SIMULATION-FREE FLOW MAP ALIGNMENT FOR HIGH-FIDELITY ANY-STEP CONSTRAINED GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Flow-based generative models for science and engineering should satisfy physical and operational constraints while preserving distributional fidelity. Inference-time correction improves constraint satisfaction but adds per-sample cost and can distort the generated distribution. Training-time alignment amortizes this cost, but propagating terminal feedback to the local dynamics typically requires costly ODE simulation. Flow map models avoid simulation by evaluating terminal losses directly on learned finite-time transitions. Yet endpoint training alone neither yields the local-velocity update implied by the ODE objective nor ensures that alignment persists when shorter transitions are composed. We introduce simulation-free terminal alignment for any-step flow map generation. A map-based adjoint uses learned intermediate states and the sensitivity of the remaining-time map to transport terminal feedback to the local velocity, approximating the gradient of the ODE terminal objective. Combined with endpoint training and map–velocity consistency, this update aligns a single model across sampling schedules. Our analysis characterizes the map-adjoint approximation error and shows how velocity and consistency errors affect distributional fidelity and cross-schedule agreement. Experiments on constrained distributions, robot trajectories, and physical fields show that the aligned model retains constraint improvements across multi-step schedules while maintaining competitive distributional quality.

1 INTRODUCTION

Generative models for scientific and engineering applications require generated samples—such as physical fields, trajectories, and designs—to be both realistic and valid. Flow-based models can learn the distribution of valid examples, but a new sample may still violate governing equations, boundary conditions, or operating limits. The goal is to improve constraint satisfaction while retaining distributional fidelity and efficient generation.

Constraint-aware or physics-informed generation can intervene at several stages. Reflected dynamics and specialized coordinates encode feasible-set geometry in the generative process (Lou & Ermon, 2023; Li et al., 2026a). Guidance and projection instead correct each sample during inference, adding constraint-dependent computation at deployment (Christopher et al., 2024; Cheng et al., 2025). Training-time methods penalize violations while learning the generator, amortizing constraint handling before deployment (Bastek et al., 2025; Huan et al., 2026). Their central difficulty is propagating a loss on the final sample to the local dynamics that generated it. Pathwise and trajectory-based policy-gradient methods use generated trajectories to obtain terminal feedback (Domingo-Enrich et al., 2025; Liu et al., 2025), making rollout cost a central bottleneck in training. DiffusionNFT instead re-noises scored endpoints for reward-weighted regression, without trajectory-based policy optimization (Zheng et al., 2026).

These costs motivate simulation-free (rollout-free) alignment with flow map models, including Flow Map Matching and MeanFlow, which learn finite-time transitions directly (Boffi et al., 2025b; Geng et al., 2025). They predict endpoints without sequentially integrating a local velocity and generate with only a few map evaluations. The same model can be sampled with different step counts. We use *any-step generation* to mean that the aligned model maintains constraint and quality performance as the sampling schedule changes. Recent work aligns flow maps through endpoint fine-tuning, reward-weighted regression, policy gradients, and inference-time guidance (Mammadov et al., 2026; Huang et al., 2026b; Li et al., 2026b; Potaptchik et al., 2026; Huang et al., 2026a). MeanFlowNFT and

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.

Table 1: Training-based alignment mechanisms and cross-schedule evidence.

Method	Terminal-feedback mechanism	No rollout for feedback[a]	Flexible-step generation[b]	Cross-schedule alignment[c]
DiffusionNFT	Reward-weighted velocity regression	–	–	–
Adjoint Matching	Trajectory adjoint for local velocity	–	–	–
MFM-FT	Posterior-weighted steered-drift update	✓	–	–
FAV	Stein particle update regression	✗	✓	–
VFM	Endpoint gradient through map and noise adapter	✓	✓	–
RMMD	One-step reward gradient to denoiser	–	✓	–
Flow-Map GRPO	Trajectory policy gradient for stochastic map	–	✓	✓
MeanFlowNFT	Reward-weighted induced-velocity regression	–	✓	✓
Ours	Map-adjoint velocity update + direct endpoint loss	✓	✓	✓

✗ indicates partial or configuration-dependent support; see Section 2.
[a] No sequential model rollout is needed to collect terminal-feedback samples.
[b] After alignment, the model generates in one or a few learned steps without further distillation.
[c] Terminal-objective gains are reported at multiple inference step counts for one aligned model.

Flow-Map GRPO report gains across several step counts, while VFM finds that one-step alignment can weaken under composition (Huang et al., 2026b; Li et al., 2026b; Mammadov et al., 2026).

Together, these advances show the promise of reward-aligned flow maps, but do not establish any-step terminal constraint alignment with one trained map while avoiding ODE rollouts during training. We use learned-map predictions to approximate the ODE adjoint and update the local velocity, directly train the one-step map with the terminal penalty, and use velocity–map consistency to align finite-time transitions with the updated dynamics. Our contributions are:

▷ **Simulation-free local-dynamics alignment.** We use learned finite-time maps to estimate the intermediate states and terminal-gradient transport required by the ODE adjoint. This supplies terminal feedback to the local velocity with a fixed number of map evaluations and derivative operations, independent of numerical ODE solver rollout.

▷ **Any-step constraint alignment.** We couple the adjoint update with velocity–map consistency so that finite transitions track the aligned local dynamics. Direct terminal training also penalizes violations of the one-step output. This design targets constraint alignment that remains effective when the sampling step count changes during inference.

▷ **From local alignment to cross-schedule performance.** We bound the error of map-based terminal feedback and derive bounds on distribution error and terminal-loss differences across sampling schedules. These results separate flow-matching excess risk from finite-map error and show how map–velocity consistency controls disagreement among finite-step schedules.

We evaluate the resulting method on constrained distributions, robot trajectories, and physical fields, comparing any-step constraint satisfaction and distributional quality across sampling schedules.

2 Related Work

Constrained generation. Existing methods encode feasible-set geometry in the dynamics (Lou & Ermon, 2023; Li et al., 2026a), correct samples by inference-time guidance or projection (Christopher et al., 2024; Cheng et al., 2025), or incorporate constraints into training through penalties or differentiable projection (Bastek et al., 2025; Huan et al., 2026; Chen et al., 2026; Christopher et al., 2026). We train with a differentiable terminal penalty and assess both violation and distributional fidelity. This leaves two questions: how endpoint feedback updates the generator, and whether its effect persists across step counts. Table 1 summarizes the comparison.

Reward alignment. Terminal alignment first requires generated endpoints and a way to turn their scores into updates. DiffusionNFT re-noises scored endpoints for velocity regression without storing trajectories, though collecting the endpoints still requires model rollouts (Zheng et al., 2026). Adjoint Matching uses trajectory costates for local control targets (Domingo-Enrich et al., 2025). FAV amortizes Stein updates into different generators, including flow maps, so its sampling cost depends on the backbone (Lee et al., 2026a). RMMD differentiates the reward through one denoising prediction after few-step sampling (Jacq et al., 2026). Thus a short gradient path does not remove the cost of collecting terminal feedback.

Learned flow maps provide direct finite-time predictions and further alignment mechanisms. Flow-Map GRPO trains on stochasticized map trajectories (Li et al., 2026b). Meta Flow Maps use

posterior maps for inference-time steering and off-policy drift fine-tuning; reward-tilted few-step map distillation is a separate proposal (Potaptchik et al., 2026). Diamond Maps use stochastic maps for inference-time reward search (Holderrieth et al., 2026); FMRG guides at inference with a remaining-time map Jacobian (Huang et al., 2026a). These works show how terminal signals act through a learned map. The second question arises because that map can be called once or composed over subintervals, so its alignment must persist across schedules.

Few/Any-step generation and schedule consistency. This schedule question connects alignment to the broader few/any-step literature. Few-step models emphasize small sampling budgets; any-step models seek one model that performs across schedules. FMM, Shortcut Models, MeanFlow, CTM, TiM, and AnyFlow learn interval transitions, primarily evaluating unconstrained generation quality (Boffi et al., 2025b; Frans et al., 2025; Geng et al., 2025; Kim et al., 2024; Wang et al., 2026; Gu et al., 2026). With a terminal objective, CTM and stochastic Itô maps guide inference (Kim et al., 2024; Pan et al., 2026). VFM finds that one-step reward gains weaken under composition as samples regress toward the unconditional distribution (Mammadov et al., 2026). MeanFlowNFT and Flow-Map GRPO report gains at several step counts (Huang et al., 2026b; Li et al., 2026b), whereas RMMD trains separate few-step regimes (Jacq et al., 2026). These results show that reward alignment can transfer across some step counts, but leave open how to train a single flow map for terminal constraints without ODE rollouts while preserving the gains across schedules. We address this gap by combining map-based terminal feedback for the local dynamics with consistency training for the finite-time transitions.

3 Flow Matching with Terminal Constraints

We formalize the terminal-constrained flow-matching objective and its training bottleneck. Let $Z \sim p_0$, $X \sim p_{\mathrm{data}}$, and $t \sim \mathrm{Unif}[0,1]$. Along the interpolation $X_t = (1-t)Z + tX$, flow matching learns a velocity v_θ through

$$\mathcal{L}_{\mathrm{fm}}(v_\theta) = \mathbb{E}_{Z,X,t}\left[\|v_\theta(X_t,t) - (X-Z)\|^2\right]. \tag{1}$$

The ODE $\dot{Y}_t = v_\theta(Y_t,t)$, $Y_0 = Z$, induces a flow $Y_t = \phi^\theta_{0,t}(Z)$ and terminal sample $Y_1 = \phi^\theta_1(Z)$.

The target requirement is that generated endpoints lie in $\mathcal{C} = \{y : g(y) \le 0,\ h(y) = 0\}$, where $g : \mathbb{R}^d \to \mathbb{R}^{m_g}$ and $h : \mathbb{R}^d \to \mathbb{R}^{m_h}$; intermediate states need not be feasible. Because approximation, optimization, and finite-step sampling errors can move generated endpoints outside $\mathcal{C}$, we measure violation with a differentiable penalty $\rho : \mathbb{R}^d \to \mathbb{R}_+$, for example

$$\rho(y) = \|[g(y)]_+\|_2^2 + \|h(y)\|_2^2, \tag{2}$$

where $[\cdot]_+$ acts componentwise.

Training-time alignment augments distribution learning with this terminal signal:

$$\mathcal{J}(\theta) = \mathcal{L}_{\mathrm{fm}}(v_\theta) + \lambda_{\mathrm{vio}}\mathbb{E}_Z\left[\rho(\phi^\theta_1(Z))\right], \qquad \lambda_{\mathrm{vio}} \ge 0. \tag{3}$$

Training challenge. The terminal violation loss in (3) is defined on the endpoint of the ODE flow, so updating the local velocity requires propagating its gradient through the dynamics. Pathwise differentiation requires a numerical ODE trajectory (Domingo-Enrich et al., 2025), while trajectory-based policy gradients use sampled trajectories (Liu et al., 2025; Li et al., 2026b). DiffusionNFT instead uses scored endpoints for forward-process regression (Zheng et al., 2026). Reducing the number of rollout steps lowers this cost but introduces endpoint discretization error (Benton et al., 2024; Lu et al., 2022) and ties the resulting update to the chosen discretization. Learned flow maps offer direct endpoint predictions, but training only the full-interval map does not supply the ODE terminal objective's local-velocity update or ensure that alignment persists when shorter transitions are composed. We therefore seek to use learned maps to align the underlying ODE dynamics without trajectory simulation and retain that improvement across sampling schedules.

4 Simulation-Free Terminal Alignment

Figure 1 shows how the shared flow map and local velocity work together. Consistency training ties finite-time transitions to the velocity, while a map-based adjoint carries terminal feedback back to that velocity without an ODE rollout. We combine this update with direct terminal training of the map to support stable performance across sampling schedules.

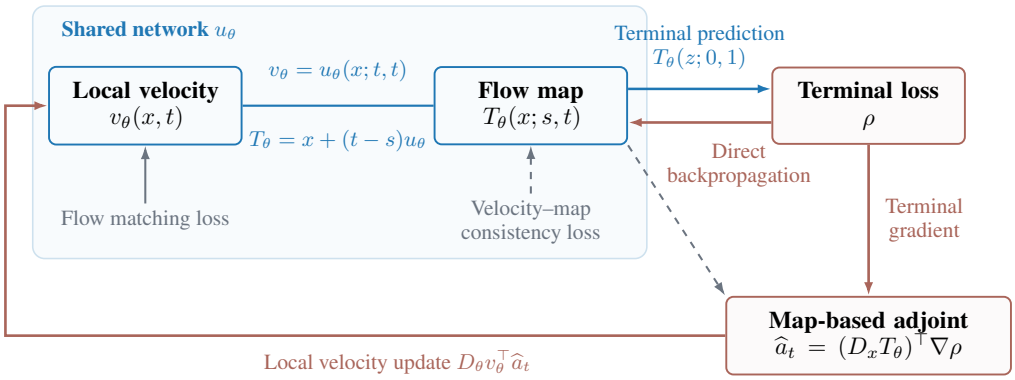


Figure 1: Terminal alignment with a shared velocity and flow map. The adjoint uses $D_x T_\theta(\widehat{x}_t; t, 1)$ and $\nabla\rho(\widehat{y}_t)$ to transfer terminal feedback to the local velocity. The gray dashed arrow supplies the map state and VJP. Velocity–map consistency couples few-step transitions to the updated dynamics.

4.1 Flow Maps across Sampling Schedules

Flow maps predict the transition from time s to time t directly, rather than repeatedly evaluating a local velocity, as in Flow Map Matching, Shortcut Models, and MeanFlow (Boffi et al., 2025a;b; Frans et al., 2025; Geng et al., 2025). We use a shared network u_θ to represent both the finite-time transition and the local velocity:

$$T_\theta(x; s, t) = x + (t - s)u_\theta(x; s, t), \qquad v_\theta(x, t) = u_\theta(x; t, t). \tag{4}$$

For a state x at time s, $u_\theta(x; s, t)$ predicts its normalized displacement over $[s, t]$; evaluating the same network at $s = t$ defines the local velocity. For $x = X_s = (1 - s)Z + sX$ and $s < t$, we adopt the Lagrangian self-distillation (LSD) loss from Flow Map Matching (Boffi et al., 2025b) to train the finite-time map alongside this local velocity:

$$\mathcal{L}_{\text{lsd}} = \mathbb{E}_{Z,X,s,t}\Big[\|\partial_t T_\theta(x; s, t) - \text{sg}[v_\theta(T_\theta(x; s, t), t)]\|^2\Big]. \tag{5}$$

Here sg stops gradients through its argument; The forward loss penalizes the map–velocity residual $r_\theta(x; s, t) = \partial_t T_\theta(x; s, t) - v_\theta(T_\theta(x; s, t), t)$. When this residual vanishes along a transition, the map reproduces the ODE flow of v_θ on that transition (Boffi et al., 2025b, Proposition 2.2).

After training, the direct map $T_\theta(Z; 0, 1)$ predicts the sample in one network evaluation. It can also generate with multiple shorter transitions along a schedule $\mathcal{P} = (0 = t_0 < \cdots < t_J = 1)$:

$$\widehat{X}_0 = Z, \qquad \widehat{X}_{j+1} = T_\theta(\widehat{X}_j; t_j, t_{j+1}), \qquad G_{\theta,\mathcal{P}}(Z) = \widehat{X}_J. \tag{6}$$

Here $\widehat{X}_j$ is the generated state after j transitions, $G_{\theta,\mathcal{P}}$ is their composition, and J is the number of map evaluations. The same map thus supports different inference budgets.

Direct endpoint alignment. The terminal objective in (3) is defined on the endpoint of the ODE flow. Thus, a natural shortcut is to apply the same penalty directly to the one-step map output:

$$\mathcal{L}^{\text{ode}}_{\text{vio}}(\theta) = \mathbb{E}_Z\big[\rho(\phi^\theta_1(Z))\big] \quad \longrightarrow \quad \mathcal{L}^{\text{map}}_{\text{vio}}(\theta) = \mathbb{E}_Z[\rho(T_\theta(Z; 0, 1))]\,. \tag{7}$$

Training on the approximated endpoint propagates this penalty through the full-interval flow map $T_\theta(\cdot; 0, 1)$. However, this gradient need not match the local-velocity update implied by the target ODE terminal objective. The loss also supervises the full-interval transition directly; whether its improvement persists under composed shorter transitions depends on map–velocity consistency. We next use learned maps to approximate the ODE local-velocity update.

4.2 Map-Based Terminal Adjoint

Changing the velocity at time t affects the final loss through the remaining trajectory. Classical optimal control and adjoint-sensitivity analysis express this dependence through a backward-propagated terminal covector (Pontryagin et al., 1962; Giles & Pierce, 2000; Chen et al., 2018).

Lemma 1 (Continuous-adjoint identity (Pontryagin et al., 1962)). *For a fixed initial state $z \in \mathbb{R}^d$, let $x^\theta(t) = \phi^\theta_{0,t}(z)$. Under the regularity conditions in Appendix B.3.1, with $t \sim \mathrm{Unif}[0,1]$,*

$$a^\theta(z,t) = D_x \phi^\theta_{t,1}(x^\theta(t))^\top \nabla \rho(\phi^\theta_1(z)), \tag{8}$$

$$\nabla_\theta \mathbb{E}_Z\big[\rho(\phi^\theta_1(Z))\big] = \mathbb{E}_{Z,t}\big[D_\theta v_\theta(\phi^\theta_{0,t}(Z),t)^\top a^\theta(Z,t)\big]\,. \tag{9}$$

The proof is given in Appendix B.3.1. This identity expresses the gradient of the original terminal objective as local velocity updates along the trajectory. Evaluating it requires the intermediate state and remaining-time sensitivity, which conventional adjoint methods obtain from expensive forward trajectory rollout and backward sensitivity solve (Domingo-Enrich et al., 2025; Shin et al., 2026). We approximate both with learned finite-time flow maps.

Flow-map approximation. The 0-to-t map predicts the intermediate state, while the Jacobian of the remaining t-to-1 map transports the terminal gradient. Write $\mathsf{J}^\theta_{s,t}(x) := D_x T_\theta(x;s,t)$. For $z \sim p_0$ and $t \sim \mathrm{Unif}[0,1]$, define

$$\widehat{x}_t = \mathrm{sg}[T_\theta(z;0,t)], \quad \widehat{y}_t = T_\theta(\widehat{x}_t;t,1), \quad \widehat{a}_t = \mathrm{sg}\big[\mathsf{J}^\theta_{t,1}(\widehat{x}_t)^\top \nabla\rho(\widehat{y}_t)\big]\,. \tag{10}$$

Thus $\widehat{y}_t$ is the terminal state obtained by composing the two map predictions, and $\widehat{a}_t$ transports its terminal gradient back to time t. Stop-gradient treats $\widehat{x}_t$ and $\widehat{a}_t$ as fixed targets, so the retained parameter-gradient contribution is $D_\theta v_\theta(\widehat{x}_t,t)^\top \widehat{a}_t$.

The map-adjoint error bound in (18) (Appendix B.3.1) separates the error in the local-velocity update into an LSD-residual term for state and endpoint accuracy and a remaining-time Jacobian term for gradient transport. LSD penalizes the residual at states sampled along the reference path but does not directly constrain the Jacobian error. Nevertheless, Figure 2 shows that most evaluated map adjoints achieve low terminal-state prediction error and positive cosine similarity with a numerically computed ODE adjoint (see Appendix D.1 for details). These results support using the map adjoint as approximate terminal feedback for the local velocity.

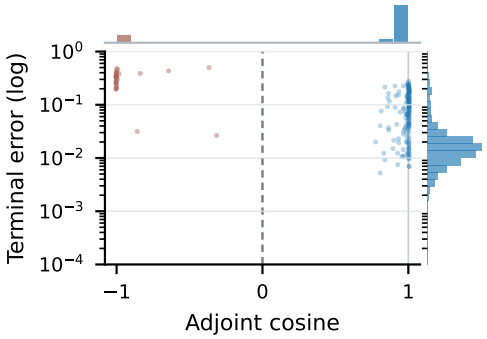


Figure 2: **Empirical accuracy of the map-based adjoint**. Margins show cosine similarity (top) and terminal prediction error (right); red marks negative cosine.

4.3 Joint Alignment for Any-Step Performance

We combine the two terminal updates in (7) with LSD on the shared model. The map adjoint approximates $\nabla_\theta \mathcal{L}^{\mathrm{ode}}_{\mathrm{vio}}$ through the local velocity, $\mathcal{L}^{\mathrm{map}}_{\mathrm{vio}}$ directly trains the one-step output, and LSD aligns finite-time transitions with the updated local dynamics.

Applying adjoint feedback. To apply the approximate local-velocity direction through automatic differentiation, we construct a zero-forward-value surrogate. For minibatch pairs (z_i,t_i), let $\widehat{x}_i$ and $\widehat{a}_i$ denote the quantities in (10), and define $\Psi_{\mathrm{adj}} = B^{-1}\sum_{i=1}^B \langle \widehat{a}_i, v_\theta(\widehat{x}_i,t_i)\rangle$. We then set

$$\mathcal{L}_{\mathrm{adj}} = \Psi_{\mathrm{adj}} - \mathrm{sg}[\Psi_{\mathrm{adj}}], \qquad \nabla^{\mathrm{AD}}_\theta \mathcal{L}_{\mathrm{adj}} = \frac{1}{B}\sum_{i=1}^B D_\theta v_\theta(\widehat{x}_i,t_i)^\top \widehat{a}_i. \tag{11}$$

The superscript AD denotes the gradient produced by automatic differentiation under stop-gradient. Although $\mathcal{L}_{\mathrm{adj}}$ is zero in the forward pass, its backward gradient applies the map-adjoint approximation of $\nabla_\theta \mathcal{L}^{\mathrm{ode}}_{\mathrm{vio}}$ through $v_\theta(\widehat{x}_i,t_i)$. Thus, $\mathcal{L}_{\mathrm{adj}}$ is used only to implement the approximate local-velocity update for $\mathcal{L}^{\mathrm{ode}}_{\mathrm{vio}}$; it does not add another forward violation penalty.

Keeping the FM term from (3), the implemented surrogate loss function is

$$\boxed{\widetilde{\mathcal{L}}_{\mathrm{train}} = \mathcal{L}_{\mathrm{fm}} + \lambda_{\mathrm{adj}}\mathcal{L}_{\mathrm{adj}} + \lambda_{\mathrm{vio}}\mathcal{L}^{\mathrm{map}}_{\mathrm{vio}} + \lambda_{\mathrm{lsd}}\mathcal{L}_{\mathrm{lsd}}}. \tag{12}$$

Here $\mathcal{L}_{\mathrm{adj}}$ approximates its ODE terminal gradient, while $\mathcal{L}^{\mathrm{map}}_{\mathrm{vio}}$ adds an auxiliary direct-map update. Any-step transfer further requires the map–velocity residual $\mathcal{L}_{\mathrm{lsd}}$ to remain small on states visited by deployed compositions, as formalized in Section 5.

Simulation-free computation. Computing $\mathcal{L}_{\mathrm{adj}}$ uses a fixed number of learned map evaluations and derivative contractions, without simulating or differentiating through an ODE trajectory. Its cost is therefore independent of numerical ODE rollout length, while pathwise differentiation depends on solver evaluations, and policy-gradient training depends on the number and length of sampled trajectories. Appendix B.2 gives the complete algorithm and forward/backward cost comparison.

5 Distribution and Cross-Schedule Error Bounds

The joint objective in (12) aligns the local dynamics and finite-time maps. We now ask when compositions of these maps retain the distributional and terminal-loss behavior of the aligned ODE. The bounds separate error in the learned local velocity from error in realizing its flow with finite transitions.

Assume that the learned and reference flows are well posed, the learned velocity is Lipschitz along the compared trajectories, and the finite maps have square-integrable map–velocity residuals. The reference FM flow is assumed to realize the data marginal path. Precise conditions and the stability constants C_θ and Λ_θ are given in Appendix B.3.2.

Theorem 2 (Distribution error and cross-schedule consistency)**.** *Under these conditions, let $v_{\mathrm{fm}}(x,t) = \mathbb{E}[X - Z \mid X_t = x, t]$ be the unrestricted population minimizer of the reference flow-matching objective and define the excess FM risk $\Delta_{\mathrm{fm}}(\theta) = \mathcal{L}_{\mathrm{fm}}(v_\theta) - \mathcal{L}_{\mathrm{fm}}(v_{\mathrm{fm}})$. For $p_{\theta,\mathcal{P}} = (G_{\theta,\mathcal{P}})_{\#}p_0$, let $\mathcal{E}_{\mathrm{lsd}}(\theta;\mathcal{P})$ denote the squared energy of the LSD residual $r_\theta(x;s,t) = \partial_t T_\theta(x;s,t) - v_\theta(T_\theta(x;s,t),t)$ along the composition defined by $\mathcal{P}$ (detailed in Appendix B.3.2). Then*

$$W_2(p_{\theta,\mathcal{P}}, p_{\mathrm{data}}) \le \sqrt{C_\theta \Delta_{\mathrm{fm}}(\theta)} + e^{\Lambda_\theta}\sqrt{\mathcal{E}_{\mathrm{lsd}}(\theta;\mathcal{P})}. \tag{13}$$

For any two finite schedules $\mathcal{P}$ and $\mathcal{Q}$, if ρ is G_ρ-Lipschitz, then

$$|\mathbb{E}[\rho(G_{\theta,\mathcal{P}}(Z))] - \mathbb{E}[\rho(G_{\theta,\mathcal{Q}}(Z))]| \le G_\rho e^{\Lambda_\theta}\left(\sqrt{\mathcal{E}_{\mathrm{lsd}}(\theta;\mathcal{P})} + \sqrt{\mathcal{E}_{\mathrm{lsd}}(\theta;\mathcal{Q})}\right). \tag{14}$$

The first bound decomposes distribution error into local-velocity error, measured by the excess FM risk Δ_{fm}, and finite-map error, measured by the map–velocity residual energy. These are the quantities targeted by $\mathcal{L}_{\mathrm{fm}}$ and $\mathcal{L}_{\mathrm{lsd}}$, respectively. The second bound compares two schedules through the same learned ODE and shows that their terminal-loss disagreement is governed by their finite-map errors; choosing $\mathcal{Q} = (0,1)$ gives the comparison with the direct one-step map. Together, the bounds explain how flow matching preserves distribution fidelity while consistency training promotes any-step performance. Appendix B.3.2 further applies the cross-schedule bound to common quadratic violation penalties and extents the distribution bound in the latent generative modeling setting.

6 Experiments

6.1 Setup and Comparisons

The main question is whether terminal alignment learned once remains useful when the same flow map is sampled directly or through shorter transitions. We evaluate each trained checkpoint at $K \in \{1, 4, 16, 64\}$ without schedule-specific retraining. Six Disks tests a distribution supported on six disconnected feasible regions; SMD9 generates conditioned trajectories for nine robots; Dynamic Stall generates conditioned physical fields subject to ideal-gas and wall-shear residuals. We report sliced Wasserstein distance (SWD) to reference outputs alongside mean distance outside the feasible set, collision penetration, and mean excess physical residual, respectively. SMD9 provides one reference trajectory per task, so its SWD measures proximity to that trajectory rather than recovery of a conditional distribution. For high-dimensional SMD9 and Dynamic Stall, we train and sample the generator in a proper orthogonal decomposition (POD) latent space (Sirovich, 1987) with a frozen encoder and decoder. Generated latents are decoded into trajectories or fields before evaluation, and terminal-loss gradients pass through the decoder during training.

The baselines distinguish where terminal feedback enters. FMM–LSD is the unaligned flow-map parent. VFM jointly updates a map and source adapter (Mammadov et al., 2026), and FM-RE trains a randomized late-stage flow with terminal rewards (Huan et al., 2026). ODE-Adjoint backpropagates through a numerical trajectory, whereas Flow-GRPO estimates a policy gradient from sampled

trajectories (Chen et al., 2018; Liu et al., 2025). FMRG-J guides a frozen FMM–LSD map during sampling (Huang et al., 2026a); FM+Proj instead projects after each step of a frozen FM sampler.

The trained alignment methods start from their matching parent checkpoints, use the same benchmark terminal penalty, and receive 10,000 post-training updates; their internal objectives remain method-specific. Rollout-based baselines use eight internal steps per training rollout. FMRG-J uses the shared smoothed penalty and a guidance scale of 1 at every K; FM+Proj uses exact set projection for Six Disks and eight-step Lagrangian corrections for SMD9 and Dynamic Stall. We use paired evaluation samples for three training seeds and report the mean over seeds. Further settings, seed variability, and results on Checkerboard, SMD6, and Stokes appear in Appendices C and D.1.

6.2 Cross-Schedule Performance

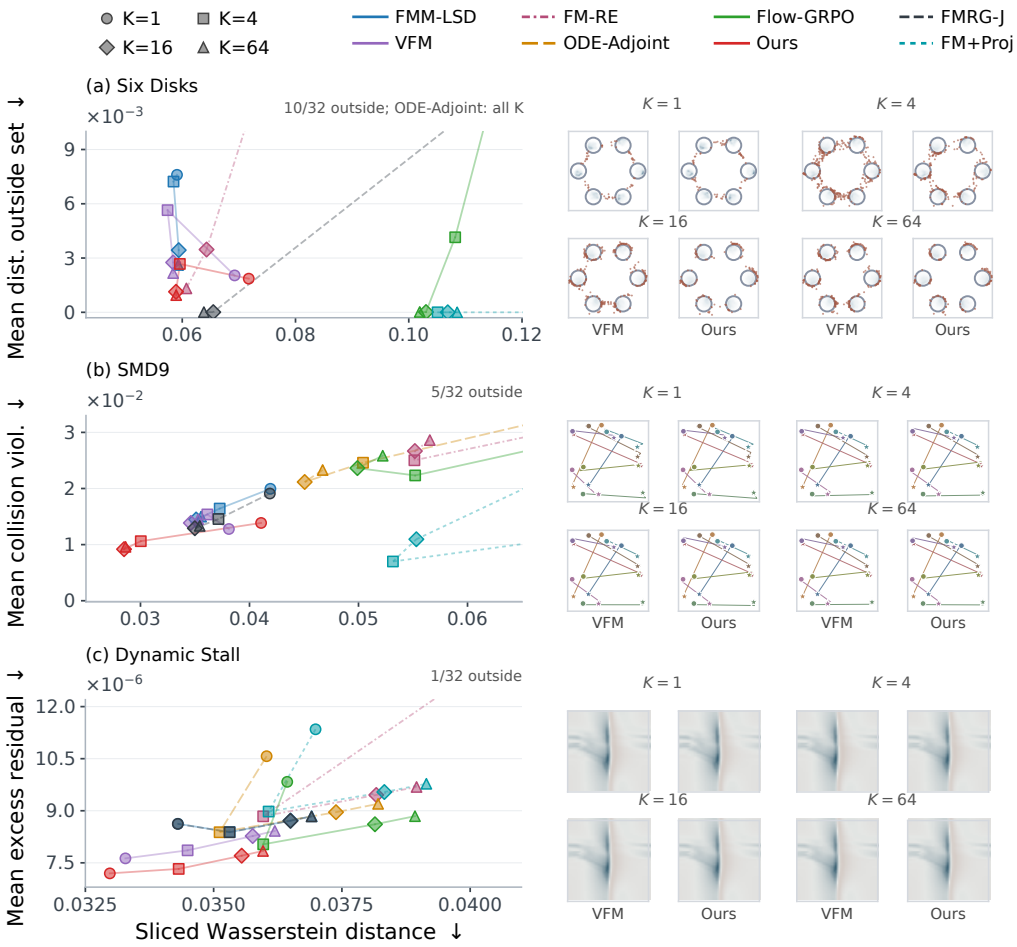


Figure 3: **Cross-schedule quality and terminal error.** Left: three-seed mean SWD versus the benchmark-specific terminal metric at $K \in \{1, 4, 16, 64\}$ (lower left is better). Shapes denote K; color and line style denote methods, with lines joining schedules of the same method. Counts indicate means outside the plotted range; their lines continue to the plot edge. Right: paired VFM and Ours outputs at all four schedules, arranged as $K = 1, 4$ above $K = 16, 64$. Six Disks shows sample densities (warm dots lie outside the set), SMD9 a nine-robot trajectory, and Dynamic Stall normalized pressure fields. Each benchmark uses the same spatial bounds or color scale across methods and schedules.

Figure 3 tests whether terminal alignment learned once retains a useful constraint–quality trade-off as K changes. Relative to the unaligned FMM–LSD parent, our method lowers the mean terminal metric at every tested K on all three tasks, while the SWD trade-off varies. On Six Disks at $K = 4$, for example, mean out-of-set distance falls from 7.2 to 2.7 in units of 10^{-3}, while SWD changes only

from 0.0585 to 0.0596. The one-step trade-off is less favorable: the violation falls, but SWD rises from 0.0591 to 0.0718. Thus improvement at the endpoint alone does not summarize performance when the same map is composed.

For $K = 4, 16, 64$, our method also has lower terminal error than the flow-map-alignment baseline VFM on all three benchmarks. On SMD9 it improves both axes: at $K = 16$, collision penetration and SWD fall from $(0.0139, 0.0346)$ for VFM to $(0.0092, 0.0285)$ for our method. On Dynamic Stall, our method lowers mean excess residual at all four tested step counts, while SWD remains similar to VFM. At $K = 16$, the excess residual falls from 8.27 to 7.71 in units of 10^{-6}, with SWD changing from 0.0358 to 0.0355. Both methods remain above the stated residual tolerance.

The alternative feedback mechanisms trace different trade-offs. Flow-GRPO nearly removes Six Disks violation at $K = 16, 64$, but its SWD is about 0.10, versus about 0.059 for our method; ODE-Adjoint is highly variable there. On SMD9 at composed schedules, our method has lower mean SWD and collision penetration than both rollout-trained baselines. Inference-time correction can drive violation lower still: FM+Proj yields zero measured Six Disks violation, but its SWD exceeds 0.10 for $K \geq 4$; FMRG-J reaches very low violation at $K = 16, 64$ but has large errors at $K = 1, 4$ with its fixed guidance scale. On SMD9, FM+Proj lowers collision penetration at $K = 4$ but increases SWD and deteriorates at $K = 64$. These comparisons show why constraint error must be read together with sample quality and sampling schedule, rather than ranking methods by violation alone.

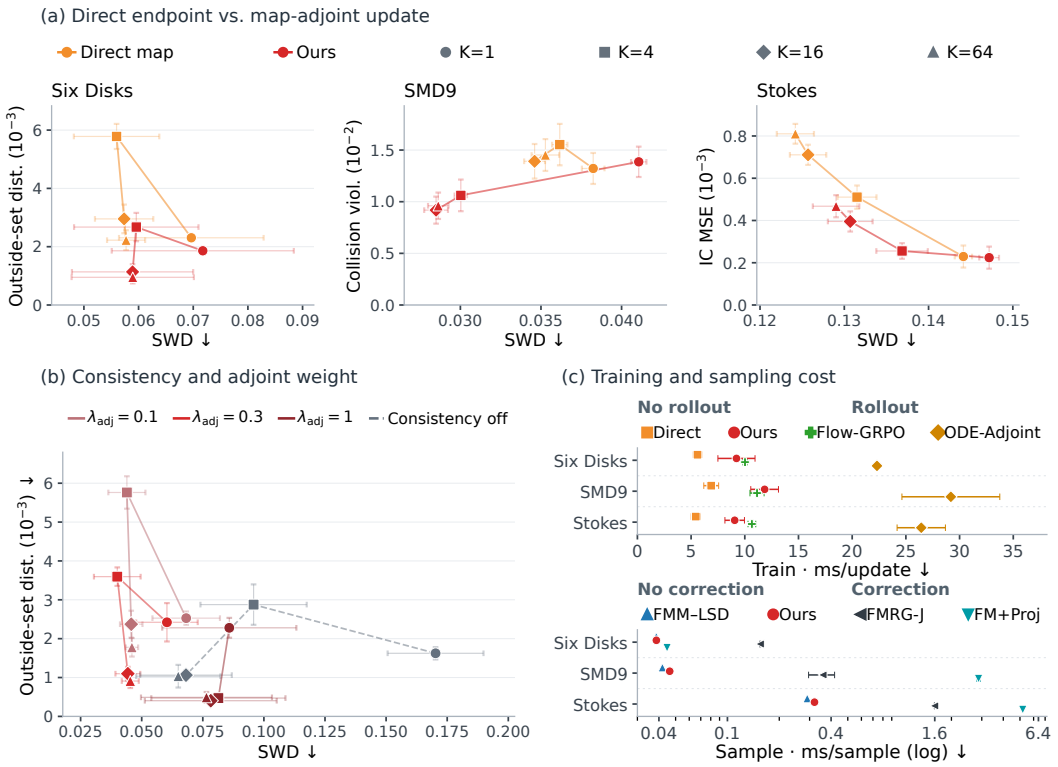


Figure 4: **Endpoint alignment and computation.** (a) Direct map and Ours on Six Disks, SMD9, and Stokes. SWD is on the horizontal axis; Stokes uses initial-condition MSE as its terminal metric. (b) Six Disks validation comparison across adjoint weights; the dashed curve omits velocity–map consistency at $\lambda_{\text{adj}} = 0.3$. In (a,b), shapes denote $K \in \{1, 4, 16, 64\}$, lines join schedules of one method, and points and bars show three-seed means and standard deviations. (c) H200 time per optimizer update and per sample at $K = 4$; the sampling axis is logarithmic.

6.3 Alignment and Cost

Figure 4 tests the roles and costs of the two training components: the map-adjoint update improves composed-schedule terminal error, velocity–map consistency helps preserve Six Disks sample quality, and updates are faster than ODE-Adjoint.

Figure 4(a) uses Direct map as an ablation: it retains flow matching and velocity–map consistency but backpropagates the terminal loss only through $T_\theta(Z; 0, 1)$, without the map-adjoint update. Because the map and velocity share parameters, this direct loss can still change the velocity; the comparison isolates the added explicit map-adjoint update. Stokes tests an initial-condition constraint, complementing the Dynamic Stall physical-residual task in Figure 3. Direct is competitive at $K = 1$ and slightly better on both SMD9 metrics there. At $K = 4, 16, 64$, however, our method has lower terminal error on all three benchmarks. On SMD9 the composed-schedule gain also comes with lower SWD, whereas Stokes again trades SWD for lower initial-condition error.

Figure 4(b) examines two choices within our method on Six Disks validation data. Raising the adjoint weight λ_{adj} from 0.1 to 1 lowers violation at the composed schedules, but the largest weight also raises SWD. At fixed $\lambda_{\mathrm{adj}} = 0.3$, removing velocity–map consistency increases one-step SWD from 0.060 to 0.170 and four-step SWD from 0.040 to 0.096. Violation can be lower without this loss, especially at small K; the ablation therefore supports its role in preserving distributional quality, rather than treating it as another direct violation penalty. A separate diagnostic on frozen, pre-alignment flow maps in Appendix D.1 finds that map-transported terminal gradients have median cosine 0.994 on Six Disks and 0.972 on Checkerboard against the ODE adjoint when both are nonzero. This supports approximate gradient transport on frozen maps; the diagnostic does not measure the parameter gradient used during training.

Figure 4(c) shows 9–12 ms per optimizer update for our method on one H200, versus 22–29 ms for ODE-Adjoint and 10–11 ms for Flow-GRPO. Our update is faster than ODE-Adjoint here and comparable in cost to Flow-GRPO. At $K = 4$, sampling remains close to FMM–LSD, while FMRG-J and FM+Proj incur correction cost at each step, especially on SMD9 and Stokes. Appendix C gives the timing boundaries.

7 Conclusion and Limitations

We introduced simulation-free terminal alignment for flow-map generative models. A learned flow map supplies the intermediate states and remaining-time sensitivity needed to convert a terminal loss into an approximate update of the local ODE velocity. Direct endpoint training and map–velocity consistency then couple this update to the finite transitions used under different sampling schedules. The analysis identifies the transition and sensitivity errors that affect the map adjoint and relates the map–velocity residual to distributional fidelity and cross-schedule terminal-loss agreement. Together, these components provide a unified route to terminal alignment and any-step generation without an ODE rollout during training.

The main limitation is the accuracy of the flow-map Jacobian used to transport terminal feedback. LSD trains the map transitions but does not directly control this sensitivity; despite the directional agreement observed empirically, improving its accuracy remains an important direction. Another direction is to combine learned alignment with an inference-time projection layer for end-to-end hard feasibility. Alignment can reduce the correction required at inference, while projection provides the final feasibility guarantee; jointly designing these components may improve efficiency and preserve sample quality. Finally, our current update assumes a differentiable terminal loss. For black-box rewards or constraints, zeroth-order gradient estimation could supply the terminal signal and extend the same map-based transport mechanism beyond differentiable objectives.

AI Use Statement

Agentic AI tools were used to assist with basic literature searches, provide feedback on the authors' theoretical claims, and refine the presentation of text and figures. The authors take full responsibility for the final manuscript, including its claims, citations, and figures.

REFERENCES

Giacomo Baldan, Qiang Liu, Alberto Guardone, and Nils Thuerey. Physics vs distributions: Pareto optimal flow matching with physics constraints. In *International Conference on Learning Representations*, 2026.

Jan-Hendrik Bastek, WaiChing Sun, and Dennis M. Kochmann. Physics-informed diffusion models. In *International Conference on Learning Representations*, 2025.

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

Joe Benton, George Deligiannidis, and Arnaud Doucet. Error bounds for flow matching methods. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=uqQPyWFDhY.

Andreas Bergmeister, Stefanie Jegelka, Nikolas Nüsken, Carles Domingo-Enrich, and Jakiw Pidstrigach. Reinforce adjoint matching: Scaling RL post-training of diffusion and flow-matching models. *arXiv preprint arXiv:2605.10759*, 2026. URL https://arxiv.org/abs/2605.10759v2.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/14f75513f0f1ca01de1e826b52e6b840-Abstract-Conference.html.

Nicholas M. Boffi, Michael S. Albergo, and Eric Vanden-Eijnden. Flow map matching with stochastic interpolants: A mathematical framework for consistency models. *Transactions on Machine Learning Research*, 2025a.

Nicholas Matthew Boffi, Michael Samuel Albergo, and Eric Vanden-Eijnden. How to build a consistency model: Learning flow maps via self-distillation. In *Advances in Neural Information Processing Systems*, 2025b.

Jinyuan Chang, Chenguang Duan, Yuling Jiao, Yi Xu, and Jerry Zhijian Yang. Inference-time alignment for diffusion models via variationally stable Doob's matching. *arXiv preprint arXiv:2601.06514*, 2026. URL https://arxiv.org/abs/2601.06514.

Nutan Chen, Jianxiang Feng, Marvin Alles, and Botond Cseke. ConFlow: Constraints-guided learning with flow matching for motion generation. *arXiv preprint arXiv:2607.14424*, 2026. URL https://arxiv.org/abs/2607.14424.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Chaoran Cheng, Boran Han, Danielle C. 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*, 2025. URL https://openreview.net/forum?id=teE4pl9ftK.

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

Jacob K. Christopher, James E. Warner, and Ferdinando Fioretto. Constraint-aware flow matching: Decision aligned end-to-end training for constrained sampling. *arXiv preprint arXiv:2605.12754*, 2026. URL https://arxiv.org/abs/2605.12754.

Kevin Clark, Paul Vicol, Kevin Swersky, and David J. Fleet. Directly fine-tuning diffusion models on differentiable rewards. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=1vmSEVL19f.

Carles Domingo-Enrich, Michal Drozdzal, Brian Karrer, and Ricky T. Q. Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. In *International Conference on Learning Representations*, 2025. URL https://iclr.cc/virtual/2025/poster/27782.

Berthy T. Feng, Ricardo Baptista, and Katherine L. Bouman. Neural approximate mirror maps for constrained diffusion models. In *International Conference on Learning Representations*, 2025.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *International Conference on Learning Representations*, 2025.

Zheming Fu, Ruizhe He, Wei Shang, Xiaoxiao Ma, Lei Wang, Chang Liu, and Siming Fu. FlowAWR: Online adaptive flow reinforcement via advantage-weighted rectification. *arXiv preprint arXiv:2606.30376*, 2026. URL https://arxiv.org/abs/2606.30376.

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

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J. Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. In *Advances in Neural Information Processing Systems*, 2025.

Paolo Giaretta, Zeyang Li, and Navid Azizan. DiRecT: Safe diffusion-based planning via receding-horizon denoising. *arXiv preprint arXiv:2606.15359*, 2026. URL https://arxiv.org/abs/2606.15359.

Michael B. Giles and Niles A. Pierce. An introduction to the adjoint approach to design. *Flow, Turbulence and Combustion*, 65:393–415, 2000. doi: 10.1023/A:1011430410075.

Yuchao Gu, Guian Fang, Yuxin Jiang, Weijia Mao, Song Han, Han Cai, and Mike Zheng Shou. AnyFlow: Any-step video diffusion model with on-policy flow map distillation. *arXiv preprint arXiv:2605.13724*, 2026. URL https://arxiv.org/abs/2605.13724.

Kehan Guo, Yili Shen, Yujun Zhou, Yue Huang, Chujie Gao, Shiyi Du, and Xiangliang Zhang. Reward transport: Property control in flow matching via noise-space alignment. *arXiv preprint arXiv:2607.08781*, 2026a. URL https://arxiv.org/abs/2607.08781.

Yi Guo, Wei Wang, Zhihang Yuan, Rong Cao, Kuan Chen, Zhengyang Chen, Yuanyuan Huo, Yang Zhang, Yuping Wang, Shouda Liu, and Yuxuan Wang. SplitMeanFlow: Interval splitting consistency in few-step generative modeling. *arXiv preprint arXiv:2507.16884*, 2025.

Zhengyi Guo, Wenpin Tang, and Renyuan Xu. Conditional diffusion guidance under hard constraint: A stochastic analysis approach. *arXiv preprint arXiv:2602.05533*, 2026b. URL https://arxiv.org/abs/2602.05533.

Samar Hadou, Yigit Berkay Uslu, and Alejandro Ribeiro. Constrained diffusion models with primal-dual inference. *arXiv preprint arXiv:2606.17192*, 2026. URL https://arxiv.org/abs/2606.17192.

Jonathan Heek, Emiel Hoogeboom, and Tim Salimans. Multistep consistency models. *arXiv preprint arXiv:2403.06807*, 2024.

Peter Holderrieth, Uriel Singer, Tommi Jaakkola, Ricky T. Q. Chen, Yaron Lipman, and Brian Karrer. GLASS flows: Transition sampling for alignment of flow and diffusion models. *arXiv preprint arXiv:2509.25170*, 2025.

Peter Holderrieth, Douglas Chen, Luca Eyring, Ishin Shah, Giri Anantharaman, Yutong He, Zeynep Akata, Tommi Jaakkola, Nicholas Matthew Boffi, and Max Simchowitz. Diamond maps: Efficient reward alignment via stochastic flow maps. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2602.05993.

Zhengyan Huan, Jacob Boerma, Li-Ping Liu, and Shuchin Aeron. Constraint-aware flow matching via randomized exploration. *Transactions on Machine Learning Research*, 2026.

Jerry Y. Huang, Justin Lin, Sheel Shah, Kartik Nair, and Nicholas M. Boffi. How to guide your flow: Few-step alignment via flow map reward guidance. In *International Conference on Machine Learning*, 2026a. URL https://icml.cc/virtual/2026/poster/62479.

Yushi Huang, Xiangxin Zhou, Jun Zhang, Liefeng Bo, and Tianyu Pang. MeanFlowNFT: Bringing forward-process RL to average-velocity generators. *arXiv preprint arXiv:2607.15273*, 2026b. URL https://arxiv.org/abs/2607.15273.

Alexis Jacq, Guillaume Couairon, Valentin De Bortoli, Quentin Berthet, Arnaud Doucet, and Romuald Elie. Diffusion fine-tuning with rewarded moment matching distillation. *arXiv preprint arXiv:2606.30414*, 2026. URL https://arxiv.org/abs/2606.30414.

Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ODE trajectory of diffusion. In *International Conference on Learning Representations*, 2024.

Cindy Xiangrui Kong, Yueqi Wang, Haoyang Zheng, Weijian Luo, and Guang Lin. Ultra fast PDE solving via physics guided few-step diffusion. *arXiv preprint arXiv:2602.03627*, 2026.

Jaewoo Lee, Hyeongyu Kang, Dohyun Kim, Kyuil Sim, Woocheol Shin, Minsu Kim, Taeyoung Yun, Jeongjae Lee, Sanghyeok Choi, Tabitha Edith Lee, Jong Chul Ye, and Jinkyoo Park. Aligning few-step generative models by amortizing sample-based variational inference. *arXiv preprint arXiv:2605.26552*, 2026a. URL https://arxiv.org/abs/2605.26552.

Jeongjae Lee, Jinho Chang, Jeongsol Kim, and Jong Chul Ye. Reward score matching: Unifying reward-based fine-tuning for flow and diffusion models. *arXiv preprint arXiv:2604.17415*, 2026b.

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

Zeyang Li, Kaveh Alim, and Navid Azizan. HardFlow: Hard-constrained sampling for flow-matching models via trajectory optimization. *arXiv preprint arXiv:2511.08425*, 2025. URL https://arxiv.org/abs/2511.08425.

Zhiqi Li, Wen Zhang, and Bo Zhu. Flow-map GRPO: Reinforcement learning for few-step flow-map generators via anchored stochastic composition. *arXiv preprint arXiv:2607.00535*, 2026b. URL https://arxiv.org/abs/2607.00535.

Jinhao Liang, Jacob K. Christopher, Sven Koenig, and Ferdinando Fioretto. Simultaneous multi-robot motion planning with projected diffusion models. In *International Conference on Machine Learning*, 2025a. URL https://arxiv.org/abs/2502.03607.

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

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

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-GRPO: Training flow matching models via online RL. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/3a10c46572628d58cb44fb705f25cbbf-Abstract-Conference.html.

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

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

Aaron Lou and Stefano Ermon. Reflected diffusion models. In *International Conference on Machine Learning*, 2023.

Cheng Lu and Yang Song. Simplifying, stabilizing and scaling continuous-time consistency models. In *International Conference on Learning Representations*, 2025.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/260a14acce2a89dad36adc8eefe7c59e-Abstract-Conference.html.

Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv preprint arXiv:2310.04378*, 2023.

Abbas Mammadov, So Takao, Bohan Chen, Ricardo Baptista, Morteza Mardani, Yee Whye Teh, and Julius Berner. Variational flow maps: Make some noise for one-step conditional generation. *arXiv preprint arXiv:2603.07276*, 2026. URL https://arxiv.org/abs/2603.07276.

Zhengkai Pan, Peter Potaptchik, Wenxi Yao, Michael S. Albergo, and Jakiw Pidstrigach. Itô maps for any-step SDEs. *arXiv preprint arXiv:2606.11156*, 2026. URL https://arxiv.org/abs/2606.11156.

Lev S. Pontryagin, Vladimir G. Boltyanskii, Revaz V. Gamkrelidze, and Evgenii F. Mishchenko. *The Mathematical Theory of Optimal Processes*. Interscience Publishers, John Wiley & Sons, New York, 1962.

Peter Potaptchik, Adhi Saravanan, Abbas Mammadov, Alvaro Prat, Michael S. Albergo, and Yee Whye Teh. Meta flow maps enable scalable reward alignment. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2601.14430.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *European Conference on Computer Vision*, 2024.

Jeongwoo Shin, Dongsoo Shin, Yuchen Zhu, Wei Guo, Yongxin Chen, Joonseok Lee, Jaewoong Choi, and Jaemoo Choi. Efficient adjoint matching for fine-tuning diffusion models. *arXiv preprint arXiv:2605.11480*, 2026.

Lawrence Sirovich. Turbulence and the dynamics of coherent structures. I. coherent structures. *Quarterly of Applied Mathematics*, 45(3):561–571, 1987. doi: 10.1090/qam/910462.

Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. In *International Conference on Learning Representations*, 2024.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning*, 2023.

Utkarsh Utkarsh, Pengfei Cai, Alan Edelman, Rafael Gomez-Bombarelli, and Christopher Rackauckas. Physics-constrained flow matching: Sampling generative models with hard constraints. In *Advances in Neural Information Processing Systems*, 2025.

Zidong Wang, Yiyuan Zhang, Xiaoyu Yue, Xiangyu Yue, Yangguang Li, Wanli Ouyang, and Lei Bai. Transition Models: Rethinking the generative learning objective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026. URL https://arxiv.org/abs/2509.04394.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Frédo Durand, and William T. Freeman. Improved distribution matching distillation for fast image synthesis. In *Advances in Neural Information Processing Systems*, volume 37, 2024a. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/54dcf25318f9de5a7a01f0a4125c541e-Abstract-Conference.html.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Frédo Durand, William T. Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6613–6623, 2024b.

Yi Zhang, Peng Wang, and Difan Zou. Physics-informed distillation of diffusion models for PDE-constrained generation. In *International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=t6dA9c3SlJ.

Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. DiffusionNFT: Online diffusion reinforcement with forward process. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2509.16117.

Linqi Zhou, Stefano Ermon, and Jiaming Song. Inductive moment matching. In *International Conference on Machine Learning*, pp. 78651–78686, 2025.

Mingyuan Zhou, Huangjie Zheng, Zhendong Wang, Mingzhang Yin, and Hai Huang. Score identity distillation: Exponentially fast distillation of pretrained diffusion models for one-step generation. In *International Conference on Machine Learning*, volume 235, pp. 62307–62331, 2024. URL https://proceedings.mlr.press/v235/zhou24x.html.

Qijie Zhu, Zeqi Ye, Han Liu, Zhaoran Wang, and Minshuo Chen. Training-free adaptation of diffusion models via Doob's h-transform. *arXiv preprint arXiv:2602.16198*, 2026. URL https://arxiv.org/abs/2602.16198.

APPENDIX CONTENTS

A Extended Related Work **15**
A.1 Constrained generation . . . 15
A.2 Reward alignment . . . 16
A.3 Few/Any-step generation and schedule consistency . . . 16
A.4 Summary and relation to our setting . . . 17

B Implementation and Proofs **17**
B.1 Terminal penalty choices . . . 17
B.2 Training implementation and computational cost . . . 17
B.3 Theoretical Analysis and Proofs . . . 18

C Experimental Setup **22**
C.1 Benchmarks and model representations . . . 22
C.2 Baselines and initialization . . . 24
C.3 Training and evaluation . . . 25
C.4 Cost Accounting and Reproducibility . . . 25

D Extended Experimental Results **25**
D.1 Cross-Schedule Benchmark Comparisons . . . 26

A EXTENDED RELATED WORK

This appendix extends the comparison in Section 2 around three questions: where constraint information enters generation, how a terminal objective produces a training update, and how a finite-transition model supports different sampling schedules.

A.1 CONSTRAINED GENERATION

Feasibility by construction. One family restricts the state space or changes coordinates so that the sampling dynamics remain in a prescribed domain. Reflected diffusion evolves a reflected process on a bounded set; mirror-map and gauge constructions instead train or specify transformations between an unconstrained representation and a feasible geometry (Lou & Ermon, 2023; Feng et al., 2025; Li et al., 2026a). These methods are well suited to constraints with an exploitable domain structure. Our setting starts from a differentiable terminal penalty.

Correction during sampling. Projection and guidance methods retain a pretrained generator and modify each sample as it is produced. Projected diffusion applies projection within the reverse process, while ECI alternates extrapolation, constraint correction, and interpolation (Christopher et al., 2024; Cheng et al., 2025). Physics-Constrained Flow Matching and DiRecT repeatedly form a clean or terminal prediction, solve a local correction problem, and propagate the correction back to the current sampling state (Utkarsh et al., 2025; Giaretta et al., 2026). Chance-Constrained Flow Matching similarly performs stochastic optimization at sampling time under an explicit risk specification (Liang et al., 2025b). PDI interleaves denoising with primal-dual updates at inference to handle constraints on expected generated outcomes (Hadou et al., 2026). Output-gradient and trajectory-optimization approaches instead differentiate through a pretrained flow or optimize its generated path (Ben-Hamu et al., 2024; Liu et al., 2023b; Li et al., 2025; Gao et al., 2026). These methods can address constraints absent during pretraining, but their constraint-dependent work is repeated for every generated sample.

FMRG uses the sensitivity of a remaining-time map to guide a frozen flow model, showing that a learned map can transport terminal information without differentiating through the preceding trajectory (Huang et al., 2026a). Posterior and Doob-transform approaches provide related inference-time or off-policy routes to terminal conditioning (Guo et al., 2026b; Zhu et al., 2026; Chang et al., 2026). These methods establish the value of map-based terminal feedback at inference. Our setting asks how to use the same information during training while keeping a shared map–velocity model consistent across alternative compositions.

Constraint information during training. A second family adds physical residuals, constraint penalties, or feasibility objectives to flow or diffusion training (Bastek et al., 2025; Huan et al., 2026;

Christopher et al., 2026; Zhang et al., 2026; Chen et al., 2026). Such training amortizes part of the constraint computation into the generator. The resulting update depends on where the residual is evaluated and how terminal credit is assigned. Phys-Instruct instead distills physics-guided generation into a faster model (Kong et al., 2026). Within this family, we focus on how a differentiable terminal signal updates the local dynamics of a flow-map model without a numerical ODE rollout.

A.2 Reward alignment

Reward alignment provides a broader view of terminal feedback because a constraint penalty can be treated as a differentiable negative reward. Methods for scalar or non-differentiable rewards commonly generate endpoints, evaluate their rewards, and then apply policy-gradient or reward-weighted regression. DDPO and Flow-GRPO follow policy-optimization routes, while DiffusionNFT re-noises scored endpoints for regression (Black et al., 2024; Liu et al., 2025; Zheng et al., 2026). Flow-Map GRPO stochasticizes map transitions for policy-gradient training, and MeanFlowNFT adapts forward-process reward regression to an average-velocity generator (Li et al., 2026b; Huang et al., 2026b); FlowAWR uses advantage-weighted rectification (Fu et al., 2026). These approaches can accept black-box rewards, but avoiding differentiation through the sampling path does not remove the endpoint rollouts used to obtain the rewards.

Adjoint Matching propagates terminal costates through trajectory-based control targets for differentiable terminal objectives (Domingo-Enrich et al., 2025). Efficient Adjoint Matching reduces the cost of this adjoint computation (Shin et al., 2026). For scalar rewards, Reinforce Adjoint Matching instead uses endpoint rewards and a REINFORCE identity to avoid a backward adjoint sweep (Bergmeister et al., 2026). FAV regresses sample-based variational updates into a generator, whereas RMMD and related reward-gradient methods differentiate a reward through a denoising or prediction map (Lee et al., 2026a; Jacq et al., 2026; Lee et al., 2026b). These methods differ in the learned object and in whether the feedback target is a local field, a one-step generator, or a finite transition.

Learned maps provide another route to terminal alignment. VFM jointly adapts an initial-noise transformation and a flow map; Meta Flow Maps use stochastic posterior maps for steering and off-policy fine-tuning (Mammadov et al., 2026; Potaptchik et al., 2026). FMRG uses a remaining-time map at inference (Huang et al., 2026a). Related approaches, including GLASS Flows and Diamond Maps, learn posterior transition samplers or stochastic maps for guidance, search, and reward alignment (Holderrieth et al., 2025; 2026); Reward Transport amortizes a selected property through noise-space alignment (Guo et al., 2026a). Our construction uses the learned map to approximate the terminal adjoint of the local velocity during training. The direct terminal loss updates the full-interval map, while map–velocity consistency propagates the updated dynamics to the finite transitions used at inference.

A.3 Few/Any-step generation and schedule consistency

Fast samplers arise either by numerically accelerating a local field or by learning a generator that directly represents a finite transition. High-order ODE solvers and rectified-flow techniques reduce the number of evaluations of a local velocity but still define generation through a numerical trajectory (Lu et al., 2022; Liu et al., 2023a). Progressive distillation, consistency models, latent consistency models, and distribution-matching distillation instead compress a teacher or terminal distribution into a small number of prescribed generation steps (Salimans & Ho, 2022; Song et al., 2023; Song & Dhariwal, 2024; Luo et al., 2023; Heek et al., 2024; Lu & Song, 2025; Yin et al., 2024b;a; Sauer et al., 2024; Zhou et al., 2024; 2025). These approaches provide fast sampling, but a model trained for one endpoint or schedule need not expose a transition for every subinterval.

FMM, Shortcut Models, MeanFlow, CTM, and TiM learn finite transitions that can be queried across time intervals (Boffi et al., 2025a;b; Frans et al., 2025; Geng et al., 2025; Kim et al., 2024; Wang et al., 2026). AnyFlow likewise exposes multiple inference budgets through on-policy flow-map distillation (Gu et al., 2026). For a transition map $T(x; s, t)$ associated with a local velocity v, the relevant population relations are

$$T(x; s, s) = x, \qquad \partial_t T(x; s, t) = v(T(x; s, t), t), \qquad T(x; s, u) = T(T(x; s, t); t, u). \tag{15}$$

The first two connect finite transitions to the local dynamics, while the last expresses composition across a sampling schedule. FMM self-distillation, Shortcut training, and MeanFlow impose different

sampled forms of these relations (Boffi et al., 2025b; Frans et al., 2025; Geng et al., 2025; Guo et al., 2025).

Terminal alignment introduces a further requirement: an improvement to the direct endpoint should persist when shorter transitions are composed. CTM's loss-guided sampling and stochastic Itô-map reward steering address downstream objectives at inference time (Kim et al., 2024; Pan et al., 2026). VFM and MeanFlowNFT already combine alignment with mechanisms that couple finite and instantaneous predictions (Mammadov et al., 2026; Huang et al., 2026b). Our method addresses this requirement by using a map-based local adjoint to update the local dynamics and forward map–velocity consistency to align the finite transitions with that updated velocity.

A.4 Summary and relation to our setting

Constrained-generation methods determine where feasibility information enters the returned sample. Reward-alignment methods determine how terminal information becomes a parameter update. Few/any-step methods determine which finite transitions are available and how they compose. Our setting connects these three questions: a learned map approximates the local terminal adjoint, and consistency training links the updated local dynamics to the transitions used under different sampling schedules.

B Implementation and Proofs

This appendix specifies the terminal penalties, consistency and adjoint updates, and terminal-feedback cost. It then gives the proofs supporting the main-text analysis.

B.1 Terminal penalty choices

The map-adjoint update differentiates the terminal penalty ρ at the generated endpoint.

Constraint penalties. Let $g(x) \le 0$ and $h(x) = 0$ be the terminal constraints. Define

$$\mathcal{C} = \{x : g(x) \le 0,\ h(x) = 0\}, \quad \nu(x) = \Big(\|[g(x)]_+\|_2^2 + \|h(x)\|_2^2\Big)^{1/2}. \tag{16}$$

A standard choice is $\rho(x) = \nu(x)^2$, with smooth surrogates where needed for differentiation.

Benchmark penalties. For a benchmark-native nonnegative penalty $P(x)$, the experiments use the smooth distance-like transform

$$\widetilde{P}_\delta(x) = \frac{P(x)}{\sqrt{P(x)+\delta^2}+\delta} = \sqrt{P(x)+\delta^2} - \delta, \qquad \delta = 0.01. \tag{17}$$

The untransformed benchmark residual remains the reported quantity.

When exact feasibility is required, a final projection onto $\mathcal{C}$ can be applied at inference.

B.2 Training implementation and computational cost

Velocity–map consistency loss. The loss in (5) uses $(Z, X) \sim p_0 \otimes p_{\mathrm{data}}$, $x = X_s$, and $0 \le s < t \le 1$. In the experiments, we approximate $\partial_t T_\theta$ by a first-order finite difference with step 0.01. Each batch assigns 75% of its examples to diagonal FM regression and 25% to this off-diagonal consistency update, with consistency weight 1. Further training settings are given in Appendix C.

Adjoint gradient injection. The implementation uses the mean terminal loss in the main text. The transported covector $\widehat{a}_i$ is computed by a reverse-mode input VJP through the suffix map and then detached together with the prefix state. In (11), the stopped copy of Ψ_{adj} has zero derivative. Differentiating the other copy while holding $\widehat{x}_i$ and $\widehat{a}_i$ fixed therefore returns the displayed local-velocity update, despite the zero forward value of $\mathcal{L}_{\mathrm{adj}}$.

Algorithm 1 Simulation-free terminal alignment of a shared flow map

Require: Pretrained shared velocity–map parameters θ^0, terminal penalty ρ, and weights $\lambda_{\mathrm{lsd}}, \lambda_{\mathrm{vio}}, \lambda_{\mathrm{adj}}$
1: **for** $k = 0, \ldots, N_{\mathrm{it}} - 1$ **do**
2: Sample paired FM data (z_i, x_i), LSD intervals, and adjoint times
3: Evaluate $\mathcal{L}_{\mathrm{fm}}$ and the finite-difference estimate of the stopped-forward $\mathcal{L}_{\mathrm{lsd}}$
4: Evaluate direct endpoints and $\mathcal{L}_{\mathrm{vio}}^{\mathrm{map}} = B^{-1} \sum_i \rho(T_\theta(z_i; 0, 1))$
5: Form stopped intermediate states $\widehat{x}_i$, suffix endpoints $\widehat{y}_i$, and stopped covectors $\widehat{a}_i$ as in (10)
6: Form $\mathcal{L}_{\mathrm{adj}}$ by (11)
7: Update θ using (12)
Ensure: Frozen checkpoint $\widehat{\theta}$

Training-time complexity. Pathwise differentiation of the ODE terminal loss evaluates and differentiates through a numerical trajectory, with work growing with the number of solver evaluations. Trajectory policy gradients require sampled trajectories, with work growing with their number and length. The map adjoint instead uses a fixed number of learned-map evaluations and VJPs to approximate the ODE terminal gradient, independent of numerical rollout length. Direct endpoint differentiation also has fixed cost, but differentiates $\mathcal{L}_{\mathrm{vio}}^{\mathrm{map}}$ rather than $\mathcal{L}_{\mathrm{vio}}^{\mathrm{ode}}$. Figure 4(c) reports complete optimizer-update times.

B.3 Theoretical Analysis and Proofs

The proofs follow the two links used by the method. We first derive the exact ODE terminal gradient in Lemma 1 and analyze the learned-map substitution used to approximate it. We then track the same map–velocity residual across a complete sampling schedule: first to control finite-map error relative to the learned ODE, and then, together with flow-matching error, to prove Theorem 2. We finally extend these conclusions to the quadratic violation penalty and the latent-decoder setting used in the paper.

B.3.1 From the exact terminal adjoint to its map approximation

Lemma 1 identifies the exact local-velocity update induced by the ODE terminal objective. The method estimates this update without an ODE rollout by replacing the exact trajectory state, terminal state, and remaining-time sensitivity with learned map predictions in (10). We prove the identity and bound the error of this substitution before relating the direct map endpoint in (7) to the ODE terminal loss.

Proof of the exact identity. Assume that v_θ is continuously differentiable in its state and parameters near the relevant trajectories, with integrable derivative bounds that justify the flow-sensitivity equations. Assume also that ρ is differentiable at the generated endpoints and that differentiation can be interchanged with expectation over Z.

Proof of Lemma 1. Fix z and write $x^\theta(t) = \phi_{0,t}^\theta(z)$, as in the main text. The state sensitivity $S(t) = D_\theta x^\theta(t)$ satisfies $\dot{S} = D_x v_\theta S + D_\theta v_\theta$ and $S(0) = 0$. Let $J(t) = D_x \phi_{t,1}^\theta(x^\theta(t))$. The flow composition $J(t) = J(t+h) D_x \phi_{t,t+h}^\theta(x^\theta(t))$ implies $\dot{J}(t) = -J(t) D_x v_\theta(x^\theta(t), t)$. Thus the transported terminal gradient $a(t) = J(t)^\top \nabla \rho(x^\theta(1))$ satisfies $-\dot{a} = D_x v_\theta^\top a$, with $a(1) = \nabla \rho(x^\theta(1))$. All velocity derivatives here are evaluated at $(x^\theta(t), t)$. Pairing the two sensitivity equations cancels the state-Jacobian terms:

$$\frac{d}{dt}\big(S(t)^\top a(t)\big) = D_\theta v_\theta(x^\theta(t), t)^\top a(t).$$

Integrating over $[0, 1]$, using $S(0) = 0$, and applying the endpoint chain rule gives

$$\nabla_\theta \rho(\phi_1^\theta(z)) = \mathbb{E}_{t \sim \mathrm{Unif}[0,1]}\left[D_\theta v_\theta(x^\theta(t), t)^\top a^\theta(z, t) \right].$$

Averaging over Z and interchanging differentiation and expectation gives (9). □

Replacing the exact flow by learned maps. The identity above supplies the target gradient. Its map-based implementation through (11) depends on the accuracy of the intermediate state, the composed endpoint, and the Jacobian that transports the terminal gradient. For a fixed base sample z, retain $x^\theta(t) = \phi^\theta_{0,t}(z)$ and write

$$y^\theta = x^\theta(1), \qquad \widehat{x}_t = T_\theta(z; 0, t), \qquad \widehat{y}_t = T_\theta(\widehat{x}_t; t, 1).$$

For the map-adjoint approximation, define

$$\epsilon_{\mathrm{state}}(t) = \left\|\widehat{x}_t - x^\theta(t)\right\|, \qquad \epsilon_{\mathrm{end}}(t) = \left\|\widehat{y}_t - y^\theta\right\|,$$
$$\epsilon_{\mathrm{jac}}(t) = \left\|\mathsf{J}^\theta_{t,1}(\widehat{x}_t) - D_x\phi^\theta_{t,1}(x^\theta(t))\right\|_{\mathrm{op}}.$$

The LSD residual along the two map transitions used by the adjoint is

$$r^{\mathrm{adj}}_\theta(z,t,u) = \begin{cases} r_\theta(z;0,u), & 0 \le u \le t, \\ r_\theta(\widehat{x}_t;t,u), & t < u \le 1. \end{cases}$$

Its squared energy is

$$\mathcal{E}_{\mathrm{adj}}(z) = \mathbb{E}_t\left[\int_0^1 \left\|r^{\mathrm{adj}}_\theta(z,t,u)\right\|^2 du\right].$$

For the gradient comparison, assume on a common region containing y^θ and all $\widehat{y}_t$ that ρ is differentiable, $\|\nabla\rho\| \le G_\rho$, and $\nabla\rho$ is L_ρ-Lipschitz. For a nonsmooth penalty, these assumptions apply to the specified smooth surrogate. On a domain containing both $x^\theta(t)$ and $\widehat{x}_t$ for all t, assume $\|D_\theta v_\theta\|_{\mathrm{op}} \le B_\theta$ and that $D_\theta v_\theta$ is L_θ-Lipschitz in state. Let also $\left\|D_x\phi^\theta_{t,1}(x^\theta(t))\right\|_{\mathrm{op}} \le B_\phi$. Suppose $v_\theta(\cdot, u)$ is $\ell_\theta(u)$-Lipschitz on a domain containing the exact ODE path and both map curves used to define r^{adj}_θ, with $\Lambda_\theta = \int_0^1 \ell_\theta(u)\,du < \infty$. Assume these map curves are absolutely continuous and $\mathcal{E}_{\mathrm{adj}}(z) < \infty$.

Define $\widehat{d}_\theta(z,t) = D_\theta v_\theta(\widehat{x}_t, t)^\top \widehat{a}_t$, with $\widehat{a}_t = \mathsf{J}^\theta_{t,1}(\widehat{x}_t)^\top \nabla\rho(\widehat{y}_t)$. Let $\overline{d}_\theta(z) = \mathbb{E}_t\left[\widehat{d}_\theta(z,t)\right]$. Here the partial parameter derivative holds the state fixed. The exact local integrand is

$$d^{\mathrm{ode}}_\theta(z,t) = D_\theta v_\theta(x^\theta(t), t)^\top a^\theta(z,t).$$

Under these assumptions, the time-averaged map-adjoint error satisfies

$$\left\|\overline{d}_\theta(z) - \nabla_\theta\rho(\phi^\theta_1(z))\right\| \le e^{\Lambda_\theta} B_\phi(B_\theta L_\rho + L_\theta G_\rho)\sqrt{\mathcal{E}_{\mathrm{adj}}(z)} + B_\theta G_\rho \mathbb{E}_t[\epsilon_{\mathrm{jac}}(t)]. \tag{18}$$

Derivation of the map-adjoint error bound. We first bound the error in the transported terminal gradient, then account for evaluating the velocity derivative at an approximate state. Let $J_t = D_x\phi^\theta_{t,1}(x^\theta(t))$, $\widehat{J}_t = \mathsf{J}^\theta_{t,1}(\widehat{x}_t)$, $q = \nabla\rho(y^\theta)$, and $\widehat{q}_t = \nabla\rho(\widehat{y}_t)$. Adding and subtracting $J_t^\top \widehat{q}_t$ gives

$$\begin{aligned}\left\|\widehat{J}_t^\top \widehat{q}_t - J_t^\top q\right\| &\le \left\|\widehat{J}_t - J_t\right\|_{\mathrm{op}} \|\widehat{q}_t\| + \|J_t\|_{\mathrm{op}} \|\widehat{q}_t - q\| \\ &\le G_\rho \epsilon_{\mathrm{jac}}(t) + B_\phi L_\rho \epsilon_{\mathrm{end}}(t).\end{aligned}$$

The first bound controls $\widehat{a}_t - a^\theta(z,t)$. The local parameter update also depends on where the velocity is evaluated:

$$\begin{aligned}\widehat{d}_\theta(z,t) - d^{\mathrm{ode}}_\theta(z,t) = {} & D_\theta v_\theta(\widehat{x}_t, t)^\top\left(\widehat{a}_t - a^\theta(z,t)\right) \\ & + \left[D_\theta v_\theta(\widehat{x}_t, t) - D_\theta v_\theta(x^\theta(t), t)\right]^\top a^\theta(z,t).\end{aligned}$$

Since $\left\|a^\theta(z,t)\right\| \le B_\phi G_\rho$, the bound and spatial Lipschitz property of $D_\theta v_\theta$ give

$$\left\|\widehat{d}_\theta(z,t) - d^{\mathrm{ode}}_\theta(z,t)\right\| \le B_\theta[G_\rho\epsilon_{\mathrm{jac}}(t) + B_\phi L_\rho \epsilon_{\mathrm{end}}(t)] + L_\theta B_\phi G_\rho \epsilon_{\mathrm{state}}(t).$$

The prefix and suffix map curves concatenate continuously and follow the learned ODE with additive residual r_θ^{adj}. Gronwall's inequality and Cauchy–Schwarz therefore give, with $R_t = (\int_0^1 \left\| r_\theta^{\mathrm{adj}}(z,t,u) \right\|^2 du)^{1/2}$,

$$\epsilon_{\mathrm{state}}(t) \le e^{\Lambda_\theta} R_t, \qquad \epsilon_{\mathrm{end}}(t) \le e^{\Lambda_\theta} R_t.$$

Substitute these bounds above, average over t, and use $\mathbb{E}_t[R_t] \le \sqrt{\mathcal{E}_{\mathrm{adj}}(z)}$. Invoking Lemma 1 then gives (18). □

Equation (18) therefore connects the time-averaged map-adjoint direction back to Lemma 1: the LSD residual controls the state and endpoint errors created by replacing the ODE trajectory with map transitions, while the remaining-time Jacobian error accounts for approximate gradient transport. The empirical comparison in Figure 2 examines these two aspects through terminal-state error and directional agreement with the numerical ODE adjoint. The direct endpoint branch has a simpler value comparison: if ρ is G_ρ-Lipschitz on the terminal region, then

$$\left| \rho(T_\theta(z;0,1)) - \rho(\phi_1^\theta(z)) \right| \le G_\rho \left\| T_\theta(z;0,1) - \phi_1^\theta(z) \right\|.$$

We next analyze how map–velocity residuals affect composed transitions across sampling schedules.

B.3.2 Finite-step sampling and distribution bounds

The adjoint bound follows the two transitions used by one map-adjoint update. For any-step inference, we evaluate the same map–velocity residual along every transition of a deployed composition, rather than on the reference-path samples used in (5). Proposition 3 first compares that composition with the ODE of the same learned velocity. The proof of Theorem 2 then adds the error between the learned and reference velocities, yielding the distribution and cross-schedule bounds in the main text.

Conditions for the sampling error bounds. Let $v_{\mathrm{fm}}(x,t) = \mathbb{E}[X - Z \mid X_t = x, t]$ be the unrestricted population FM minimizer. Choose a version that generates a well-posed flow $\phi_{s,t}^{\mathrm{FM}}$; write $\phi_t^{\mathrm{FM}} = \phi_{0,t}^{\mathrm{FM}}$, and $p_\theta^{\mathrm{ode}} = (\phi_1^\theta)_\# p_0$. Define $\Delta_{\mathrm{fm}}(\theta) = \mathcal{L}_{\mathrm{fm}}(v_\theta) - \mathcal{L}_{\mathrm{fm}}(v_{\mathrm{fm}})$. Assume the learned and reference ODEs are well posed and that the relevant maps are measurable with finite second moments. Suppose $v_\theta(\cdot,t)$ is $\ell_\theta(t)$-Lipschitz on a common domain $\mathcal{D}_\theta$ containing the reference, learned-ODE, and finite-map trajectories, and define

$$C_\theta = \int_0^1 \exp\left(2 \int_t^1 \ell_\theta(a)\, da \right) dt, \qquad \Lambda_\theta = \int_0^1 \ell_\theta(t)\, dt < \infty. \tag{19}$$

Assume $T_\theta(x;s,s) = x$. For each compared partition, each curve $u \mapsto T_\theta(\widehat{X}_{j-1}; t_{j-1}, u)$ is absolutely continuous almost surely and its map–velocity residual is square-integrable. The stated terminal-loss bound additionally requires ρ to be Lipschitz. For the distribution bound, assume $(\phi_t^{\mathrm{FM}})_\# p_0 = \mathrm{Law}(X_t)$ for almost every t, and $(\phi_1^{\mathrm{FM}})_\# p_0 = p_{\mathrm{data}}$. For the composed states $\widehat{X}_j$ in (6), define

$$\mathcal{E}_{\mathrm{lsd}}(\theta;\mathcal{P}) = \sum_{j=1}^{J} \int_{t_{j-1}}^{t_j} \mathbb{E}_Z \left[\left\| r_\theta(\widehat{X}_{j-1}; t_{j-1}, u) \right\|^2 \right] du. \tag{20}$$

To compare finite-map sampling with the ODE, concatenate the transitions as $\psi_{\mathcal{P}}(u,z) = T_\theta(\widehat{X}_{j-1}; t_{j-1}, u)$ on each interval $[t_{j-1}, t_j]$, with $r_{\mathcal{P}}(u,z) = r_\theta(\widehat{X}_{j-1}; t_{j-1}, u)$. Identity initialization makes the pieces agree at the knots, so $\psi_{\mathcal{P}}$ is absolutely continuous from z to $G_{\theta,\mathcal{P}}(z)$, and $\dot{\psi}_{\mathcal{P}} = v_\theta(\psi_{\mathcal{P}}, u) + r_{\mathcal{P}}$ almost everywhere. Its residual energy is $\mathbb{E}_Z\left[\int_0^1 \|r_{\mathcal{P}}(u,Z)\|^2 du\right] = \mathcal{E}_{\mathrm{lsd}}(\theta;\mathcal{P})$.

For fixed partitions $\mathcal{P}$ and $\mathcal{Q}$, write $\|U\|_{L^2(Z)} = (\mathbb{E}_Z\left[\|U\|^2\right])^{1/2}$, and define

$$e_{\mathrm{ode}}(\theta;\mathcal{P}) = \left\|G_{\theta,\mathcal{P}}(Z) - \phi_1^\theta(Z)\right\|_{L^2(Z)}, \tag{21}$$

$$\kappa_{\theta;\mathcal{P},\mathcal{Q}} = \left\|G_{\theta,\mathcal{P}}(Z) - G_{\theta,\mathcal{Q}}(Z)\right\|_{L^2(Z)}. \tag{22}$$

Proposition 3 (LSD-to-composition stability)**.** *Suppose $T_\theta(x;s,s) = x$, the finite map trajectories are absolutely continuous with square-integrable residuals, and $v_\theta(\cdot,t)$ is $\ell_\theta(t)$-Lipschitz on a domain containing the exact and finite-map trajectories compared below. Then each fixed finite partition $\mathcal{P}$ and each pair $(\mathcal{P},\mathcal{Q})$ satisfy*

$$e_{\mathrm{ode}}(\theta;\mathcal{P}) \le e^{\Lambda_\theta}\sqrt{\mathcal{E}_{\mathrm{lsd}}(\theta;\mathcal{P})}, \tag{23}$$

$$\kappa_{\theta;\mathcal{P},\mathcal{Q}} \le e^{\Lambda_\theta}\left(\sqrt{\mathcal{E}_{\mathrm{lsd}}(\theta;\mathcal{P})} + \sqrt{\mathcal{E}_{\mathrm{lsd}}(\theta;\mathcal{Q})}\right). \tag{24}$$

Proof of Proposition 3. Compare the concatenated curve $\psi_{\mathcal{P}}(u,Z)$ with $Y_u = \phi_{0,u}^\theta(Z)$. They start from the same Z, and the former follows the learned ODE with the additive residual $r_{\mathcal{P}}$. Gronwall's inequality gives the first bound below; Cauchy–Schwarz over the unit time interval gives the second:

$$\begin{aligned}\left\|G_{\theta,\mathcal{P}}(Z) - \phi_1^\theta(Z)\right\| &\le \sum_{j=1}^{J}\int_{t_{j-1}}^{t_j}\exp\left(\int_u^1 \ell_\theta(a)\,da\right)\left\|r_\theta(\widehat{X}_{j-1};t_{j-1},u)\right\|\,du \\ &\le e^{\Lambda_\theta}\left(\sum_{j=1}^{J}\int_{t_{j-1}}^{t_j}\left\|r_\theta(\widehat{X}_{j-1};t_{j-1},u)\right\|^2\,du\right)^{1/2}.\end{aligned}$$

The intervals partition total time length one, so this step introduces no additional factor depending on their number. Squaring, taking expectation, and using (20) proves (23). Applying the same result to $\mathcal{Q}$ and using the triangle inequality in $L^2(Z)$ gives

$$\kappa_{\theta;\mathcal{P},\mathcal{Q}} \le e_{\mathrm{ode}}(\theta;\mathcal{P}) + e_{\mathrm{ode}}(\theta;\mathcal{Q}),$$

which proves (24). □

Proposition 3 isolates the error caused by using finite map transitions while holding the learned velocity fixed. Zero deployed residual energy makes the composition agree almost surely with the learned ODE for that schedule. If the residual vanishes for every state and interval, identity initialization and ODE uniqueness give $T_\theta(x;s,t) = \phi_{s,t}^\theta(x)$, as in Boffi et al. (2025b, Proposition 2.2).

Combining velocity and finite-map errors. The proposition above controls approximation of the learned ODE, but does not yet compare that ODE with the data distribution. The remaining step controls this second error through the excess flow-matching risk and combines the two estimates.

Proof of Theorem 2. We first control the learned ODE through the excess FM risk. Couple $Y_t = \phi_{0,t}^\theta(Z)$ and $Y_t^{\mathrm{FM}} = \phi_t^{\mathrm{FM}}(Z)$ through the same Z, and let $\delta v_t = v_\theta(Y_t^{\mathrm{FM}},t) - v_{\mathrm{fm}}(Y_t^{\mathrm{FM}},t)$. For $U = X - Z$, the population minimizer satisfies $\mathbb{E}[U - v_{\mathrm{fm}}(X_t,t) \mid X_t,t] = 0$. The regression identity and $(\phi_t^{\mathrm{FM}})_{\#}p_0 = \mathrm{Law}(X_t)$ then give $\mathbb{E}[\int_0^1 \|\delta v_t\|^2\,dt] = \Delta_{\mathrm{fm}}(\theta)$. The learned-velocity Lipschitz bound, Gronwall's inequality, and Cauchy–Schwarz therefore yield

$$\mathbb{E}\left[\left\|Y_1 - Y_1^{\mathrm{FM}}\right\|^2\right] \le C_\theta\Delta_{\mathrm{fm}}(\theta).$$

Since the reference terminal law is p_{data}, this shared-base coupling and Proposition 3 give

$$\begin{aligned}W_2(p_{\theta,\mathcal{P}},p_{\mathrm{data}}) &\le W_2(p_{\theta,\mathcal{P}},p_\theta^{\mathrm{ode}}) + W_2(p_\theta^{\mathrm{ode}},p_{\mathrm{data}}) \\ &\le e_{\mathrm{ode}}(\theta;\mathcal{P}) + \sqrt{C_\theta\Delta_{\mathrm{fm}}(\theta)} \\ &\le \sqrt{C_\theta\Delta_{\mathrm{fm}}(\theta)} + e^{\Lambda_\theta}\sqrt{\mathcal{E}_{\mathrm{lsd}}(\theta;\mathcal{P})},\end{aligned}$$

where the last inequality uses (23).

For a G_ρ-Lipschitz terminal loss and any two finite schedules $\mathcal{P}$ and $\mathcal{Q}$, the common-input coupling gives

$$\begin{aligned}&|\mathbb{E}[\rho(G_{\theta,\mathcal{P}}(Z))] - \mathbb{E}[\rho(G_{\theta,\mathcal{Q}}(Z))]\,| \\ &\quad\le G_\rho \kappa_{\theta;\mathcal{P},\mathcal{Q}} \\ &\quad\le G_\rho e^{\Lambda_\theta}\left(\sqrt{\mathcal{E}_{\mathrm{lsd}}(\theta;\mathcal{P})} + \sqrt{\mathcal{E}_{\mathrm{lsd}}(\theta;\mathcal{Q})}\right),\end{aligned}$$

by Cauchy–Schwarz and (24). This proves (14). □

Squared violation penalties. Theorem 2 bounds cross-schedule changes for Lipschitz terminal losses, while the constraint objective in (2) uses the squared violation magnitude. Write $\nu(y) = (\|[g(y)]_+\|_2^2 + \|h(y)\|_2^2)^{1/2}$, so $\rho = \nu^2$. Because the square need not be Lipschitz, we bound its expectation through ν instead. Suppose $\nu \ge 0$ is L_{con}-Lipschitz, $V_{\mathcal{Q}} = \|\nu(G_{\theta,\mathcal{Q}}(Z))\|_{L^2(Z)} < \infty$, and $\kappa_{\theta;\mathcal{P},\mathcal{Q}} < \infty$. Cauchy–Schwarz, Lipschitz continuity, and Minkowski's inequality give

$$\begin{aligned}\left|\mathbb{E}\big[\nu(G_{\theta,\mathcal{P}}(Z))^2\big] - \mathbb{E}\big[\nu(G_{\theta,\mathcal{Q}}(Z))^2\big]\right| &\le \|\nu(G_{\theta,\mathcal{P}}(Z)) - \nu(G_{\theta,\mathcal{Q}}(Z))\|_{L^2(Z)} \\ &\quad\times \big(\|\nu(G_{\theta,\mathcal{P}}(Z))\|_{L^2(Z)} + V_{\mathcal{Q}}\big) \\ &\le L_{\mathrm{con}}\kappa_{\theta;\mathcal{P},\mathcal{Q}}\big(2V_{\mathcal{Q}} + L_{\mathrm{con}}\kappa_{\theta;\mathcal{P},\mathcal{Q}}\big).\end{aligned} \tag{25}$$

Taking $\mathcal{Q} = (0,1)$, low one-step violation carries over to $\mathcal{P}$ when $\kappa_{\theta;\mathcal{P},\mathcal{Q}}$ is small; Proposition 3 bounds this discrepancy by the deployed LSD residual energies.

Frozen latent-space decoders. In SMD9 and Dynamic Stall, the flow is learned in a POD latent space with a frozen encoder and decoder, while generated trajectories and fields are evaluated after decoding. We therefore transfer the distribution bound in Theorem 2 to physical outputs. Let μ be the physical data law, E a fixed measurable encoder, and D a fixed L_D-Lipschitz decoder. For a generated latent law p, assume $p, E_\#\mu \in \mathcal{P}_2$ and $\epsilon_{\mathrm{rec}} = \|D(E(X)) - X\|_{L^2(X\sim\mu)} < \infty$. Then

$$W_2(D_\#p, \mu) \le L_D W_2(p, E_\#\mu) + \epsilon_{\mathrm{rec}}. \tag{26}$$

Push a latent coupling through D, then use the coupling $(D(E(X)), X)$ and the Wasserstein triangle inequality. When the latent FM target is $E_\#\mu$, applying Theorem 2 to $p = p_{\theta,\mathcal{P}}$ shows that physical-space distribution error is bounded by the latent generator error amplified by L_D plus the reconstruction error ϵ_{rec}. For the affine POD decoder $D(h) = b + Ah$, $L_D = \|A\|_{\mathrm{op}}$ includes coefficient rescaling. If physical-space violation ν is L_{con}-Lipschitz, then $\nu \circ D$ is $L_{\mathrm{con}}L_D$-Lipschitz, so (25) also applies to decoded penalties across schedules.

C Experimental Setup

This appendix specifies the benchmarks and reported metrics, model representations, comparison methods, and shared evaluation and cost settings used in Section 6. Numerical results appear in Appendix D.

C.1 Benchmarks and model representations

Table 2: Benchmark roles and representations. Constraint and quality metrics are evaluated after decoding in the benchmark's physical space.

Benchmark	Domain	Representation	Dimension	Role
Six Disks	2D samples	Native	2	Main comparison
Checkerboard	2D samples	Native	2	Appendix comparison
SMD6	Joint trajectories	Frozen POD	6×16	Appendix comparison
SMD9	Joint trajectories	Frozen POD	9×16	Main comparison
Stokes	100×100 field	Frozen POD	128	Appendix comparison
Dynamic Stall	$6 \times 128 \times 128$ fields	Frozen POD	512	Main comparison

Six Disks. For $j = 0, \dots, 5$, let

$$c_j = 1.05(\cos(2\pi j/6), \sin(2\pi j/6)), \qquad \mathcal{C} = \bigcup_j B(c_j, 0.27). \tag{27}$$

The target is a six-mode distribution supported on these disconnected disks. The Gaussian modes have standard deviation 0.16 and centers displaced by $0.22d_j$ from c_j, where the directions d_j are generated with seed 314159. The data distribution retains only feasible samples. The comparison tables report mean distance outside the union of disks and sliced Wasserstein distance (SWD).

Checkerboard. The feasible set contains disconnected cells in a 4×4 checkerboard. The gap fraction is 0.08, the within-cell mode offset fraction is 0.25, the mode standard-deviation fraction is 0.18, and the geometry seed is 314159. It uses the same toy architecture, comparison set, sample count, and distribution metrics as Six Disks.

SMD6. SMD6 contains joint trajectories for six robots with 64 time points over a duration of 5. We use the fixed dataset package and its predefined splits. The training penalty averages squared continuous collision residuals over time and sums over robot pairs. The reported violation is the positive worst continuous penetration per trajectory, averaged over samples and tasks. The world-bound constraint is disabled. Each evaluation task is the independent aggregation unit; its 64 source-noise samples are not treated as independent tasks.

SMD9. The SMD9 adapter uses nine robots and 64 waypoints, with the same forms of training penalty and reported violation as SMD6 and no world-bound term. It has duration 5, supplies start and goal positions as conditions, and uses a 9×16-dimensional frozen POD latent. The evaluation uses 50 test tasks and 64 generated samples per task. The public SMD waypoint collision ratio is recorded separately from the continuous-penetration metric used in the comparison tables.

Physical fields. Stokes uses 100×100 fields. The comparison tables report mean positive excess initial-condition MAE above the 10^{-3} tolerance, while the ablation figure reports initial-condition MSE. Each field benchmark retains its original data split, normalization, residual, and native quality definitions.

Dynamic Stall uses six-channel 128×128 fields from the PBFM setting (Baldan et al., 2026). Its native residual MSE is the spatial mean of ideal-gas and wall-shear squared residuals. The project-specific 2.5×10^{-9} reference tolerance defines the reported mean positive excess residual; the raw residual and violation rate are also recorded.

Table 3: Benchmark sources and adaptations.

Benchmark	Source	Implementation used here
Six Disks	Constructed here	Seeded six-mode distribution and analytic union-of-disks violation.
Checkerboard	Flow-map self-distillation example (Boffi et al., 2025b)	Reimplemented with the stated cell gap, mode offset, and seed.
SMD6	Six-robot SMD subset (Liang et al., 2025a)	Released task representation and collision evaluator with a frozen POD generator.
SMD9	Nine-robot SMD setting (Liang et al., 2025a)	Joint 64-waypoint paths in the empty/no-wait setting; continuous penetration for evaluation.
Stokes	Analytic sampler; ECI initial-condition setting (Cheng et al., 2025)	Decoded 100×100 fields with initial-condition evaluation.
Dynamic Stall	PBFM field data (Baldan et al., 2026)	Decoded six-channel 128×128 fields; ideal-gas and wall-shear residuals with the stated tolerance.

The released SMD implementation supplies the task representation and collision evaluator. The field-data versions and analytic-sampler seeds are recorded with the experimental artifacts.

Model architecture and frozen representations. Toy generators use a residual MLP with width 64 and depth 3. State and time embeddings are combined in every residual block. For two-time map inputs, the two independently embedded times are averaged after the complete time embedder rather than averaging their raw sinusoidal features.

For the high-dimensional benchmarks, let $y \in \mathbb{R}^d$ be a represented state, m its training mean, $U \in \mathbb{R}^{d\times r}$ the orthonormal POD basis, and $\bar{c}, s \in \mathbb{R}^r$ the training mean and standard deviation of its coefficients. The frozen encoder and decoder are

$$h = \big(U^\top (y - m) - \bar{c}\big) \oslash s, \qquad D(h) = m + U(\bar{c} + s \odot h). \tag{28}$$

The operations $\odot$ and $\oslash$ act componentwise; y contains each agent's interior waypoints for robot trajectories and the vectorized field for PDE data. The basis and coefficient statistics are fitted on the training split only and then frozen. The affine decoder passes terminal gradients to the latent generator, while discarded modes can leave reconstruction error in physical space. Small robot covariance matrices use an exact FP64 eigendecomposition. PDE matrices use seeded randomized FP32 SVD with 16 oversamples and four power iterations. The coefficient standard deviation has floor 10^{-4}. The conditional latent generator is a residual MLP with width 128 and depth 3. Codec records include retained variance, basis orthogonality, and held-out reconstruction error. POD fitting is accounted for separately from FM, FMM, and constrained post-training.

C.2 Baselines and initialization

Table 4: Comparison methods grouped by their role in terminal alignment.

Group	Methods
Base generators	FM (Euler), FMM–LSD
Inference-time correction	FMRG-J, FM+Proj (per step)
Training-time alignment	VFM, FM-RE, ODE-Adjoint, Flow-GRPO, Direct map (ablation)
Ours	Joint training in (12)

FM is trained from a fresh initialization. FMM–LSD is initialized from the matching FM checkpoint. Direct map and our method initialize their shared map–velocity model from the matching FMM checkpoint. VFM initializes its flow map from the matching FMM–LSD checkpoint and jointly trains that map and its source adapter. FM-RE, Flow-GRPO, and ODE-Adjoint initialize from the matching FM checkpoint. Training-time alignment methods use the same benchmark terminal transform while retaining their method-specific objectives, coefficients, and internal mechanisms. FMRG-J corrects the frozen FMM–LSD sampler; FM+Proj corrects the frozen FM sampler after each generation step.

The public implementations of FMM, VFM, Flow-GRPO, and FMRG inform the adaptations below.

Table 5: Generator and terminal-alignment baselines.

Method	Reference	Adaptation used here
FM (Euler)	Flow Matching (Lipman et al., 2023)	Shared local velocity trained by conditional-OT regression; fixed-step Euler sampling.
FMM–LSD	Flow Map Matching and self-distillation (Boffi et al., 2025a;b)	Shared two-time map and velocity trained with FM and LSD; finite-difference LSD estimator below.
Direct map	Direct reward backpropagation (Clark et al., 2024)	FMM–LSD parent with a direct endpoint penalty and no map-adjoint update.
ODE-Adjoint	Neural-ODE adjoint differentiation (Chen et al., 2018)	FM parent; terminal penalty differentiated through a numerical ODE trajectory.

FM+Proj projects after each generation step: exact set projection for the toy benchmarks and eight Lagrangian iterations for the other four. The latter use primal and dual step sizes 0.1 and 1, penalty parameter 1, and maximum update norm 1 throughout. FMRG-J applies one terminal-gradient step per interval, with the shared smoothed penalty, guidance scale 1, and no gradient normalization.

Our implementation estimates the velocity–map consistency loss by finite differences. Each batch allocates 75% of its examples to diagonal FM regression and 25% to off-diagonal consistency, using finite-difference step 0.01 and consistency weight 1. This approximates the time derivative in (5); the map adjoint uses an input VJP computed by automatic differentiation. The direct terminal term and map adjoint use the same transform in (17).

Table 6: Additional alignment and inference-time comparators.

Method	Reference	Adaptation used here
VFM	Variational Flow Maps (Mammadov et al., 2026)	FMM–LSD map with a jointly trained diagonal-Gaussian source adapter.
FM-RE	Constraint-aware flow matching (Huan et al., 2026)	Late-stage stochastic velocity policy with REINFORCE terminal feedback.
Flow-GRPO	Flow-GRPO (Liu et al., 2025)	Stochastic flow transitions with grouped policy gradients, clipping, and reference KL.
FMRG-J	Flow Map Reward Guidance (Huang et al., 2026a)	Remaining-time map guidance applied to the frozen FMM–LSD parent at each interval.
FM+Proj	This study	Frozen FM sampler corrected after each step: exact set projection for toy distributions and eight Lagrangian iterations for trajectory and field tasks.

C.3 Training and evaluation

The final comparison uses training seeds $\{11, 23, 47\}$ and paired source samples across methods and schedules. Each frozen checkpoint is evaluated at uniform $K \in \{1, 4, 16, 64\}$, without schedule-specific retraining; FM-RE retains its stage-split time grid with the same number of steps. We use 4096 samples for each toy benchmark, 64 per SMD task or Stokes condition, and 128 for each of 12 Dynamic Stall conditions. The tables pair each benchmark's terminal metric defined above with SWD, using 256 fixed projections (64 for Dynamic Stall).

Table 7: Optimizer-update budgets and learning rates.

Stage and domain	Updates	Learning rate
Toy FM and FMM (each)	50,000	10^{-3}
Toy post-training	10,000	5×10^{-4}
Robot/field FM and FMM (each)	50,000	10^{-4}
Robot/field post-training	10,000	5×10^{-5}

All stages use Adam with $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon = 10^{-8}$, no weight decay or global gradient clipping, a square-root learning-rate schedule (decay fraction 0.14), and EMA (coefficient 0.999). Evaluation uses EMA parameters. All runs use FP32 with TF32 disabled.

C.4 Cost Accounting and Reproducibility

Training cost times complete optimizer updates after 30 warmup and 120 measured updates, with two fresh-process repetitions per method and seed. It excludes parent training, POD fitting, startup, and checkpoint serialization; rollout-based baselines use eight internal steps per rollout. At $K = 4$, we time complete batched samplers at fixed benchmark batch sizes, including decoding and any guidance or projection, but excluding data transfers and evaluation metrics. Model, constraint, and VJP call counts accompany the wall-clock measurements. VFM's source adapter and physical-space decoding are counted separately from map intervals.

D Extended Experimental Results

Figure 3 focuses on Six Disks, SMD9, and Dynamic Stall. Here we give their numerical results alongside Checkerboard, SMD6, and Stokes. Each table pairs sample quality with the benchmark's terminal-violation metric across sampling step counts; benchmark definitions and evaluation details appear in Appendix C.

FM+Proj uses the frozen FM sampler as its parent. FMRG-J uses the same fixed guidance setting as in Figure 3.

D.1 Cross-Schedule Benchmark Comparisons

These tables show how the quality–violation trade-off changes with sampling steps across all six benchmarks. Relative to FMM–LSD, our method reduces mean violation at every tested step count on Six Disks, SMD9, and Dynamic Stall, with the clearest simultaneous SWD gains on SMD9. The remaining tasks qualify this pattern: Checkerboard shows no uniform advantage over VFM, SMD6 has similar collision violation, and Stokes has lower initial-condition violation at the cost of higher SWD.

Cells give the mean and sample standard deviation over training seeds 11, 23, and 47. Table headers state the scale of SWD and the benchmark-specific violation. Every method is evaluated at $K \in \{1, 4, 16, 64\}$ without schedule-specific retraining. Both metrics are lower-is-better, as indicated by the table headers. For Stokes, the tables report initial-condition MAE above the 10^{-3} tolerance, while the ablation figure reports initial-condition MSE. The tables group base generators, inference-time corrections, and training-time alignment methods; the final row highlights ours. Means are displayed with up to three significant digits and standard deviations with up to two; the frozen result records retain full precision.

D.1.1 Six Disks

Table 8: **Six Disks: quality and constraint violation.**

	SWD ($\times 10^{-2}$) ↓				**Mean violation** ($\times 10^{-3}$) ↓			
Method	$K=1$	$K=4$	$K=16$	$K=64$	$K=1$	$K=4$	$K=16$	$K=64$
Base generators								
FM	72.4 (0.2)	11.4 (0.16)	6.2 (0.11)	5.9 (0.17)	748 (2.1)	57 (1.1)	5 (0.36)	3 (0.31)
FMM-LSD	5.9 (0.22)	5.8 (0.54)	5.9 (0.4)	5.9 (0.3)	8 (0.84)	7 (0.25)	3 (0.18)	3 (0.32)
Inference-time correction								
FMRG-J	163 (8)	13.5 (3.2)	6.55 (0.24)	6.38 (0.37)	460 (7.2)	17 (2.8)	0.0092 (0.0035)	0.0102 (0.003)
FM+Proj	39.8 (7.8)	10.5 (0.13)	10.7 (0.074)	10.9 (0.063)	0 (0)	0 (0)	0 (0)	0 (0)
Training-time alignment								
FM-RE	56.8 (2.7)	10 (0.18)	6.4 (0.083)	6.1 (0.15)	519 (18)	38 (0.7)	3 (0.24)	1 (0.24)
ODE-Adjoint	152 (160)	2560 (4400)	417 (670)	85.5 (94)	1440 (1700)	11300 (19000)	1420 (2300)	651 (950)
Flow-GRPO	60.4 (0.87)	10.8 (1.7)	10.3 (2.4)	10.2 (2.5)	595 (9.9)	4 (2.3)	0.01 (0.01)	0.02 (0.02)
Direct map	7 (1.3)	5.6 (0.78)	5.7 (0.53)	5.8 (0.35)	2 (0.11)	6 (0.43)	3 (0.49)	2 (0.34)
VFM	6.9 (1.4)	5.7 (0.79)	5.8 (0.48)	5.8 (0.34)	2 (0.45)	6 (0.61)	3 (0.42)	2 (0.4)
Ours	7.2 (1.7)	6 (1.1)	5.9 (1.1)	5.9 (1.1)	2 (0.13)	3 (0.49)	1 (0.26)	0.95 (0.21)

D.1.2 SMD9

Table 9: **SMD9: quality and constraint violation.**

	SWD ($\times 10^{-2}$) ↓				**Mean violation** ($\times 10^{-3}$) ↓			
Method	$K=1$	$K=4$	$K=16$	$K=64$	$K=1$	$K=4$	$K=16$	$K=64$
Base generators								
FM	10.4 (0.48)	5.3 (0.65)	5.5 (0.7)	5.7 (0.68)	49 (1.6)	24 (2.9)	28 (2.3)	30 (2.2)
FMM-LSD	4.2 (0.12)	3.7 (0.065)	3.5 (0.085)	3.6 (0.098)	20 (1.6)	16 (1.7)	14 (1.8)	15 (1.7)
Inference-time correction								
FMRG-J	4.19 (0.12)	3.72 (0.065)	3.5 (0.085)	3.54 (0.098)	19.1 (1.7)	14.6 (1.5)	12.9 (1.7)	13.3 (1.6)
FM+Proj	10.4 (0.48)	5.32 (0.65)	5.53 (0.7)	9.44 (0.59)	20.2 (1.1)	6.99 (1.3)	11 (1.2)	46.7 (1.4)
Training-time alignment								
FM-RE	15.9 (0.31)	5.5 (0.71)	5.5 (0.73)	5.7 (0.72)	67 (1.6)	25 (2.5)	27 (1.9)	29 (1.9)
ODE-Adjoint	11.5 (0.58)	5 (0.45)	4.5 (0.22)	4.7 (0.18)	54 (3.2)	25 (4.9)	21 (3.1)	23 (3.4)
Flow-GRPO	13.1 (0.37)	5.5 (0.55)	5 (0.76)	5.2 (0.74)	55 (1)	22 (3.1)	24 (2.1)	26 (2)
Direct map	3.8 (0.07)	3.6 (0.047)	3.5 (0.065)	3.5 (0.086)	13 (1.5)	16 (2)	14 (1.7)	15 (1.5)
VFM	3.8 (0.069)	3.6 (0.052)	3.5 (0.067)	3.5 (0.088)	13 (1.4)	15 (1.8)	14 (1.5)	15 (1.4)
Ours	4.1 (0.048)	3 (0.038)	2.8 (0.074)	2.9 (0.064)	14 (1.5)	11 (1.5)	9 (1.3)	10 (1.3)

D.1.3 STOKES

Table 10: **Stokes: quality and constraint violation.**

	SWD ($\times 10^{-2}$) ↓				Mean violation ($\times 10^{-3}$) ↓			
Method	$K=1$	$K=4$	$K=16$	$K=64$	$K=1$	$K=4$	$K=16$	$K=64$
Base generators								
FM	14.1 (0.73)	12.9 (0.55)	12.2 (0.43)	12 (0.43)	32 (2)	16 (0.93)	17 (1)	18 (1)
FMM-LSD	14.1 (0.13)	13 (0.15)	12.5 (0.18)	12.3 (0.22)	19 (2)	13 (0.63)	14 (0.58)	15 (0.57)
Inference-time correction								
FMRG-J	14.1 (0.13)	13 (0.15)	12.5 (0.17)	12.4 (0.22)	18.5 (1.7)	12.2 (0.6)	12.8 (0.55)	13.5 (0.54)
FM+Proj	14.1 (0.73)	12.9 (0.56)	12.3 (0.48)	12.3 (0.52)	30.9 (1.9)	15.3 (0.93)	14.6 (1.2)	13.2 (1.3)
Training-time alignment								
FM-RE	15.5 (1.1)	13.4 (0.64)	12.7 (0.53)	12.5 (0.51)	59 (5)	14 (0.79)	14 (0.71)	15 (0.74)
ODE-Adjoint	14.3 (0.74)	13.6 (0.4)	13 (0.22)	12.8 (0.21)	38 (1.7)	10 (0.22)	9 (0.72)	10 (0.78)
Flow-GRPO	14.8 (0.91)	13.6 (0.71)	12.8 (0.62)	12.6 (0.61)	57 (3.7)	12 (0.89)	12 (0.93)	13 (0.94)
Direct map	14.4 (0.1)	13.2 (0.23)	12.6 (0.22)	12.4 (0.22)	9 (1.1)	11 (1.1)	13 (0.9)	13 (0.86)
VFM	14.4 (0.073)	13.1 (0.19)	12.4 (0.24)	12.3 (0.27)	9 (1.2)	12 (1.1)	13 (0.86)	14 (0.8)
Ours	14.7 (0.12)	13.7 (0.31)	13.1 (0.27)	12.9 (0.28)	8 (1.2)	8 (1.2)	9 (1.1)	10 (1.1)

D.1.4 CHECKERBOARD

Table 11: **Checkerboard: quality and constraint violation.**

	SWD ($\times 10^{-2}$) ↓				Mean violation ($\times 10^{-4}$) ↓			
Method	$K=1$	$K=4$	$K=16$	$K=64$	$K=1$	$K=4$	$K=16$	$K=64$
Base generators								
FM	55.3 (0.41)	16.7 (0.097)	5.8 (0.092)	3.2 (0.062)	400 (12)	510 (10)	40 (4.3)	20 (1.1)
FMM-LSD	3.7 (0.27)	3.2 (0.3)	3 (0.21)	2.9 (0.076)	50 (4.3)	50 (3.1)	20 (1.8)	20 (1.4)
Inference-time correction								
FMRG-J	77.7 (11)	9.93 (2.9)	3.61 (0.51)	3.59 (0.43)	2120 (260)	96.3 (30)	0.115 (0.13)	0.045 (0.039)
FM+Proj	55.4 (0.49)	15.6 (0.14)	8.65 (0.023)	9.52 (0.024)	0 (0)	0 (0)	0 (0)	0 (0)
Training-time alignment								
FM-RE	49.2 (2)	17 (0.081)	5.9 (0.12)	3.2 (0.11)	700 (360)	320 (4.9)	30 (1.6)	10 (0.79)
ODE-Adjoint	44.9 (5.2)	24.8 (10)	18 (14)	17.4 (14)	550 (390)	690 (23)	720 (57)	740 (83)
Flow-GRPO	39.8 (4.7)	15 (2.5)	11.1 (3.4)	10.3 (3.2)	740 (170)	80 (26)	0.206 (0.16)	0.735 (0.34)
Direct map	5.6 (0.75)	3.3 (0.07)	2.9 (0.086)	2.8 (0.21)	8.1 (0.95)	40 (2)	20 (0.34)	20 (0.48)
VFM	4.9 (0.09)	2.9 (0.32)	2.6 (0.1)	2.5 (0.14)	7.07 (1.2)	40 (2.1)	20 (2.5)	20 (1.9)
Ours	5.5 (0.24)	3.4 (0.28)	3.1 (0.2)	3 (0.16)	7.93 (0.75)	40 (1.1)	20 (1.3)	10 (1.1)

D.1.5 SMD6

Table 12: **SMD6: quality and constraint violation.**

	SWD ($\times 10^{-2}$) ↓				Mean violation ($\times 10^{-4}$) ↓			
Method	$K=1$	$K=4$	$K=16$	$K=64$	$K=1$	$K=4$	$K=16$	$K=64$
Base generators								
FM	7.5 (0.18)	2.7 (0.1)	2.7 (0.12)	2.9 (0.12)	230 (10)	40 (5.7)	40 (6)	50 (7)
FMM-LSD	3.4 (0.13)	2.2 (0.016)	1.9 (0.046)	1.8 (0.058)	60 (8.2)	30 (0.72)	20 (1.3)	20 (1.8)
Inference-time correction								
FMRG-J	3.43 (0.13)	2.21 (0.016)	1.87 (0.046)	1.84 (0.058)	64.7 (8.6)	26.9 (0.39)	19 (1)	18.5 (1.5)
FM+Proj	7.49 (0.18)	2.69 (0.1)	2.77 (0.13)	7.23 (0.31)	69 (2.6)	5.99 (1.3)	10.5 (1.5)	247 (13)
Training-time alignment								
FM-RE	12.3 (0.3)	2.9 (0.1)	2.6 (0.12)	2.7 (0.1)	400 (13)	50 (7.8)	40 (5.9)	40 (6.2)
ODE-Adjoint	7.2 (0.19)	2.5 (0.074)	2.5 (0.11)	2.5 (0.1)	220 (9.6)	40 (5)	30 (4.5)	40 (5.4)
Flow-GRPO	9.7 (0.15)	2.7 (0.14)	2.5 (0.071)	2.6 (0.11)	280 (13)	30 (1.5)	30 (4.4)	40 (7.5)
Direct map	3.1 (0.14)	2.2 (0.018)	1.9 (0.052)	1.8 (0.066)	50 (7)	30 (1.8)	20 (2.3)	20 (2.4)
VFM	3.2 (0.14)	2.2 (0.023)	1.8 (0.055)	1.8 (0.07)	50 (7.2)	30 (1.5)	20 (1.5)	20 (1.6)
Ours	3.3 (0.14)	2.1 (0.033)	1.7 (0.061)	1.7 (0.069)	50 (8.9)	30 (1.7)	20 (1.6)	20 (1.4)

D.1.6 DYNAMIC STALL

Table 13: **Dynamic Stall: quality and constraint violation.**

	SWD ($\times 10^{-2}$) ↓				Mean violation ($\times 10^{-6}$) ↓			
Method	$K=1$	$K=4$	$K=16$	$K=64$	$K=1$	$K=4$	$K=16$	$K=64$
Base generators								
FM	3.7 (0.036)	3.6 (0.068)	3.8 (0.058)	3.9 (0.055)	11.3 (0.41)	8.98 (0.65)	9.54 (0.61)	9.78 (0.61)
FMM-LSD	3.4 (0.1)	3.5 (0.1)	3.6 (0.09)	3.7 (0.087)	8.62 (0.81)	8.38 (0.58)	8.72 (0.55)	8.84 (0.55)
Inference-time correction								
FMRG-J	3.43 (0.1)	3.53 (0.1)	3.65 (0.09)	3.69 (0.087)	8.62 (0.81)	8.38 (0.58)	8.71 (0.55)	8.84 (0.55)
FM+Proj	3.7 (0.036)	3.61 (0.068)	3.83 (0.058)	3.91 (0.055)	11.3 (0.41)	8.98 (0.65)	9.54 (0.61)	9.78 (0.61)
Training-time alignment								
FM-RE	4.9 (0.12)	3.6 (0.074)	3.8 (0.065)	3.9 (0.062)	21.8 (2.8)	8.84 (0.63)	9.45 (0.6)	9.69 (0.6)
ODE-Adjoint	3.6 (0.034)	3.5 (0.071)	3.7 (0.063)	3.8 (0.06)	10.6 (0.41)	8.38 (0.58)	8.96 (0.55)	9.2 (0.55)
Flow-GRPO	3.6 (0.034)	3.6 (0.062)	3.8 (0.05)	3.9 (0.047)	9.83 (0.28)	8.03 (0.55)	8.61 (0.54)	8.84 (0.54)
Direct map	3.4 (0.096)	3.5 (0.098)	3.6 (0.086)	3.6 (0.083)	7.77 (0.64)	7.93 (0.5)	8.31 (0.47)	8.45 (0.46)
VFM	3.3 (0.087)	3.4 (0.095)	3.6 (0.086)	3.6 (0.082)	7.63 (0.59)	7.86 (0.48)	8.27 (0.45)	8.42 (0.45)
Ours	3.3 (0.078)	3.4 (0.09)	3.6 (0.082)	3.6 (0.081)	7.2 (0.53)	7.33 (0.39)	7.71 (0.37)	7.84 (0.36)

Examples in the main figure. The qualitative panels in Figure 3 use saved seed-11 outputs from the selected checkpoints. Six Disks uses all 4,096 paired draws at each schedule. Dynamic Stall and SMD9 use the fixed first task and sample at all four step counts, without outcome-based selection. Each paired display shares spatial bounds or a field color scale; the aggregate metrics use all evaluated tasks and samples.

Dynamic Stall residual metric. The main figure reports the positive part of the native physical residual MSE minus the 2.5×10^{-9} reference tolerance. The qualitative panels show normalized pressure fields on a shared color scale. Every evaluated sample exceeds the tolerance, so the unthresholded residual means equal the tabulated excess values plus 2.5×10^{-9}. The frozen POD codec also crosses this threshold on all $1{,}024$ held-out validation reconstructions (mean excess residual 1.84×10^{-7}). These are relative residual comparisons, not evidence of physical feasibility.

Additional benchmark behavior. Checkerboard does not yield a uniform improvement over VFM. At $K = 64$ our mean violation is lower, whereas its SWD is higher; at $K = 4, 16$ the recorded violation means are similar. On SMD6, our SWD is slightly lower at composed schedules while the mean collision violation is nearly unchanged. Thus this robot comparison primarily tests distributional quality under composition.

On Stokes, ours has lower initial-condition MSE than VFM at all tested step counts but higher SWD. This case therefore does not show dominance in both quality and terminal error.

FM+Proj reaches zero measured Checkerboard violation but raises SWD relative to FM at $K = 16, 64$. It lowers SMD6 violation at $K = 4, 16$, yet worsens both metrics at $K = 64$; on Dynamic Stall it matches FM at the displayed precision.

Map-adjoint accuracy on frozen models. We freeze the pre-constraint FMM–LSD checkpoint for each toy benchmark and training seed, then compare its composed terminal prediction and transported terminal gradient with the ODE flow and adjoint of its *own* diagonal velocity. This isolates map approximation from subsequent constraint training. For each of three seeds, $1{,}024$ source–time pairs use the same smoothed terminal penalty on both paths. The ODE reference uses 128 RK4 steps; halving that count changes its aggregate adjoint by at most 0.6% across the six runs. Figure 5 shows a median absolute terminal error of about 0.015 in both benchmarks. Among pairs where both gradients are nonzero, the median cosine is 0.994 on Six Disks and 0.972 on Checkerboard, so the transported directions are usually aligned. Nevertheless, 17% and 22% of those pairs point opposite to the ODE adjoint; the respective reference-active sets also contain 34 and 40 pairs missed by the map adjoint. Thus the map provides an informative but imperfect direction, and small endpoint error alone does not ensure accurate gradient transport.

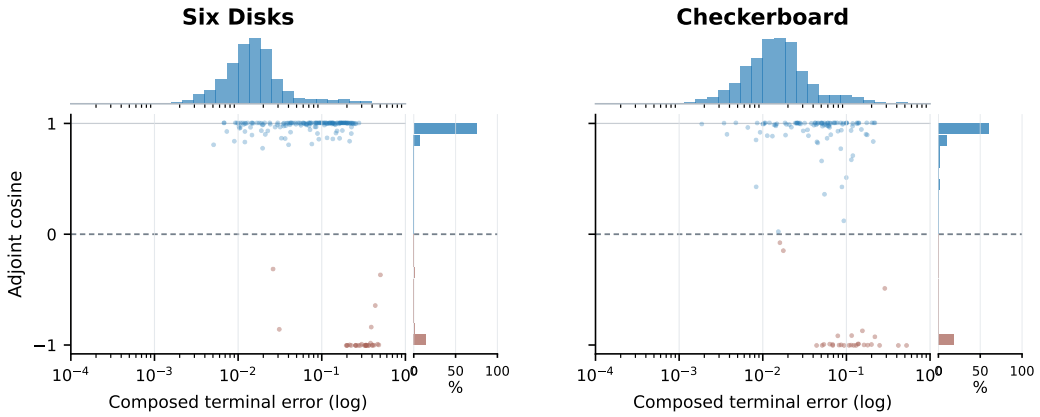


Figure 5: **Frozen FMM terminal prediction and task-adjoint direction.** Six Disks and Checkerboard each pool three pre-constraint FMM–LSD checkpoints (seeds $11, 23, 47$; $1{,}024$ source–time pairs per seed). The upper histograms include all pairs and show absolute error between the composed map endpoint and the RK4 reference of the same model's diagonal velocity. The lower panels show the cosine between their terminal-penalty adjoints only when both are nonzero; the right-hand histograms show the percentage distribution of those same cosines. Red denotes negative cosine. Missing or spurious nonzero adjoints are excluded from the cosine panels and counted in the saved records. The logarithmic horizontal scale does not truncate the observations.