \psi = \omega, \qquad \mathbf{v} = (\partial_{s_2} \psi, -\partial_{s_1} \psi), \qquad \mu = 10^{-3}. \tag{64}$$

Initial vorticity is sampled from a two-dimensional Gaussian random field. The forcing family is $f_j(\mathbf{s}) = 0.1\sqrt{2} \sin(2\pi(s_1 + s_2) + \phi_j)$ for 100 phases $\phi_j \in [0, \pi/2]$. Fourier-space Crank–Nicolson integration with $2/3$ dealiasing and internal step 10^{-3} runs to time 49 and records 50 snapshots.

B.1.2 GENERATIVE BACKBONE

We use unconditional functional flow matching (FFM) backbones (Kerrigan et al., 2023) so that the physical conditions imposed at test time are not available to the generative model during training. Each PDE-specific model uses an Fourier neural operator (FNO) (Li et al., 2020) to learn a velocity field $v_\theta(x,t)$ that transports a functional prior $x_0 \sim p_0$ to a solution $x_1 \sim p_{\text{data}}$. For $t \sim \mathcal{U}(0,1)$, training uses

$$x_t = (1-t)x_0 + tx_1, \qquad v_t = x_1 - x_0, \qquad \mathcal{L}_{\text{FFM}} = \mathbb{E}\left[\|v_\theta(x_t,t) - v_t\|^2\right]. \tag{65}$$

Initial conditions, boundary conditions, and forcings are not provided as explicit conditioning channels. The backbone instead models the distribution of complete solution fields induced by their variation.

The Heat, Burgers, and RD models use four Fourier layers, while the NS model uses two. For Heat, Burgers, and RD, p_0 is a Matérn Gaussian-process prior with length scale 0.001 and variance 1; NS uses an i.i.d. Gaussian prior. Heat, Burgers, and RD are trained with Adam using learning rate 3×10^{-4}, batch size 256, and 20k optimization steps. NS is trained with AdamW using the same learning rate, batch size 24, and 500k optimization steps.

B.1.3 TEST-TIME CONSTRAINTS

At inference time, each task introduces physical information that was not provided to the unconditional backbone during training. The constraints include direct conditions on the solution, such as an initial field, together with conservation, balance, or local-dynamics residuals that couple multiple parts of the generated trajectory. This provides a range of constraint dimensions and nonlinearities across the four PDEs.

Table 5 summarizes the residuals. Each constraint maps a flattened solution field $x \in \mathbb{R}^d$ to a residual $h(x) \in \mathbb{R}^m$, with quadratic constraint loss $\mathcal{L}(x) = \frac{1}{2}\|h(x)\|^2$. For Burgers, the local nonlinear-dynamics residual applies the Godunov update over the first five time transitions. Its mass-balance residual accounts for transport through the boundaries using the Burgers flux $F(x) = x^2/2$ and trapezoidal integration in time, rather than requiring total mass to remain constant. For RD, we define the mass $M^n = \int_\Omega x(s,\tau_n)\,ds.$, the spatially integrated reaction source $S^n = \int_\Omega \rho x(s,\tau_n)(1 - x(s,\tau_n))\,ds$, and the boundary diffusion contribution $\Phi^n = \nu[x_s(s_R,\tau_n) - x_s(s_L,\tau_n)]$. For NS, the zero-mean forcing motivates a total-vorticity conservation constraint.

Table 5: Summary of the test-time PDE constraints.

Task	Type	m	Residual
1D Heat	Linear IC	100	$x(:,0) - x^{\text{ref}}(:,0)$
	Linear CL	99	$\sum_s x(s,\tau)\Delta s - \sum_s x(s,0)\Delta s,\ \tau = 1,\dots,99$
	Total	199	
1D Burgers	Linear IC	101	$x(:,0) - x^{\text{ref}}(:,0)$
	Nonlinear Dynamics	495	$\mathcal{R}_j^n = x_j^{n+1} - x_j^n + \frac{\Delta\tau}{\Delta s}[G(x_j^n, x_{j+1}^n) - G(x_{j-1}^n, x_j^n)],\ n = 0,\dots,4$
	Nonlinear CL	100	$M^n - M^0 + \Delta\tau \sum_{q=0}^{n-1}(\Phi^q + \Phi^{q+1})/2,\ n = 1,\dots,100$
	Total	696	
1D RD	Linear IC	128	$x(:,0) - x^{\text{ref}}(:,0)$
	Nonlinear CL	99	$M^n - M^0 - \Delta\tau \sum_{q=0}^{n-1} \frac{(\Phi^q + S^q) + (\Phi^{q+1} + S^{q+1})}{2}, \qquad n = 1,\dots,100.$
	Total	227	
2D NS	Linear IC	4,096	$\omega(:,:,0) - \omega_0$
	Linear CL	49	$\sum_{s_1,s_2} \omega(\tau)\Delta s_1 \Delta s_2 - \sum_{s_1,s_2} \omega(0)\Delta s_1 \Delta s_2,\ \tau = 1,\dots,49$
	Total	4,145	

B.2 BASELINES AND EVALUATION PROTOCOL

We compare LyapuFlow with samplers that intervene at different points along the generative trajectory. DPS (Chung et al., 2023) and SAD-Flower (Huang et al., 2025) correct intermediate states x_t; FlowDPS (Kim et al., 2025), ProFlow (Yu et al., 2026), and PCFM (Utkarsh et al., 2025) operate on terminal predictions $\hat{x}_{1|t}$; and OC-Flow (Wang et al., 2024) optimizes the complete trajectory $x_{[0,1]}$.

DPS denotes its flow-based variant and provides a comparison that, like LyapuFlow, differentiates through the pretrained velocity model.

For a controlled comparison, all samplers use the same final projection after generation. The FFM reference performs unconditional sampling followed by this projection, without an intermediate test-time correction. We tune the principal sampler-specific hyperparameters on each task and report the selected values below. The evaluation configurations and method-specific hyperparameters are summarized in the following tables.

Table 6: Construction of the test-reference data used for PDE simulation. For each selected condition, reference trajectories are fetched from the indicated complementary condition set.

Dataset	Selected conditions	Reference fetch per condition	Total
1D Heat	1 IC ($\phi = \pi/4$)	512 analytic ν-trajectories, $\nu \sim \mathcal{U}[1, 5]$	512
1D Reaction–Diffusion	16 IC	32 BC trajectories per IC	512
1D Burgers	16 IC	32 BC trajectories per IC	512
2D Navier–Stokes	10 IC (vorticity)	10 forcing trajectories per IC (vorticity)	100

Table 7: Hyperparameters used in physics-constrained generation. Hyperparameter sweeps use $N = 64$ samples per dataset and $N_{\mathrm{idx}} \in \{16, 10\}$ selected indices for multi-condition datasets. α and α_{SF} are the LyapuFlow and SAD-Flower gains, respectively; η_{DPS} is the DPS base step size for $\eta_{\mathrm{DPS}}/\|\boldsymbol{h}(\boldsymbol{x})\|$; K_{iter} is the number of iterations; η_{FD}, η_{PF}, and η_{OC} are the FlowDPS, ProFlow, and OC-Flow step sizes, respectively.

Dataset	LyapuFlow	DPS	SAD-Flower	FlowDPS	ProFlow	OC-Flow
1D Heat	$\alpha = 50; c_{\mathrm{tr}} = 10$	$\eta_{\mathrm{DPS}} = 0.2$	$\alpha_{\mathrm{SF}} = 14$	$K_{iter} = 3; \eta_{\mathrm{FD}} = 0.02$	$K_{iter} = 3; \eta_{\mathrm{PF}} = 0.1$	$K_{iter} = 3; \beta = 0.995; \eta_{\mathrm{OC}} = 0.5$
1D Burgers	$\alpha = 10; c_{\mathrm{tr}} = 1$	$\eta_{\mathrm{DPS}} = 0.4$	$\alpha_{\mathrm{SF}} = 5$	$K_{iter} = 3; \eta_{\mathrm{FD}} = 0.06$	$K_{iter} = 3; \eta_{\mathrm{PF}} = 0.1$	$K_{iter} = 3; \beta = 0.995; \eta_{\mathrm{OC}} = 2.5$
1D RD	$\alpha = 30; c_{\mathrm{tr}} = 1$	$\eta_{\mathrm{DPS}} = 0.2$	$\alpha_{\mathrm{SF}} = 5$	$K_{iter} = 3; \eta_{\mathrm{FD}} = 0.06$	$K_{iter} = 3; \eta_{\mathrm{PF}} = 0.5$	$K_{iter} = 3; \beta = 0.995; \eta_{\mathrm{OC}} = 0.5$
2D NS	$\alpha = 50; c_{\mathrm{tr}} = 1$	$\eta_{\mathrm{DPS}} = 1$	$\alpha_{\mathrm{SF}} = 14$	$K_{iter} = 3; \eta_{\mathrm{FD}} = 0.08$	$K_{iter} = 3; \eta_{\mathrm{PF}} = 0.5$	$K_{iter} = 3; \beta = 0.995; \eta_{\mathrm{OC}} = 0.5$

B.3 Evaluation Metrics

A partial physical condition generally leaves other initial, boundary, or forcing variables unspecified and therefore does not determine a unique PDE trajectory. Each condition instead defines a distribution of admissible solutions. We evaluate whether the generated samples recover this conditional distribution in addition to satisfying the imposed constraints.

For a single condition group, MMSE compares the pointwise sample-mean fields,

$$\mathrm{MMSE} = \frac{1}{d}\|\bar{x} - \bar{x}^{\mathrm{ref}}\|^2, \qquad \mathrm{SMSE} = \frac{1}{d}\|\sigma_x - \sigma_x^{\mathrm{ref}}\|^2. \tag{66}$$

where $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$ and $\bar{x}^{\mathrm{ref}} = \frac{1}{N}\sum_{i=1}^{N} x_i^{\mathrm{ref}}$. SMSE compares the corresponding pointwise standard-deviation fields.

For Burgers, RD, and NS, we evaluate multiple partial conditions and report macro-averaged conditional metrics. Given G condition groups with index sets I_g,

$$\mathrm{CMMSE} = \frac{1}{G}\sum_{g=1}^{G}\frac{1}{d}\|\bar{x}_g - \bar{x}_g^{\mathrm{ref}}\|^2, \qquad \mathrm{CSMSE} = \frac{1}{G}\sum_{g=1}^{G}\frac{1}{d}\|\sigma_{x,g} - \sigma_{x,g}^{\mathrm{ref}}\|^2. \tag{67}$$

Here, $\bar{x}_g$ and $\sigma_{x,g}$ denote the group-wise mean and standard-deviation fields, with analogous definitions for the reference samples.

Constraint error measures the mean residual norm of individual generated samples,

$$\mathrm{CE} = \frac{1}{N}\sum_{i}\|h(x_i)\|. \tag{68}$$

We additionally report Fréchet Poseidon Distance (FPD) as a feature-space measure of distributional similarity. Generated and reference fields are embedded using the PDE foundation model Poseidon–B (Herde et al., 2024), and the Fréchet distance is computed from the resulting Gaussian feature

statistics. For the 1D datasets, each spatiotemporal field is processed as one feature. For NS, each time frame is embedded separately and the frame-wise distances are averaged. The 128×100 RD fields are center-cropped to 100×100 before feature extraction to match the model input shape.

Together, CE measures physical feasibility, while CMMSE, CSMSE, and FPD measure complementary aspects of fidelity to the conditional solution distribution.

B.4 Sampling Efficiency

The evaluated samplers differ substantially in how much computation is required to impose a test-time constraint. Table 8 compares per-sample runtime and peak GPU memory, including the common final projection used in the evaluation. FlowDPS, ProFlow, and OC-Flow use $K = 3$ inner iterations in the reported configuration, so additional iterations would further increase their cost. OC-Flow is the most expensive method across all four tasks, consistent with its repeated optimization of the complete control trajectory. PCFM also incurs additional cost from Jacobian evaluation and linear-system solves. LyapuFlow avoids trajectory-level iterative optimization and remains computationally competitive while applying constraint-aware feedback throughout sampling.

Table 8: Sampling efficiency measured on the same NVIDIA A100 GPU with batch size 1 and 100 sampling steps. Wall time is the mean elapsed time per generated sample over 5 timed runs after 1 warm-up run; peak allocated memory is the maximum CUDA allocation across those runs. Both measurements include one-step final projection.

Dataset	Metric ↓	LyapuFlow (Ours)	DPS	SAD-Flower	FlowDPS	ProFlow	PCFM	OC-Flow
1D Heat	Second / sample	1.106	0.802	0.401	0.71	0.774	0.618	2.992
	Peak memory (GiB)	0.167	0.157	0.115	0.115	0.833	0.493	0.545
1D Burgers	Second / sample	1.87	1.29	0.579	2.242	2.295	1.712	3
	Peak memory (GiB)	0.168	0.158	0.141	0.141	0.862	0.62	0.561
1D RD	Second / sample	1.351	0.959	0.475	1.297	1.375	0.987	2.867
	Peak memory (GiB)	0.186	0.174	0.119	0.119	1.31	0.748	0.805
2D NS	Second / sample	1.985	1.505	1.008	1.281	1.325	3.468	4.277
	Peak memory (GiB)	6.382	6.382	6.382	6.382	6.382	9.607	6.382

B.5 Component Ablations

We isolate the two components examined in the main-text ablation, the explicit time derivative in the Lyapunov dynamics and the feedback trust region. Table 9 shows that removing the time-derivative term generally increases fidelity error, with a particularly clear effect on FPD. This is consistent with its role in accounting for changes in the predicted terminal constraint along the pretrained flow. Removing the trust region has a smaller effect, with performance remaining close to the full method on several metrics. This supports its role as a safeguard against large finite-step corrections rather than as the primary source of the feedback signal.

Table 9: Ablations on feedback trust region and time derivative across all PDE tasks.

	1D Heat					1D Burgers					
Method	$\frac{\text{MMSE}}{10^{-2}}$	$\frac{\text{SMSE}}{10^{-2}}$	FPD	$\frac{\text{CE(IC)}}{10^{-2}}$	$\frac{\text{CE(CL)}}{10^{-2}}$	$\frac{\text{CMMSE}}{10^{-2}}$	$\frac{\text{CSMSE}}{10^{-2}}$	FPD	$\frac{\text{CE(IC)}}{10^{-2}}$	$\frac{\text{CE(ND)}}{10^{-2}}$	$\frac{\text{CE(CL)}}{10^{-2}}$
LyapuFlow (Ours)	**0.012**	**0.004**	**0.006**	0	0	**0.163**	**0.212**	**0.204**	0	0.010	0.005
w/o trust region	**0.012**	**0.004**	**0.006**	0	0	0.283	0.235	0.458	0	0.008	0.004
w/o time derivative	0.067	0.018	0.043	0	0	0.584	0.398	2.240	0	2.208	4.800

	1D RD					2D NS				
Method	$\frac{\text{CMMSE}}{10^{-2}}$	$\frac{\text{CSMSE}}{10^{-2}}$	FPD	$\frac{\text{CE(IC)}}{10^{-2}}$	$\frac{\text{CE(NCL)}}{10^{-2}}$	$\frac{\text{CMMSE}}{10^{-2}}$	$\frac{\text{CSMSE}}{10^{-2}}$	FPD	$\frac{\text{CE(IC)}}{10^{-2}}$	$\frac{\text{CE(CL)}}{10^{-2}}$
LyapuFlow (Ours)	**0.019**	**0.028**	**0.002**	0	0	**5.457**	2.480	**0.568**	0	0
w/o trust region	**0.019**	**0.028**	**0.002**	0	0	5.497	**2.473**	0.576	0	0
w/o time derivative	0.052	0.029	3.525	0	0	6.038	2.712	0.761	0	0

B.6 Hyperparameter Sensitivity

We evaluate whether the principal guidance parameter of each sampler transfers reliably across PDE dynamics or requires substantial task-specific adjustment. Figure 7 varies one parameter at a time while keeping the remaining settings fixed. PCFM has no corresponding tunable guidance parameter and is shown as a horizontal reference.

For each method, we first identify a coarse range that avoids non-finite evaluations and then tune within that range. Methods that directly parameterize the correction magnitude can still become unstable for some candidate values. LyapuFlow instead tunes the decay rate α, which specifies the requested Lyapunov decrease while the feedback magnitude is determined from the current state. Across the evaluated tasks, LyapuFlow varies less across the tested values than methods whose principal parameter directly controls a correction step size. The comparison with DPS in Fig. 5 provides the corresponding linear-scale view used in the main paper.

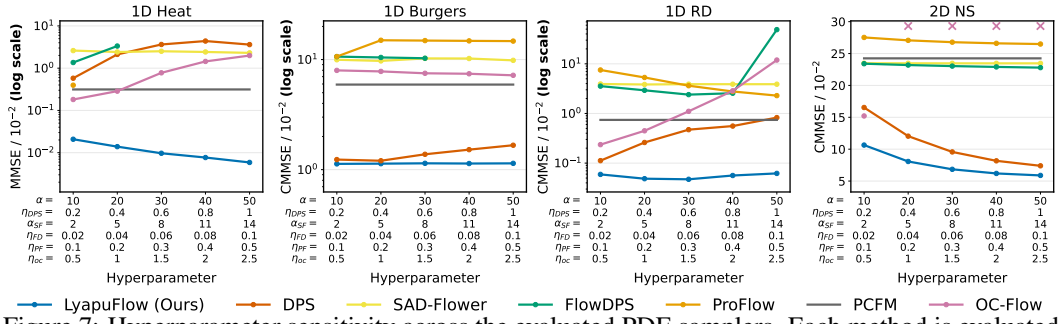

Figure 7: Hyperparameter sensitivity across the evaluated PDE samplers. Each method is evaluated over five values of its principal guidance parameter: α for LyapuFlow, η_{DPS} for DPS, α_{SF} for SAD-Flower, η_{FD} for FlowDPS, η_{PF} for ProFlow, and η_{OC} for OC-Flow. PCFM has no tuned guidance parameter and is shown as a horizontal reference. The 1D tasks use a logarithmic vertical scale, and non-finite evaluations are marked with $\times$.

B.7 Qualitative Results

The following examples complement the aggregate metrics by showing how the different samplers modify complete PDE solution fields under the imposed test-time conditions.

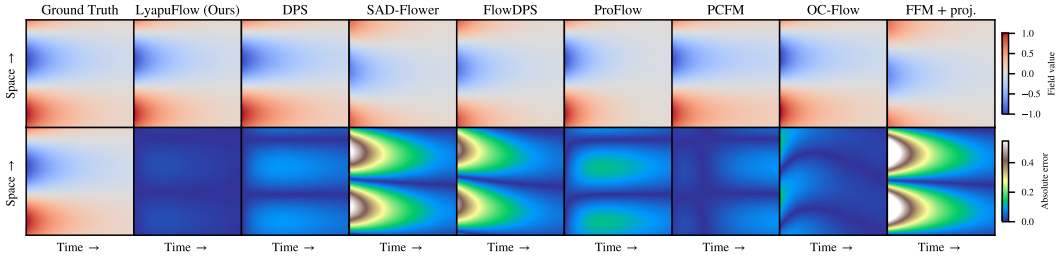

Figure 8: Qualitative comparison on 1D Heat under the evaluated initial-condition and conservation-law constraints.

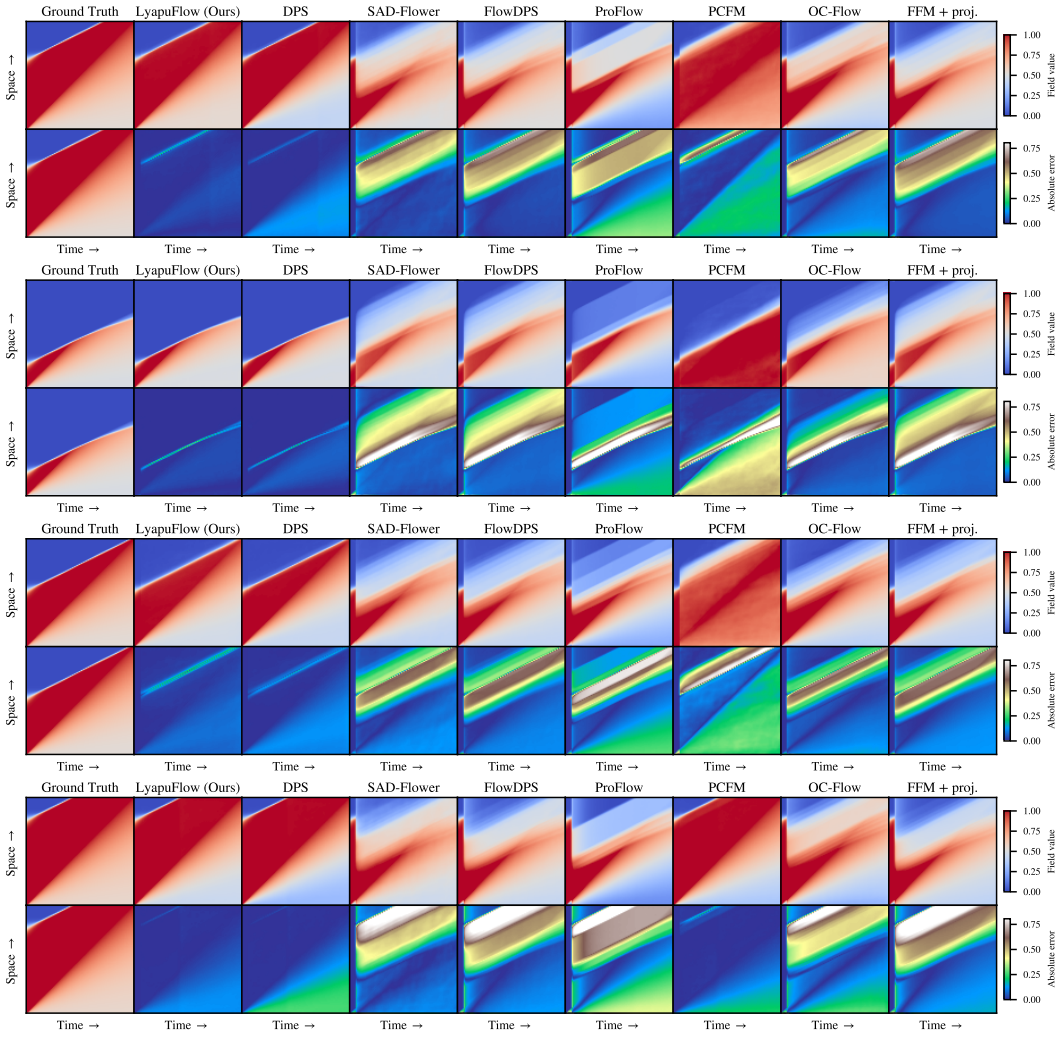

Figure 9: Qualitative comparisons on 1D Burgers under initial-condition, local-dynamics, and mass-balance constraints.

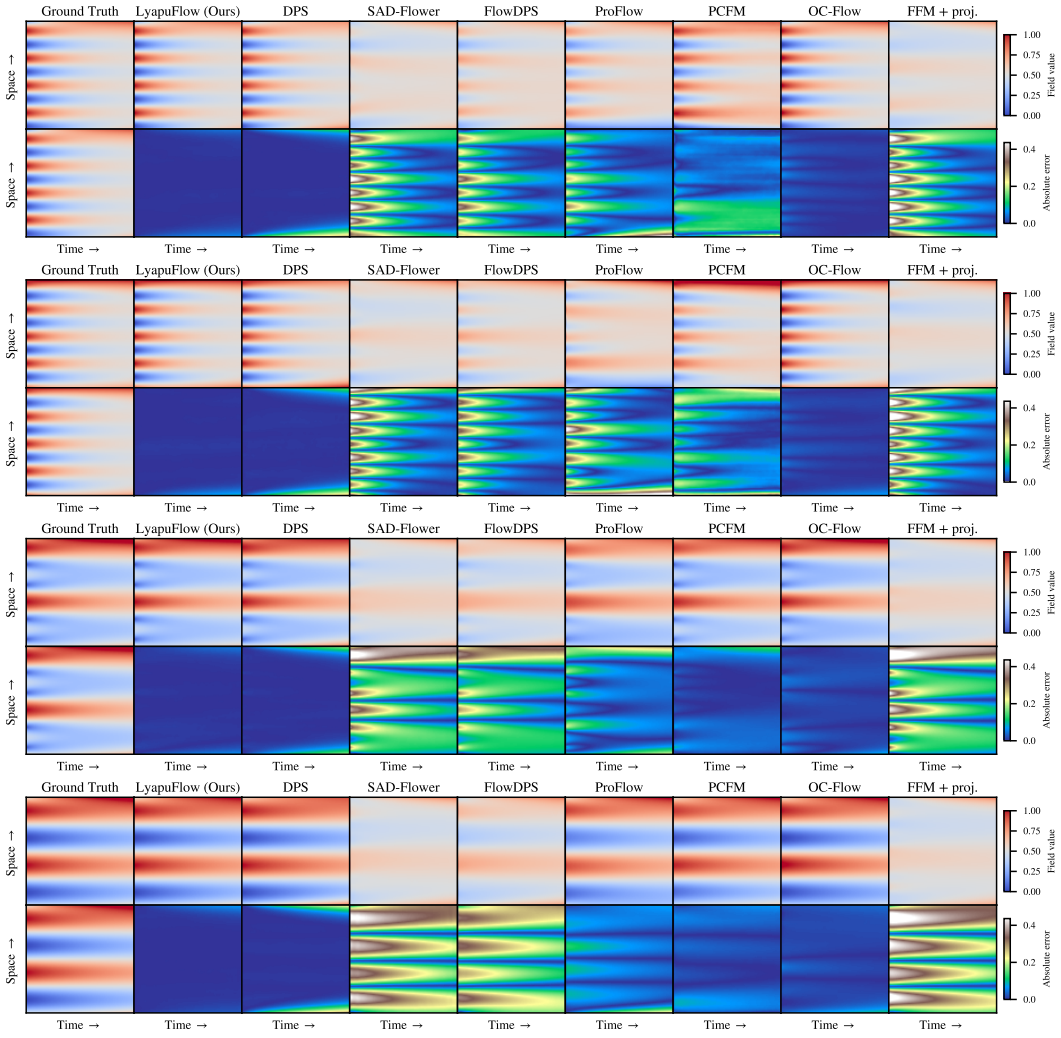

Figure 10: Qualitative comparisons on 1D Reaction–Diffusion under initial-condition and balance-law constraints.

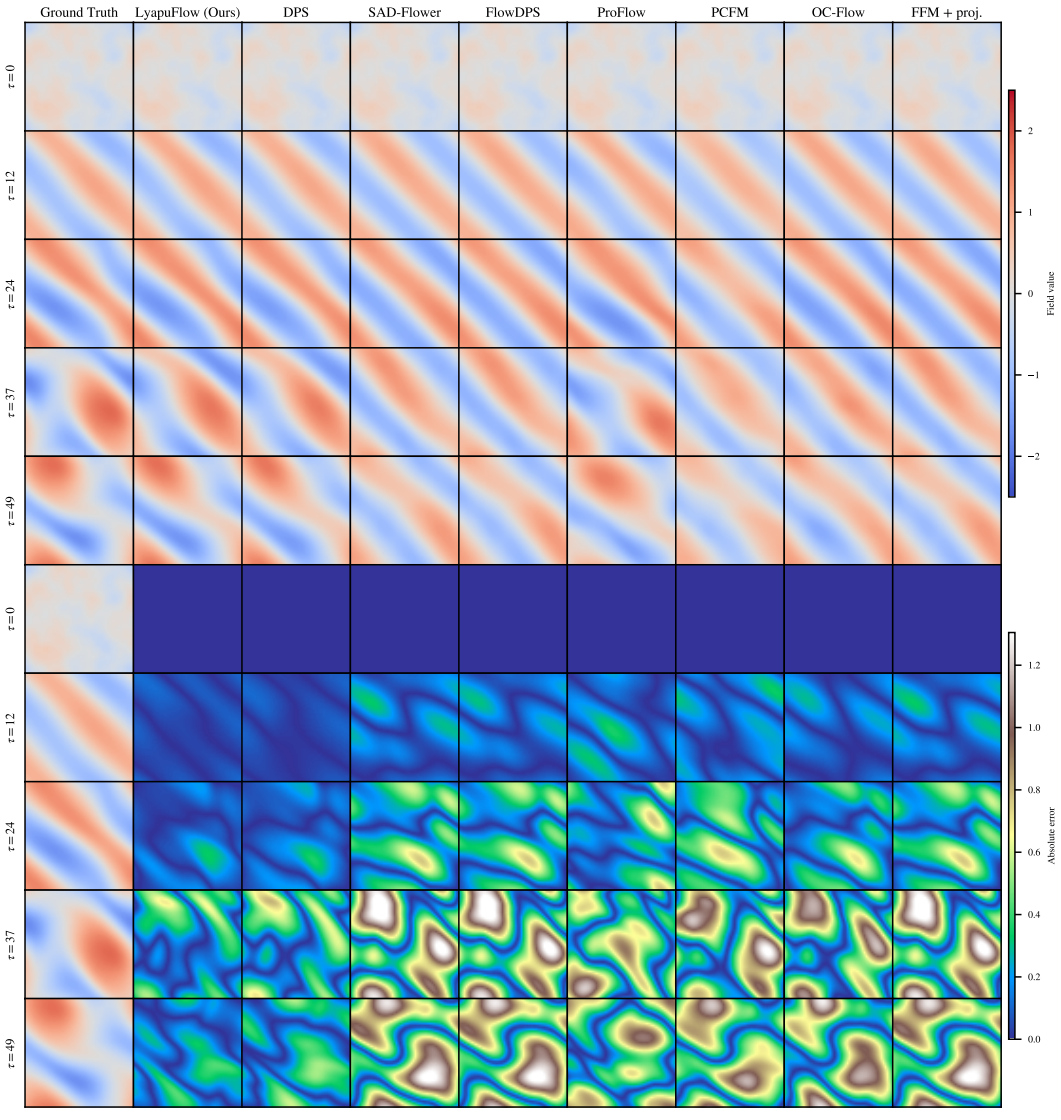

Figure 11: Qualitative comparisons on 2D Navier–Stokes under initial-vorticity and total-vorticity constraints.

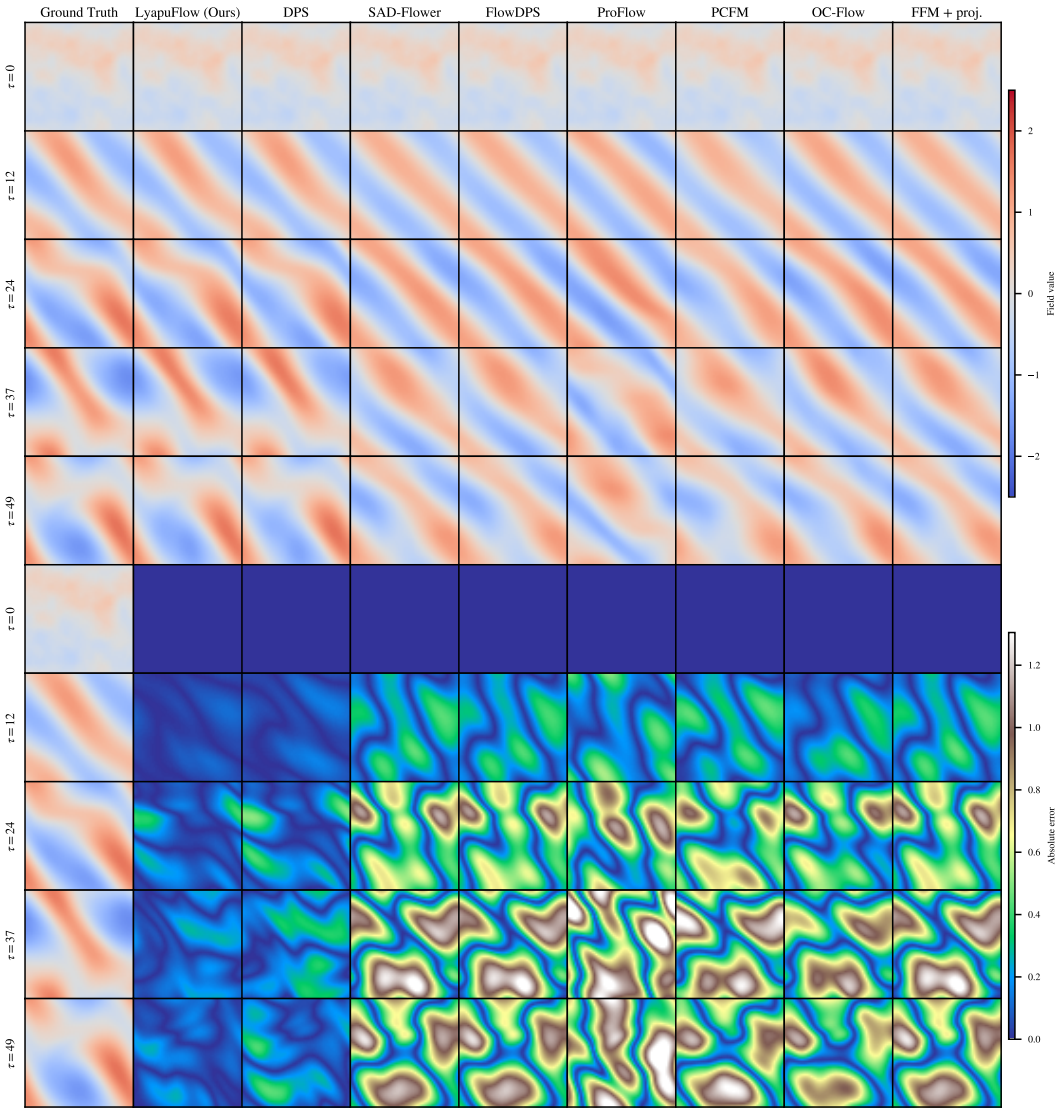

Figure 12: Qualitative comparisons on 2D Navier–Stokes under initial-vorticity and total-vorticity constraints.

C IMAGE INVERSE PROBLEMS

This section provides the complete experimental setup and additional results for latent-space image inverse problems. We first describe the datasets, pretrained backbone, degradation operators, baseline configurations, and evaluation protocol. We then examine sampling efficiency, performance at a larger integration budget, frequency-domain behavior, and additional qualitative reconstructions.

C.1 EXPERIMENTAL SETUP

C.1.1 DATASETS AND BACKBONE

We evaluate on 1k AFHQ images, comprising 0.5k dog and 0.5k cat images, 1k FFHQ images, and 0.8k DIV2K images, all at 768×768 resolution. All methods use Stable Diffusion 3.5-Medium (Esser et al., 2024) as the pretrained latent-space flow model with classifier-free guidance scale 2 (Ho & Salimans, 2022). AFHQ uses the prompts "a photo of a dog" and "a photo of a cat", FFHQ uses "a photo of a person," and DIV2K uses "a high quality photo of {DAPE tags}", with the tags extracted from the DAPE-degraded images (Wu et al., 2024).

C.1.2 INVERSE PROBLEMS AND OBSERVATIONS

We consider $\times 4$ bicubic super-resolution, Gaussian deblurring, and motion deblurring with kernel size 61. For each task, the observed image is obtained by applying the corresponding forward operator and adding measurement noise with $\sigma_n = 0.03$. The resulting observation defines the test-time data-consistency constraint. The same pretrained latent flow is therefore reused across all inverse problems, while only the observation operator and measurement change at inference time.

C.2 BASELINES AND EVALUATION PROTOCOL

We consider LatentDAPS (Zhang et al., 2025), FlowChef (Patel et al., 2025), FlowDPS (Kim et al., 2025), FLAIR (Erbach et al., 2026), and FlowLPS (Park & Ye, 2025) as candidate baselines. The primary main-text comparison focuses on FlowChef and FlowDPS because, like LyapuFlow, they operate without the much larger inner optimization loops used by several of the other methods. We report the computational cost of all candidate baselines separately in Appendix C.4.

Method-Specific Settings.

- **LatentDAPS (Zhang et al., 2025).** 20 EDM annealing steps from $\sigma = 1$ to 0.2 with $\rho = 7$; each interval uses two-step midpoint denoising, for 40 total NFEs. Non-SR tasks use 15 Langevin MCMC updates per annealing step with learning rate 10^{-4}. Super-resolution uses 60 updates with learning rate 10^{-3}, decayed by 0.7 every 5 MCMC updates; $\tau = 0.03$.
- **FlowChef (Patel et al., 2025).** One data-consistency update per flow step with step size 15.
- **FlowDPS (Kim et al., 2025).** Three data-consistency updates per flow step with step size 15.
- **FLAIR (Erbach et al., 2026).** Official repository defaults with 15 data-term updates per step, data-term SGD learning rate 1.0 for super-resolution and 0.1 for deblurring, regularizer scale 0.5, the bundled calibration schedule, and $t_{\min} = 0.18$.
- **FlowLPS (Park & Ye, 2025).** The unified setting from Appendix B.1 with 15 total Langevin/proximal updates, using $N_L = 5$ Langevin and $N_P = 10$ proximal-SGD updates. The Langevin step size is 10^{-4}, the proximal-SGD learning rate is 0.1 without decay, the schedule offset is 3, and pCN re-noising is used.
- **LyapuFlow.** See Table 10.

C.3 EVALUATION METRICS

We report peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) to measure reconstruction distortion and structural agreement. We additionally report Learned Perceptual Image Patch Similarity (LPIPS) and Fréchet Inception Distance (FID) to capture perceptual and distributional differences that are not fully reflected by pixel-space distortion.

Table 10: Task-specific LyapuFlow settings for 28 and 40 NFEs. k_{sg} denotes the control step at which stop-gradient on v_θ is turned off (zero-indexed); k_{sg} equal to the number of NFEs keeps it on throughout sampling.

		28 NFEs			40 NFEs		
Dataset	Task	α	k_{sg}	c_{tr}	α	k_{sg}	c_{tr}
AFHQ	Super-resolution (Bicubic)	5	28	1.0	5	40	1.0
AFHQ	Deblurring (Gaussian)	5	3	1.0	5	40	1.0
AFHQ	Deblurring (Motion)	5	7	1.0	5	40	1.0
FFHQ	Super-resolution (Bicubic)	5	28	1.0	5	40	1.0
FFHQ	Deblurring (Gaussian)	5	3	1.0	5	7	1.0
FFHQ	Deblurring (Motion)	5	7	1.0	5	7	1.0
DIV2K	Super-resolution (Bicubic)	5	28	1.0	5	40	1.0
DIV2K	Deblurring (Gaussian)	5	3	1.0	5	40	1.0
DIV2K	Deblurring (Motion)	7	7	1.0	5	40	1.0

C.4 Sampling Efficiency

The candidate image baselines differ substantially in the amount of computation performed within each sampling step. Table 11 compares reconstruction quality and runtime on a small evaluation subset. The iteration-heavy methods LatentDAPS, FLAIR, and FlowLPS incur substantially greater sampling cost than single-update methods. For example, FlowLPS improves PSNR over LyapuFlow by 1.21 dB on motion deblurring while requiring more than $8\times$ longer sampling time. This motivates the primary main-text comparison with FlowChef and FlowDPS, which operates in a more comparable quality–efficiency regime.

Table 11: Trade-off between sampling speed and quality, evaluated on the first 10 FFHQ images.

Sampler	N_{iter}	NFE	SR $\times 4$		Deblur (Gaussian)		Deblur (Motion)		Second / image
			PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	
LyapuFlow (Ours)	1	40	<u>30.63</u>	**0.041**	<u>28.76</u>	0.117	<u>28.87</u>	0.120	<u>13.8</u>
FlowChef	1	40	**30.90**	0.060	26.62	0.219	28.52	0.175	**11.9**
FlowDPS	3	40	30.09	0.101	26.57	0.208	27.83	0.171	26.6
FlowLPS	15	40	24.86	0.060	**28.87**	**0.075**	**30.08**	**0.065**	112.6
FLAIR	15	40	22.84	<u>0.044</u>	26.29	<u>0.094</u>	26.93	<u>0.088</u>	112.8
LatentDAPS	15 or 60	40	27.05	0.116	25.22	0.238	25.29	0.249	112.6

C.5 Additional Results and Ablations

We additionally evaluate the methods at 40 NFEs to test whether the differences observed at 28 NFEs are explained by a limited integration budget. The results follow the same overall trend as the main evaluation, with LyapuFlow maintaining strong reconstruction quality as the number of flow evaluations increases.

Table 12: Quantitative evaluation for image inverse problems. All experiments use Stable Diffusion 3.5-Medium with 40 NFEs. Bold: best; Underline: second best.

AFHQ 1k (768 × 768)												
Method	Super-resolution x4 (Bicubic)				Deblurring (Gauss)				Deblurring (Motion)			
	PSNR↑	SSIM↑	FID↓	LPIPS↓	PSNR↑	SSIM↑	FID↓	LPIPS↓	PSNR↑	SSIM↑	FID↓	LPIPS↓
FlowChef	<u>29.67</u>	<u>0.807</u>	22.34	<u>0.077</u>	25.42	<u>0.719</u>	38.55	0.256	<u>26.56</u>	<u>0.743</u>	36.37	0.2107
FlowDPS	28.51	0.764	<u>18.45</u>	0.124	<u>25.67</u>	0.697	<u>29.65</u>	<u>0.244</u>	26.20	0.710	<u>25.21</u>	<u>0.2106</u>
LyapuFlow (Ours)	**30.35**	**0.833**	**8.93**	**0.040**	**27.42**	**0.732**	**17.86**	**0.137**	**27.57**	**0.750**	**17.06**	**0.1429**
FFHQ 1k (768 × 768)												
Method	Super-resolution x4 (Bicubic)				Deblurring (Gauss)				Deblurring (Motion)			
	PSNR↑	SSIM↑	FID↓	LPIPS↓	PSNR↑	SSIM↑	FID↓	LPIPS↓	PSNR↑	SSIM↑	FID↓	LPIPS↓
FlowChef	<u>29.99</u>	<u>0.835</u>	61.02	<u>0.061</u>	25.78	<u>0.763</u>	113.95	0.219	<u>27.13</u>	**0.786**	94.87	0.179
FlowDPS	29.39	0.778	<u>28.45</u>	0.098	<u>26.29</u>	0.739	<u>50.23</u>	<u>0.206</u>	27.09	0.747	<u>40.28</u>	<u>0.173</u>
LyapuFlow (Ours)	**30.28**	**0.836**	**26.48**	**0.044**	**28.04**	**0.770**	**40.20**	**0.122**	**28.21**	<u>0.782</u>	**35.86**	**0.123**
DIV2K 0.8k (768 × 768)												
Method	Super-resolution x4 (Bicubic)				Deblurring (Gauss)				Deblurring (Motion)			
	PSNR↑	SSIM↑	FID↓	LPIPS↓	PSNR↑	SSIM↑	FID↓	LPIPS↓	PSNR↑	SSIM↑	FID↓	LPIPS↓
FlowChef	<u>24.48</u>	<u>0.683</u>	43.25	<u>0.087</u>	<u>20.84</u>	<u>0.536</u>	95.28	<u>0.327</u>	<u>21.94</u>	<u>0.581</u>	80.43	<u>0.263</u>
FlowDPS	24.07	0.637	<u>32.96</u>	0.113	20.81	0.505	<u>89.69</u>	0.334	21.66	0.541	<u>73.33</u>	0.269
LyapuFlow (Ours)	**24.82**	**0.710**	**23.35**	**0.047**	**22.24**	**0.564**	**68.25**	**0.211**	**22.74**	**0.607**	**61.19**	**0.204**

As shown in Fig. 13, full gradients through the velocity field ($k_{\text{sg}} = 0$) yield weaker results across the evaluated metrics. Introducing a suitable initial stop-gradient window ($0 < k_{\text{sg}} < 28$) improves performance, while $k_{\text{sg}} = 28$—which stops velocity gradients throughout and uses decoder-Jacobian-only feedback—also outperforms the full-gradient setting. The best intermediate switch point varies by task and metric.

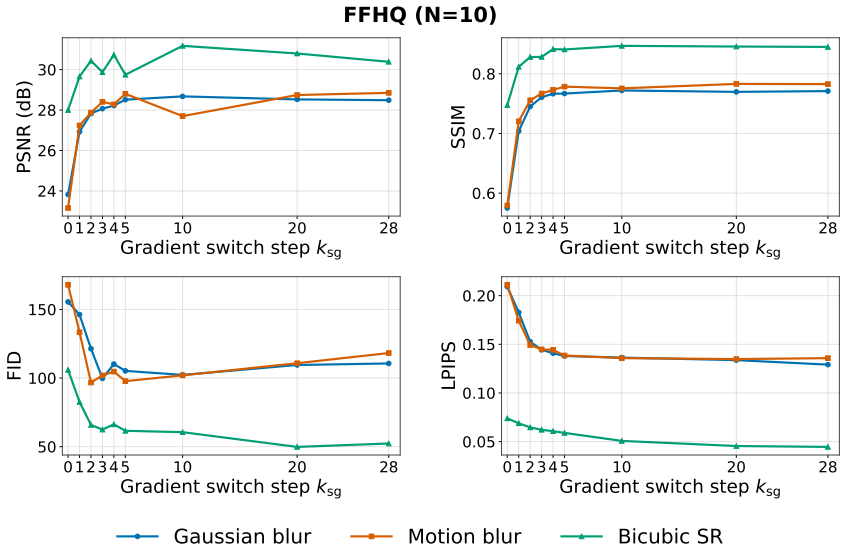

Figure 13: Ablation on gradient switch step k_{sg} on 10 FFHQ images.

C.6 Frequency Analysis

The latent-space formulation uses the decoder-propagated constraint gradient by default and optionally restores the velocity-Jacobian contribution at later sampling steps when finer correction is required. To complement the pixel-space and perceptual metrics, Fig. 14 compares the spectral density of the reconstructions across datasets and inverse problems. This view helps distinguish improvements in spatial-frequency recovery from changes that are visible only through aggregate reconstruction metrics.

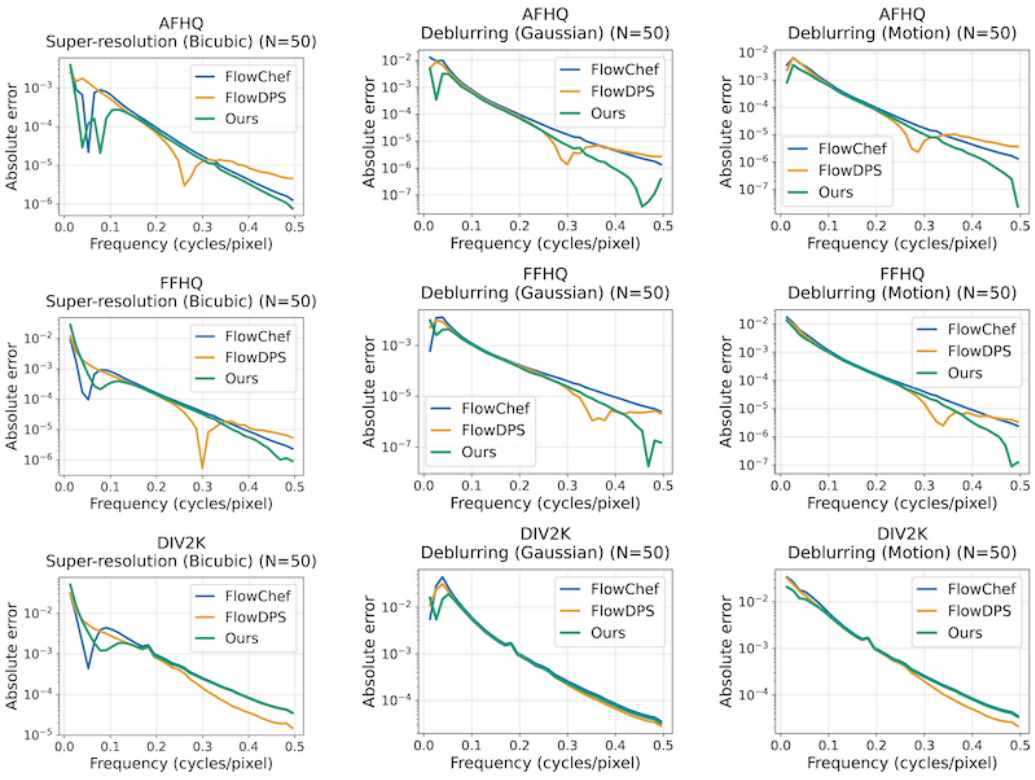

Figure 14: Spectral-density comparisons across datasets and inverse-problem tasks for 28 NFEs.

C.7 Qualitative Results

The following figures provide additional qualitative comparisons across all datasets and inverse problems. Figures 15 to 17 shows the AFHQ results, Figs. 18 to 20 the FFHQ results, and Figs. 21 to 23 the DIV2K results. Across these examples, the reconstructions complement the aggregate metrics by showing how the different guidance methods recover instance-specific structure while satisfying the observation constraint.

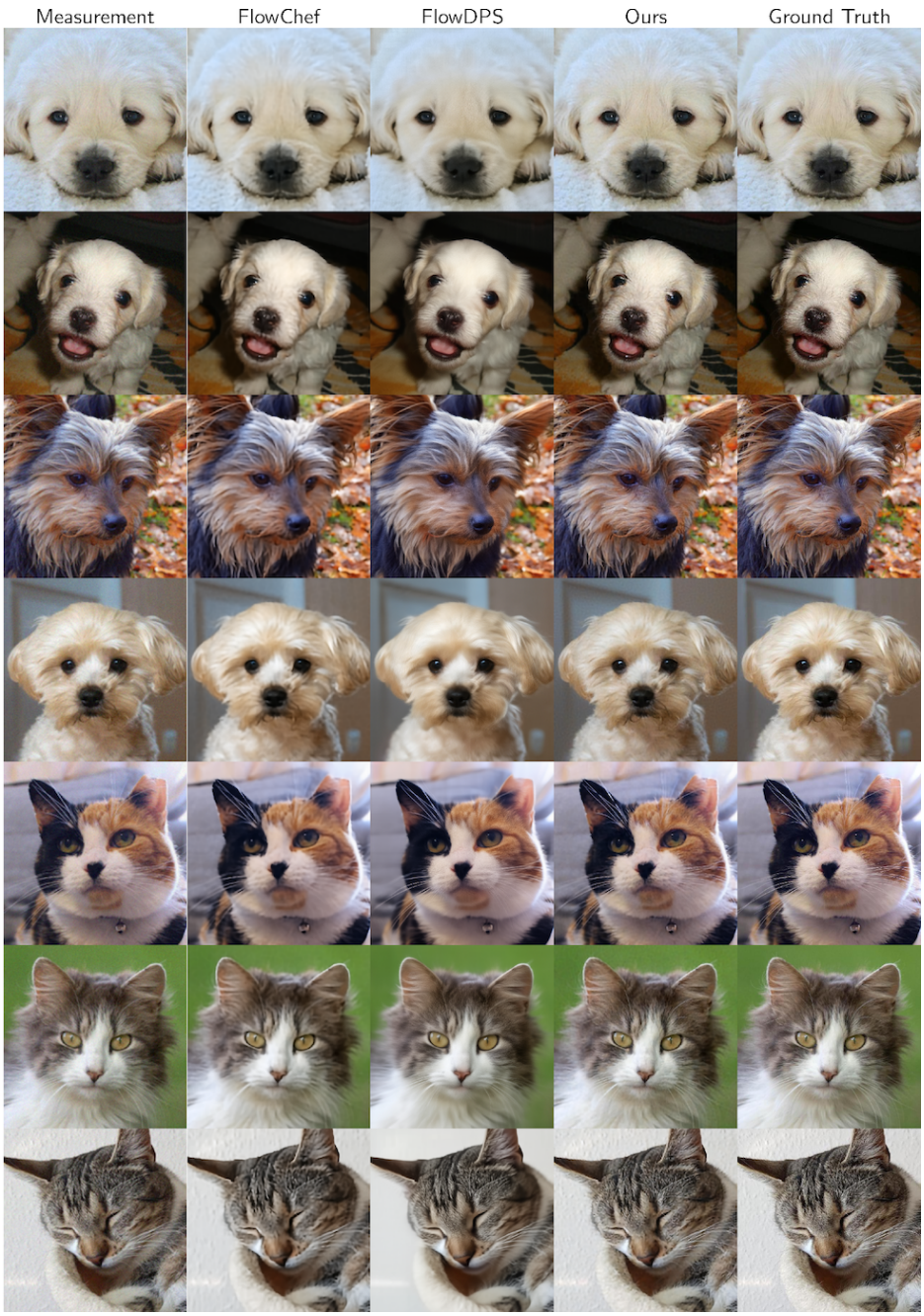

Figure 15: AFHQ bicubic super-resolution using Stable Diffusion 3.5-Medium for 28 NFEs.



Figure 16: AFHQ Gaussian deblurring using Stable Diffusion 3.5-Medium for 28 NFEs.



Figure 17: AFHQ motion deblurring using Stable Diffusion 3.5-Medium for 28 NFEs.



Figure 18: FFHQ bicubic super-resolution using Stable Diffusion 3.5-Medium for 28 NFEs.

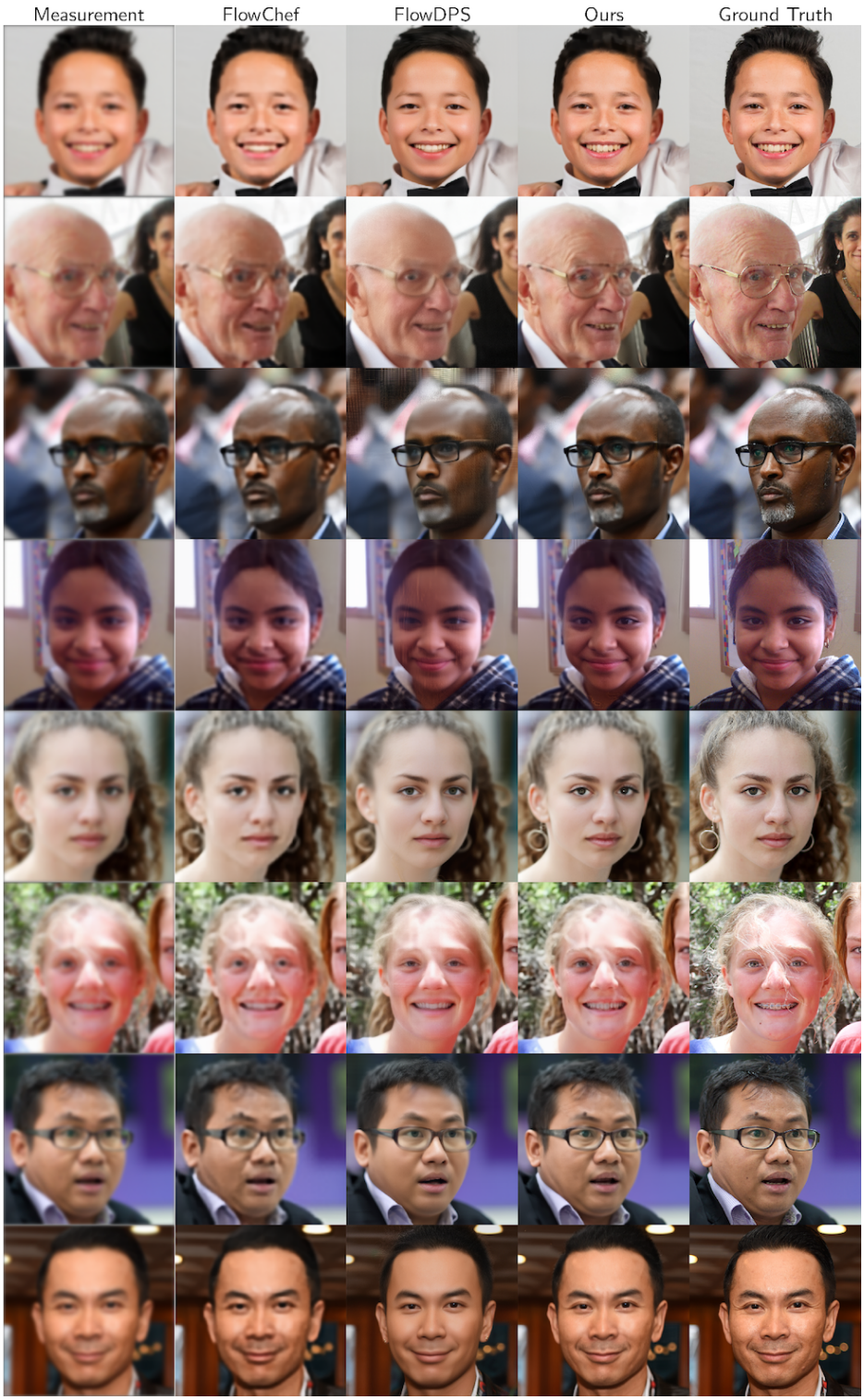

Figure 19: FFHQ Gaussian deblurring using Stable Diffusion 3.5-Medium for 28 NFEs.



Figure 20: FFHQ motion deblurring using Stable Diffusion 3.5-Medium for 28 NFEs.



Figure 21: DIV2K bicubic super-resolution using Stable Diffusion 3.5-Medium for 28 NFEs.



Figure 22: DIV2K Gaussian deblurring using Stable Diffusion 3.5-Medium for 28 NFEs.

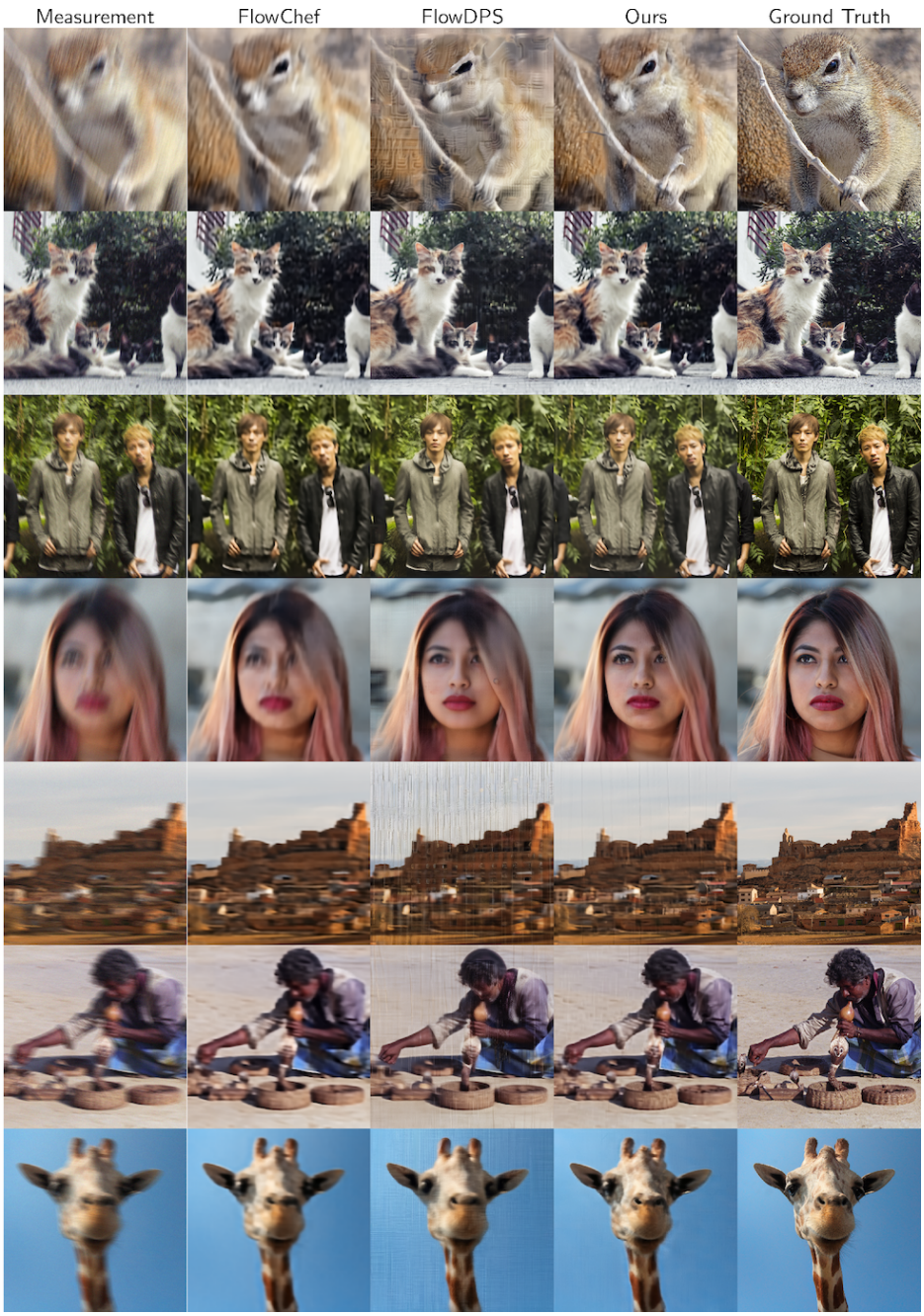

Figure 23: DIV2K motion deblurring using Stable Diffusion 3.5-Medium for 28 NFEs.