SoF: Second-Order Flow Matching with Learning Acceleration

Anonymous authors
Paper under double-blind review

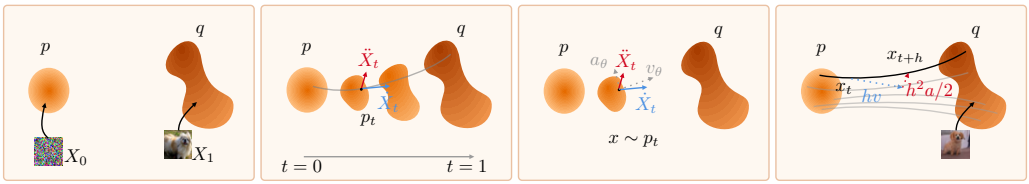
Abstract

Continuous-time generative models are bottlenecked by the number of function evaluations (NFE). We introduce Second-Order Flow Matching (SoF), which equips flow-based models with an explicit acceleration field via the squared-generator identity $\mathcal{H}_t = \mathcal{G}_t^2 + \partial_t \mathcal{G}_t$ and a phase-space equivalence, realized by a dual-head backbone enabling single-NFE second-order sampling at $< 2\%$ overhead. Combined with a schedule-matched sampling formula and an endpoint-refined inference grid, second-order sampling with the trained acceleration head yields large gains over same-model first-order solvers: 50-NFE FID drops from 21.24 to 2.48 on CIFAR-10 and from 32.03 to 3.40 on CelebA-HQ with 2nd-order Heun, reaching 2.41 at 100 NFE on CIFAR-10 without distillation, while DPM-Solver++ (2V) halves the 20-NFE FID relative to the best first-order solver (3.31 versus 7.16). At very low NFE ($N \leq 4$), first-order solvers remain competitive. Inpainting on CelebA-HQ and LSUN-Church confirms the benefit extends to conditional generation. Code: `https://ANONYMOUS.net/`.



Figure 1: **Generated samples.** Datasets (left to right): CIFAR-10 (32×32), CelebA-HQ (64×64, 128×128), LSUN-Classroom and LSUN-Church (128×128).

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.



(a) Data **(b)** Path design **(c)** Training **(d)** Sampling

Figure 2: **Second-Order Flow Matching framework.** (a) Source and target distributions. (b) Affine probability path with a convex polynomial schedule, yielding velocity and non-degenerate acceleration targets $(\dot{X}_t, \ddot{X}_t)$. (c) Dual-head training with combined loss $\mathcal{L}_{\text{vel}} + \lambda(e)\mathcal{L}_{\text{acc}}$. (d) Single-NFE second-order Euler sampling $x_{t+h} = x_t + hv + \frac{h^2}{2}a$.

1 Introduction

Continuous-time generative models, whether they reverse a diffusion SDE (Ho et al., 2020; Song et al., 2021b) or integrate a deterministic velocity field (Lipman et al., 2023; Tong et al., 2024), share a single practical bottleneck: the number of function evaluations (NFE). The root cause is structural. The learned score or velocity captures only instantaneous drift and discards temporal curvature, so higher-order accuracy must be reconstructed through repeated network queries at every step (Lu et al., 2022a).

Generator Matching (Holderrieth et al., 2025) formalizes the velocity field $\mathcal{G}_t$ as a principled training target, yet this first-order abstraction is inherently blind to curvature. The standard CondOT schedule $(\mu_t = t, \sigma_t = 1 - t)$ compounds the problem: its linear path yields a vanishing acceleration target, leaving no second-order signal to learn. Existing remedies, from trajectory straightening (Liu et al., 2023) and distillation (Salimans & Ho, 2022; Song et al., 2023) to multi-stage solvers (Butcher, 2016), all pay for curvature through repeated queries at sampling time. This motivates a fundamental question:

Can a generative model maintain an explicit second-order representation of its dynamics, so that higher-order accuracy no longer demands multiple NFE per step?

We answer affirmatively with Second-Order Flow Matching (SoF). Our contributions are:

- **Theory.** Extending Generator Matching (Holderrieth et al., 2025) to second order, we derive the squared-generator identity $\mathcal{H}_t = \mathcal{G}_t^2 + \partial_t \mathcal{G}_t$ (Theorem 2.1) and a phase-space equivalence (Dupont et al., 2019) that reduces second-order dynamics to jointly learning (v_θ, a_θ), without Hessians (Theorem 2.2).
- **Algorithm.** Progressive acceleration weighting and zero-module initialization stabilize a dual-head backbone, enabling a single-NFE Taylor step at $\mathcal{O}(h^3)$ local truncation error and two DPM-Solver++ variants: the zero-acceleration degenerate (2V, $\lambda = 0$) and the acceleration-aware (2A).
- **Empirics.** On CIFAR-10 (Krizhevsky, 2009) and CelebA-HQ (Karras et al., 2018), we address two inference-time issues, a mismatch between the DPM-Solver++ formula and the polynomial schedule ($75.31 \rightarrow 4.53$ FID at 8 NFE) and a uniform-grid corrector distortion ($14.40 \rightarrow 4.22$ at 50 NFE), and show acceleration supervision yields consistent gains at moderate budgets (2.48 vs. 2.58 at 50 NFE); at $N \leq 4$ first-order solvers remain competitive, per our remainder-stability bound.

From insight to algorithm. Concretely, SoF trains a shared-decoder dual-head network to predict (v_θ, a_θ) along a polynomial convex path whose second derivative supplies analytic acceleration targets; the acceleration term upgrades Euler/Heun/RK4 to second or higher order at one extra head readout ($< 2\%$ overhead, Figure 2) and plugs into exponential integrators as the schedule-matched DPM-Solver++ variants 2V/2A. Section 2 develops the framework with derivations deferred to the appendices, Section 3 reports CIFAR-10 and CelebA-HQ results under a unified 50k/30k FID protocol (LSUN (Yu et al., 2015) results in the appendices), and Section 4 positions SoF among fast samplers, distillation, and higher-order solvers.

2 SECOND-ORDER GENERATOR

We build on the theory of continuous-time Markov processes and their infinitesimal generators. For a self-contained treatment, covering Polish state spaces, the Kolmogorov forward equation, and the generator formalism for deterministic flows, see Appendix B.

2.1 MOTIVATION

The first-order generator $\mathcal{G}_t$ provides a linear approximation to the infinitesimal evolution of a Markov process (Feller, 1952). For a smooth test function f (see Appendix B.3):

$$\langle p_{t+h|t}, f\rangle(x) = f(x) + h[\mathcal{G}_t f](x) + o(h). \tag{1}$$

This approximation underlies the Kolmogorov forward equation (Kallenberg, 2021) and Generator Matching (Holderrieth et al., 2025), but as a *tangent-line approximation* it discards curvature, incurring a systematic $\mathcal{O}(h^2)$ local error. As in classical mechanics, velocity alone forces solvers to reconstruct curvature through repeated queries (Heun, RK4), whereas with acceleration directly available a single Taylor step $x(t+h) \approx x(t) + hv(t) + \frac{1}{2}h^2 a(t)$ attains $\mathcal{O}(h^3)$ local accuracy at one evaluation. Figure 3 validates this on a 2D checkerboard flow: second-order trajectories track the ground truth, while first-order Euler drifts.

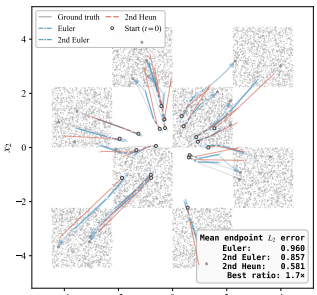


Figure 3: **Why acceleration matters.** Trajectories at NFE $= 5$: ground truth vs. first-order vs. second-order.

This motivates a *quadratic expansion* of the transition action:

$$\langle p_{t+h|t}, f\rangle(x) = f(x) + h[\mathcal{G}_t f](x) + \frac{h^2}{2}[\mathcal{H}_t f](x) + o(h^2), \tag{2}$$

where the second-order generator $\mathcal{H}_t$ encodes the curvature absent from $\mathcal{G}_t$.

2.2 DEFINITION AND DERIVATION

Definition. The second-order generator $\mathcal{H}_t$, defined on test functions $f \in \mathcal{T}$ (see Appendix B.3), is introduced as the second derivative of the transition action:

$$[\mathcal{H}_t f](x) := \frac{\partial^2}{\partial h^2}\bigg|_{h=0} \langle p_{t+h|t}, f\rangle(x), \tag{3}$$

provided the limit exists uniformly. Theorem 2.1 expresses $\mathcal{H}_t$ in terms of $\mathcal{G}_t$, and Theorem 2.2 reduces the second-order dynamics to a learnable acceleration field in an augmented phase space.

Theorem 2.1 (Second-Order Generator Formula)**.** *Assume continuity of the map $s \mapsto \mathcal{G}_s$ and the interchangeability of limits with expectations (see Assumption A.7). Then the second-order generator satisfies*

$$[\mathcal{H}_t f](x) = [\mathcal{G}_t^2 f](x) + [\partial_t \mathcal{G}_t f](x), \tag{4}$$

where $\mathcal{G}_t^2$ denotes the composition $\mathcal{G}_t \circ \mathcal{G}_t$.

This identity is the time-inhomogeneous counterpart of the classical second-order semigroup expansion (Kallenberg, 2021), here made operational for flow matching via the phase-space lift below. A complete derivation is given in Appendix C.1; intuitively, $\mathcal{H}_t$ decomposes into a compounded evolution $\mathcal{G}_t^2$ (analogous to convective acceleration) and an explicit time-variation term $\partial_t \mathcal{G}_t$, and neither alone suffices (Figure 10).

Theorem 2.2 (Phase-Space Equivalence for Deterministic Flows)**.** *Let $\mathcal{H}_t$ be the second-order generator for deterministic flows, defined as*

$$[\mathcal{H}_t f](x) = u_t^\top(x)\nabla^2 f(x) u_t(x) + \nabla f^\top(x)\alpha_t(x), \tag{5}$$

where $u_t : \mathbb{R}^d \to \mathbb{R}^d$ is a time-dependent velocity field, and $\alpha_t := (\nabla u_t)u_t + \partial_t u_t$ is an acceleration field. Let $\tilde{\mathcal{G}}_t$ be the expanded first-order generator on phase space $\mathbb{R}^d \times \mathbb{R}^d$:

$$[\tilde{\mathcal{G}}_t g](x, v) = \nabla_x g(x, v)^\top v + \nabla_v g(x, v)^\top \alpha_t(x). \tag{6}$$

For any test function $f \in C_c^\infty(\mathbb{R}^d)$, define its lifting $\bar{f}(x, v) = f(x)$ (independent of v). Then the second-order generator in the original space equals the squared expanded generator restricted to $v = u_t(x)$:

$$[\mathcal{H}_t f](x) = [\tilde{\mathcal{G}}_t^2 \bar{f}](x, v)\Big|_{v=u_t(x)}. \tag{7}$$

Proof sketch. For deterministic flows, applying $\mathcal{G}_t$ twice and adding the time derivative yields equation 5 with α_t equal to the material acceleration of the flow; the full derivation is in Appendix C.2.

From theory to practice. Theorem 2.2 eliminates the Hessian $\nabla^2 f$ in equation 5: second-order dynamics in $\mathbb{R}^d$ are exactly captured by a first-order evolution in phase space $\mathbb{R}^d \times \mathbb{R}^d$ using only first derivatives (full computation in Appendix C.3). It therefore suffices to learn the concrete acceleration field $\alpha_t(x)$ alongside $u_t(x)$. For affine paths $X_t = \mu_t X_1 + \sigma_t X_0$, the conditional acceleration $\ddot{X}_t = \ddot{\mu}_t X_1 + \ddot{\sigma}_t X_0$ provides explicit supervision targets, making the path schedule the key determinant of second-order supervision quality, a point we address next.

Implicit vs. explicit curvature. Theorem 2.1 also implies that the acceleration field already exists implicitly in any smooth velocity field: the classical Heun corrector reconstructs α_t through two velocity queries per step, whereas a model that directly outputs α_t attains the same leading-order accuracy at one NFE per step. The full expansion and a matching argument for multi-stage solvers appear in Appendix D.2.

2.3 Path Design

Affine paths and acceleration targets. Given $X_0 \sim p_0$, $X_1 \sim p_1$, the affine path and its derivatives are

$$X_t = \mu_t X_1 + \sigma_t X_0, \quad \dot{X}_t = \dot{\mu}_t X_1 + \dot{\sigma}_t X_0, \quad \ddot{X}_t = \ddot{\mu}_t X_1 + \ddot{\sigma}_t X_0 \quad (\mu_0{=}\sigma_1{=}0,\ \mu_1{=}\sigma_0{=}1), \tag{8}$$

so non-degenerate supervision for $\alpha_t(x)$ exists iff $\ddot{\mu}_t, \ddot{\sigma}_t \not\equiv 0$.

Proposition 2.1 (Second-order degeneracy of CondOT)**.** *Under the CondOT scheduler ($\mu_t = t, \sigma_t = 1 - t$), the conditional acceleration target satisfies $\ddot{X}_t = 0$ for all $t \in [0, 1]$; thus no non-degenerate second-order information can be extracted from the affine path.*

Polynomial convex scheduler. To obtain non-degenerate acceleration targets, we propose the polynomial convex scheduler:

$$\mu_t = t^n, \quad \sigma_t = 1 - t^n, \quad n \geq 2, \tag{9}$$

For $n = 2$ the second derivatives are constant ($\ddot{\mu}_t = 2, \ddot{\sigma}_t = -2$), providing uniform acceleration supervision; $n > 2$ lets $\ddot{\mu}_t \to 0$ near $t = 0$, while $n < 2$ introduces singularities. Thus $n = 2$ balances signal uniformity and numerical stability, and empirically yields the lowest FID across all training epochs (Table 6 and Figure 9, Appendix D.1). Uniform time sampling $t \sim U(0, 1)$ oversamples the quasi-stationary regime near $t = 0$; we address this with polynomial-aware resampling in Section 2.5.

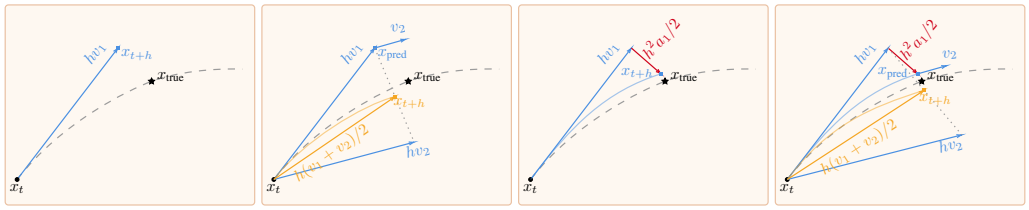
2.4 Loss Functions

Combined velocity-acceleration loss. The joint objective matches both orders of the path dynamics:

$$\mathcal{L}_{\text{combined}} = \mathcal{L}_{\text{vel}} + \lambda(e)\mathcal{L}_{\text{acc}}, \quad \mathcal{L}_{\text{vel}} = \mathbb{E}\left[\|v_\theta(X_t, t) - \dot{X}_t\|^2\right], \quad \mathcal{L}_{\text{acc}} = \mathbb{E}\left[\|a_\theta(X_t, t) - \ddot{X}_t\|^2\right], \tag{10}$$

with $\lambda(e)$ an epoch-dependent warmup weight (Section 2.5).

Velocity-acceleration conversion. Given a velocity prediction $v_\theta(X_t, t)$, the endpoints X_0, X_1 are recovered analytically and a_t follows in closed form; the explicit formulas are deferred to Appendix D.1.



(a) 1st-order Euler **(b)** 1st-order Heun **(c)** 2nd-order Euler **(d)** 2nd-order Heun

Figure 4: **Geometric illustration of the four ODE solvers.** (a) 1st-order Euler steps along the velocity hv_1 only (1 NFE). (b) 1st-order Heun evaluates v_2 at the predicted point and averages $\frac{h}{2}(v_1 + v_2)$ (2 NFE). (c) 2nd-order Euler adds the acceleration correction $\frac{h^2}{2}a_1$, landing closer to x_{true} (1 NFE). (d) 2nd-order Heun uses the acceleration-aware predictor then averages velocities, achieving the best accuracy (2 NFE).

2.5 Training Strategies

Joint training introduces three challenges: inefficient uniform time sampling, early gradient interference between objectives, and instability of a randomly initialized acceleration head.

Time sampling and inference grid. Under $\mu_t = t^n$, uniform $t \sim U(0,1)$ overweights the quasi-stationary regime near $t = 0$; we instead draw $u \sim U(0,1)$, $t = u^{1/n}$ so that $\mu_t = u$ is uniform. At inference we use the endpoint-refined cosine grid $t_i = \frac{1}{2}(1 - \cos(\pi i/N))$, which clusters steps near both endpoints; no reweighting is needed since the model predicts $(\dot{X}_t, \ddot{X}_t)$ for all t.

Progressive acceleration weighting. To prevent early-stage gradient interference between the velocity and acceleration objectives, we apply a cosine warmup to the acceleration weight:

$$\lambda(e) = \begin{cases} \lambda_{\text{target}} \cdot \frac{1}{2}\left(1 - \cos(\pi e / E_w)\right), & e < E_w, \\ \lambda_{\text{target}}, & e \geq E_w, \end{cases} \tag{11}$$

with warmup period E_w (default 15% of epochs); the acceleration head is further stabilized by zero-module initialization (Appendix D.3).

2.6 Sampling Algorithms

One forward pass, two quantities. At any state (x_t, t) a single forward pass returns both the velocity and the acceleration, $(v_\theta(x_t, t), a_\theta(x_t, t))$. Classical solvers use only v_θ; SoF additionally exploits a_θ to raise a fixed-step integrator's accuracy without extra network queries.

From first- to second-order Euler. Consider a step of size h from t to $t + h$. Truncating the Taylor expansion of the true trajectory at third order gives

$$x_{t+h} = x_t + hv_t + \frac{h^2}{2}a_t + \mathcal{O}(h^3), \tag{12}$$

where the acceleration term $\frac{h^2}{2}a_t$ is exactly the curvature that plain Euler discards. Because a_t comes from the same forward pass, this update costs 1 NFE per step—the same as first-order Euler—yet its local error drops from $\mathcal{O}(h^2)$ to $\mathcal{O}(h^3)$.

Adaptive and exponential-integrator variants. For completeness we also evaluate a velocity-only adaptive DOPRI5 (Dormand & Prince, 1980) (Taylor corrections would break its embedded error estimator) and two 1-NFE DPM-Solver++ (Lu et al., 2022a) variants: 2V ($\lambda = 0$) disables the acceleration head, while 2A ($\lambda = 0.3$) feeds it into the exponential-integrator update (Algorithms 1–2, Appendix D.4). One caveat: at $N \leq 4$ large steps break the dominance of the leading Taylor term, so the correction can amplify propagated error (it raises consistency order without enlarging the stability region (Hairer & Wanner, 1996)) and first-order Euler remains preferable (Appendix D.3); step counts for all solvers are in Appendix D.2.

Table 1: **Computational Overhead.** FLOPs and wall-clock time of first- vs. second-order UNets (RTX A6000, PyTorch 2.8). Shared-decoder SoF incurs $< 2\%$ overhead.

Model	CIFAR-10 (32×32, batch 64)				CelebA-HQ (64×64, batch 32)			
	Params (M)	GFLOPs	Fwd (ms)	Train (ms)	Params (M)	GFLOPs	Fwd (ms)	Train (ms)
1st-order UNet	55.68	1296.38	92.32	202.89	131.23	1607.91	154.01	377.91
2nd-order (shared dec.)	55.69 (+0.02%)	1296.92 (+0.04%)	93.42 (+1.19%)	206.48 (+1.77%)	131.23 (+0.00%)	1608.45 (+0.03%)	155.46 (+0.94%)	384.06 (+1.63%)
2nd-order (unshared dec.)	89.49 (+60.72%)	2148.54 (+65.73%)	152.81 (+65.52%)	334.42 (+64.83%)	210.43 (+60.35%)	2750.13 (+71.04%)	257.50 (+67.20%)	624.09 (+65.14%)

Table 2: **First-order vs. second-order flow matching.** FID (↓) of first- and second-order ODE solvers and DPM-Solver++ variants on CIFAR-10 and CelebA-HQ; (SoF) rows sample our dual-head model with acceleration disabled. RK4 at 50 NFE uses 48 steps (NFE must be a multiple of four). †Schedule-matched prediction $\hat{x}_1 = x_t + \frac{1-t^2}{2t} v_\theta$ (Appendix D.4); unmarked DPM rows use the CondOT formula.

Sampling	Conditional CIFAR-10 (32×32, 900 epochs, 50k samples)						Unconditional CelebA-HQ (64×64, 500 epochs, 30k samples)					
	2 NFE	4 NFE	8 NFE	20 NFE	50 NFE	100 NFE	2 NFE	4 NFE	8 NFE	20 NFE	50 NFE	100 NFE
1st Euler	102.70	29.30	13.53	7.16	4.97	4.50	79.94	33.88	13.93	6.47	3.66	2.74
1st Heun	421.22	296.90	138.98	54.17	14.40	4.77	372.67	213.68	95.88	45.19	14.49	4.28
1st RK4	–	127.50	69.15	22.05	4.73	4.20	–	125.68	54.54	19.64	4.37	2.44
Ours: ODE solvers												
← SoF dual-head model (acceleration disabled), sampled by 1st-order solvers; vs. the rows above. Subscripts: relative FID change (green = improvement).												
1st Euler	309.96 (-201.81%)	168.59 (-475.39%)	52.03 (-284.55%)	6.97 (+2.65%)	2.52 (+49.30%)	2.68 (+40.44%)	357.22 (-346.86%)	244.43 (-621.46%)	92.66 (-565.18%)	37.78 (-483.93%)	5.89 (-60.93%)	2.50 (+8.76%)
1st Heun	366.94 (+12.89%)	266.14 (+10.36%)	108.54 (+21.90%)	16.52 (+69.50%)	21.24 (-47.50%)	8.22 (-72.33%)	396.32 (-6.35%)	321.69 (-50.55%)	34.44 (+64.08%)	48.55 (-7.44%)	32.03 (-121.05%)	15.80 (-269.16%)
1st RK4	–	288.49 (-126.27%)	115.17 (-66.55%)	7.03 (+68.12%)	4.32 (+8.67%)	3.93 (+6.43%)	–	302.69 (-140.84%)	32.94 (+39.60%)	14.39 (+26.73%)	10.47 (-139.59%)	4.45 (-82.38%)
← vs. the SoF 1st-order rows above (pairwise).												
2nd Euler	125.97 (+59.36%)	44.13 (+73.82%)	17.83 (+65.73%)	6.66 (+4.45%)	3.61 (-43.25%)	2.71 (-1.12%)	95.58 (+73.24%)	45.68 (+81.31%)	17.75 (+80.84%)	6.51 (+82.77%)	3.46 (+41.26%)	2.75 (-10.00%)
2nd Heun	466.67 (-27.18%)	298.46 (-12.14%)	118.86 (-9.51%)	13.29 (+19.55%)	2.48 (+88.32%)	2.41 (+70.68%)	439.83 (-10.98%)	330.92 (-2.87%)	112.61 (-226.97%)	24.47 (+49.60%)	3.40 (+89.38%)	2.56 (+83.80%)
2nd RK4	–	269.84 (+6.46%)	145.71 (-26.52%)	17.06 (-142.67%)	2.57 (+40.51%)	2.43 (+38.17%)	–	265.90 (+12.15%)	134.49 (-308.29%)	18.42 (-28.01%)	2.72 (+74.02%)	2.53 (+43.15%)
Ours: DPM-Solver++ variants												
← vs. best 1st-order per NFE (Euler for NFE≤20, RK4 for NFE≥50).												
DPM++ (2V)	154.57 (-50.51%)	218.91 (-647.13%)	74.96 (-454.03%)	3.31 (+53.77%)	2.50 (+47.15%)	2.47 (+41.19%)	301.75 (-277.47%)	215.85 (-537.10%)	68.54 (-392.03%)	3.57 (+44.82%)	2.90 (+20.77%)	3.04 (-24.59%)
DPM++ (2A)	423.80 (-312.67%)	203.08 (-593.11%)	34.40 (-154.25%)	7.30 (-1.96%)	3.63 (+23.26%)	2.69 (+35.95%)	336.21 (-320.58%)	159.33 (-370.28%)	66.43 (-376.88%)	6.70 (-3.55%)	3.45 (+5.74%)	2.73 (-11.89%)
← Schedule-matched counterparts vs. their CondOT rows (Appendix D.4).												
DPM++ (2V)†	428.35 (-177.12%)	26.58 (+87.86%)	4.53 (+93.96%)	2.67 (+19.34%)	2.45 (+2.00%)	2.44 (+1.21%)	401.39 (-33.02%)	328.74 (-52.30%)	27.71 (+59.57%)	2.80 (+21.57%)	2.95 (-1.72%)	3.05 (-0.33%)
DPM++ (2A)†	383.28 (+9.56%)	150.51 (+25.88%)	13.54 (+60.64%)	6.25 (+14.38%)	3.52 (+3.03%)	2.84 (-5.58%)	437.41 (-30.10%)	444.97 (-179.27%)	62.04 (+6.61%)	6.20 (+7.46%)	3.39 (+1.74%)	2.79 (-2.20%)

Extending the correction. The same acceleration-augmented predictor lifts multi-stage solvers (Figure 4, panels (c)–(d) vs. (a)–(b)): 2nd-Heun (2 NFE/step) predicts with equation 12 and averages velocities, $x_{t+h} = x_t + \frac{h}{2}(v_1 + v_2)$; the acceleration enters only the predictor, since adding it to the corrector would double-count curvature already captured by the two-point evaluation. 2nd-Euler and 2nd-Heun both reach $\mathcal{O}(h^3)$ local error, while 2nd-RK4 (4 NFE/step) augments its midpoint stages with the acceleration for $\mathcal{O}(h^5)$.

3 Experiments

Setup. Figure 1 shows representative generations on all datasets. We use a shared-decoder dual-head UNet (Appendix D.3) with $< 2\%$ extra FLOPs (Table 1); the acceleration head is zero-initialized, all runs use the polynomial schedule ($n = 2$), 15%-epoch cosine warmup of $\lambda(e)$, and classifier-free guidance. First-order references comprise an independently trained model (gray curves) and our dual-head model with acceleration disabled ($\lambda = 0$), isolating the curvature gain under identical parameters; component ablations are in Table 10.

Evaluation protocol. FID is computed on 50k samples (CIFAR-10) / 30k (CelebA-HQ) with a fixed pipeline; the bare 1st-order rows use an independently trained model, the (SoF) 1st-order rows sample our dual-head model with acceleration disabled, and the 2nd-order and DPM-Solver++ rows use the same dual-head model, isolating the sampling algorithm and acceleration supervision. Endpoint-refined cosine-grid and schedule-matched re-evaluations are reported in Appendices D.4 and D.5.

FID Comparison. Figure 5 reports the throughput-FID Pareto fronts on CIFAR-10. Three effects drive Table 2: (i) the inference grid, since uniform grids distort multi-step correctors at the endpoints (first-order Heun 14.40 at 50 NFE, restored to 4.22 by the cosine grid; Appendix D.5); (ii) schedule-matched formulas, where the polynomial data prediction of Appendix D.4 improves 8-NFE FID from 75.31 to 4.53; and (iii) acceleration supervision, read pairwise against the SoF 1st-order rows: second-order sampling wins at moderate-to-high budgets (2nd Heun 2.48 at 50 NFE, 2.41 at 100

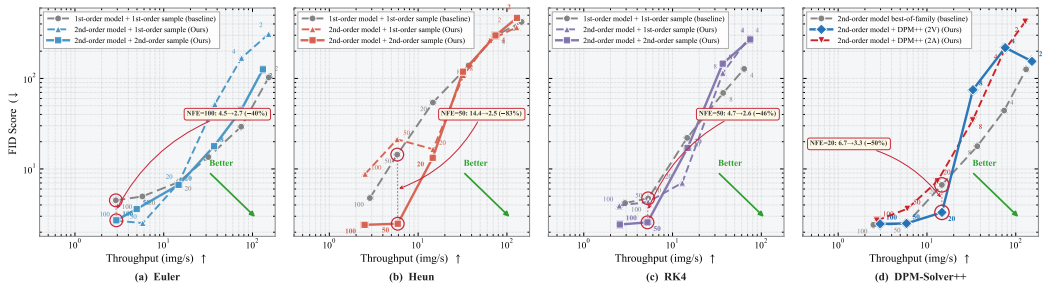


Figure 5: **Throughput-FID trade-off on CIFAR-10.** (a)–(c): first-order baseline (gray, dashed) vs. our dual-head model sampled with first-order (colored, dashed) or second-order (colored, solid) Euler/Heun/RK4 at NFE $\in \{2, 4, 8, 20, 50, 100\}$; red callouts mark the largest gain. (d): DPM-Solver++ (2V) vs. (2A) against the best of (a)–(c) (gray, dashed), annotated with the 20-NFE FID gap. Throughput on an RTX 3090; second-order curves include the $< 2\%$ dual-head overhead (Table 1).

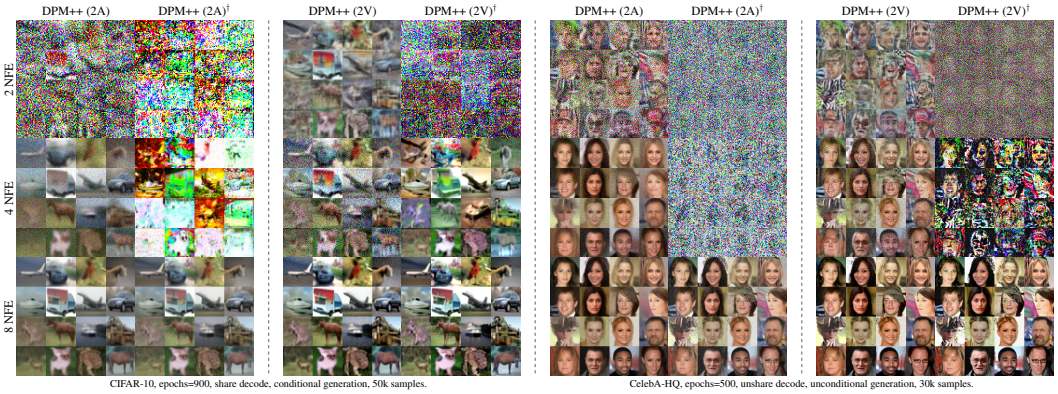


Figure 6: **DPM-Solver++ (2A) vs. (2V) at low NFE: CondOT vs. schedule-matched (†) formulas.** Rows: NFE $\in \{2, 4, 8\}$; columns: 2A ($\lambda = 0.3$) and 2V ($\lambda = 0$) under each formula. Left: CIFAR-10 (32×32, conditional); right: CelebA-HQ (64×64, unconditional). The matched prediction resolves structure already at NFE $= 2$, where CondOT yields near-noise; the formulas converge by NFE $= 8$ (Appendix D.4).

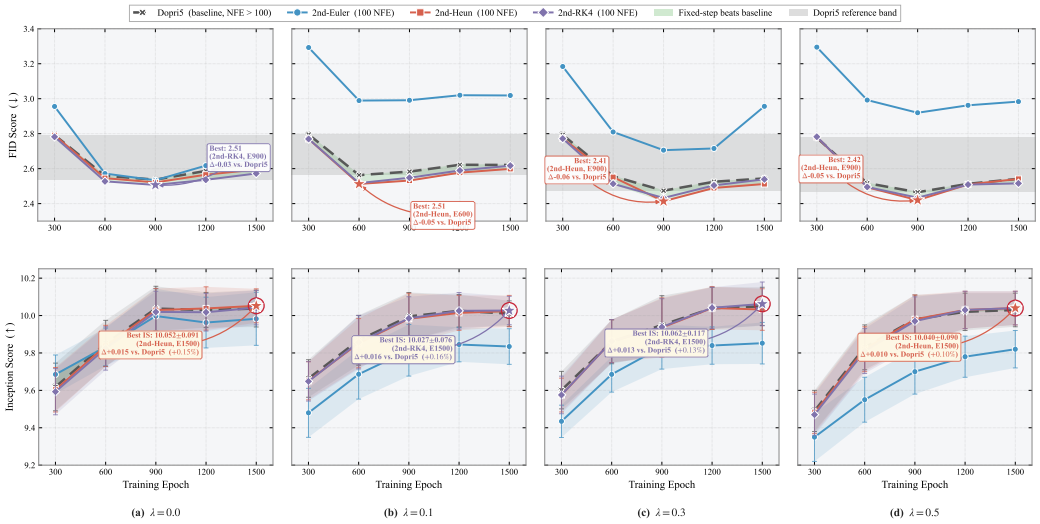
NFE), and DPM++ (2V) beats the best first-order solver per NFE from 20 NFE onward, while at very low NFE first-order sampling remains competitive.

With the matched formula (Appendix D.4), 2V attains 4.53 at 8 NFE and 2.67 at 20 NFE, the best low-budget entries, while 2A trails 2V at low NFE and converges at high NFE (Appendix D.5); Figure 6 contrasts both formulas at NFE$\leq$8, where the matched prediction already resolves structure at NFE$=$ 2 while CondOT still produces noise. Table 3 decomposes the 50-NFE result across schedule and supervision; extended results are in Appendix E.

To isolate acceleration supervision from the grid and formula effects, Table 3 fixes the cosine grid and crosses training schedule (CondOT vs. polynomial) with supervision (first-order vs. dual-head). At 50 NFE the four Heun cells span only 0.17 (2.42 vs. 2.41): high-budget gains are grid-dominated, whereas supervision matters at lower budgets—at 20 NFE Heun (SoF 13.41 vs. 17.75/21.28) and at 50 NFE Euler, where the CondOT-schedule models (3.45/3.45) beat 3.58 and the acceleration-free polynomial outlier (7.64) reflects first-order Euler's endpoint sensitivity on the cosine grid.

Table 3: **Attribution of sampling gains on CIFAR-10 (50k, cosine grid).** Rows: training schedule and supervision; columns: sampling method and NFE budget. Uniform-grid counterparts: Appendix D.5.

Schedule	Supervision	Heun@20	Heun@50	Eul.@50	Eul.@100
CondOT	1st order	17.75	2.42	**3.45**	2.60
CondOT	dual (λ=0.3)	14.41	2.46	**3.45**	2.74
Polynomial	1st order	21.28	2.58	7.64	**2.59**
Polynomial	dual (SoF)	**13.41**	**2.41**	3.58	2.87



(a) $\lambda = 0.0$ (b) $\lambda = 0.1$ (c) $\lambda = 0.3$ (d) $\lambda = 0.5$

Figure 7: **FID (↓, top) and IS (↑, bottom) vs. epoch on CIFAR-10 for $\lambda_{\text{target}} \in \{0, 0.1, 0.3, 0.5\}$.** Solid: 2nd-Euler/Heun/RK4 at 100 NFE; dashed gray: Dopri5 (> 100 NFE); IS bands show mean $\pm$ std. Green shading: best fixed-step surpasses Dopri5; stars mark per-panel optima with Δ to Dopri5.

Table 4: **Inpainting quality metrics.** Full-image metrics (PSNR↑, SSIM_{10^2}↑, LPIPS_{10^3}↓) assess global reconstruction fidelity; mask-only metrics (PSNR↑, SSIM_{10^2}↑, MSE_{10^3}↓) isolate the inpainted region. **Bold:** best per column; underline: second best.

Method	CelebA-HQ (Full)			CelebA-HQ (Mask)			LSUN-Church (Full)			LSUN-Church (Mask)		
	PSNR↑	SSIM_{10^2}↑	LPIPS_{10^3}↓	PSNR↑	SSIM_{10^2}↑	MSE_{10^3}↓	PSNR↑	SSIM_{10^2}↑	LPIPS_{10^3}↓	PSNR↑	SSIM_{10^2}↑	MSE_{10^3}↓
NFE = 20												
2nd-Euler	**22.30**	**88.10**	**64.01**	**21.95**	**87.10**	**27.26**	**18.51**	**81.40**	**148.98**	**18.16**	**79.85**	**65.23**
2nd-Heun	21.87	85.52	68.17	21.52	84.31	30.07	18.05	80.00	155.27	17.71	78.34	72.41
2nd-RK4	19.82	82.63	119.75	19.47	81.18	48.26	16.79	78.71	198.13	16.44	76.93	96.90
DPM++(2V)	21.93	87.52	67.89	21.58	86.47	29.69	18.03	80.93	156.64	17.68	79.34	72.77
DPM++(2A)	22.28	88.07	64.40	21.93	87.07	27.40	18.49	81.38	149.67	18.14	79.82	65.45
NFE = 50												
2nd-Euler	22.11	**87.88**	65.87	21.76	**86.87**	28.46	**18.18**	**81.18**	**152.83**	**17.83**	**79.61**	70.40
2nd-Heun	21.98	87.70	66.95	21.63	86.67	29.32	17.97	81.01	156.12	17.62	79.42	73.77
2nd-RK4	21.12	86.44	78.97	20.77	85.31	35.74	17.43	80.09	173.10	17.08	78.43	83.64
DPM++(2V)	21.98	87.70	67.04	21.64	86.68	29.29	17.98	81.01	156.37	17.63	79.43	73.67
DPM++(2A)	**22.12**	**87.88**	**65.85**	**21.77**	**86.87**	**28.41**	**18.18**	**81.18**	152.86	**17.83**	**79.61**	**70.31**

Design ablations. Two design choices are validated in Figure 7 and Appendix D.1. Sweeping $\lambda_{\text{target}} \in \{0, 0.1, 0.3, 0.5\}$ at 100 NFE, even $\lambda_{\text{target}} = 0$ matches adaptive Dopri5, and supervision lifts all fixed-step solvers above it, peaking at 0.3 (our default). The degree $n = 2$ yields the sharpest samples, as $n = 1$ carries no acceleration signal and $n > 2$ suffers imbalanced supervision (Figure 9, Appendix D.1).

Robustness. Perturbing the acceleration head leaves FID unchanged up to $\varepsilon = 3$, whereas identical noise on the velocity head degrades it to 114.3 (Table 9), since the correction is only $\sim 0.2\%$ of the step magnitude.

Inpainting. To assess generalization beyond unconditional generation, we evaluate image inpainting on CelebA-HQ (Karras et al., 2018) and LSUN-Church (Yu et al., 2015) at 128×128: masks come from dilated MNIST digits, and a repaint constraint enforces the ground-truth interpolation $x_t = (1 - t)\epsilon + tx_0$ on known pixels at each step.

Figure 8 compares five second-order samplers qualitatively; Table 4 reports full-image and mask-only metrics (PSNR, SSIM_{10^2}, LPIPS_{10^3} / MSE_{10^3}; subscripts denote $100\times$/$1000\times$ scaling). All runs use the original CondOT formula, since the schedule-matched variants yield no significant gains at the 20–50 NFE used here (Table 2). DPM-Solver++ (2A) and 2nd-Euler consistently achieve the best or second-best results, confirming the benefit of explicit acceleration for conditional generation.

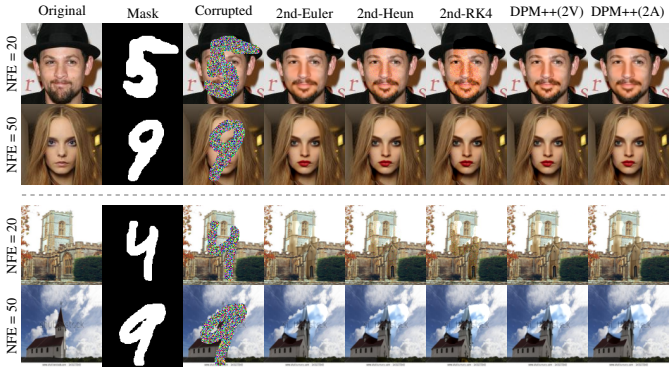


Figure 8: **Inpainting comparison.** Top: CelebA-HQ (128×128). Bottom: LSUN-Church (128×128). Five second-order solvers are compared: masks are derived from dilated MNIST digits, and the repaint constraint enforces the ground-truth path on known pixels.

4 Related Work

Training-free fast samplers. DDIM (Song et al., 2021a), DPM-Solver (Lu et al., 2022a;b), UniPC (Zhao et al., 2023), and EDM (Karras et al., 2022) reduce sampling steps by reconstructing higher-order information via multiple queries per step; our single-NFE Taylor step avoids these, whereas Taylor integrators built on automatic differentiation pay a $10\times$ per-step cost (Appendix D.5). SoF instead amortizes curvature into training-time supervision.

Flow matching variants and few-step distillation. Recent work scales flow matching to large-scale synthesis (Esser et al., 2024; Ma et al., 2024) and discrete spaces (Gat et al., 2024). Consistency Models (Song et al., 2023) and CTM (Kim et al., 2024) map any ODE point directly to data at 1–2 NFE (FID $\approx$ 2.9–3.6; further refinements (Frans et al., 2025; Yin et al., 2024; Geng et al., 2025)), yet all distill a pretrained teacher; SoF instead trains from scratch with an *explicit* acceleration head.

Higher-order solvers and dynamics. Classical multi-step and RK schemes (Butcher, 2016; Dormand & Prince, 1980), Bespoke Solvers (Shaul et al., 2024), and BELM (Wang et al., 2024) treat the velocity network as a black box. Generator Matching (Holderrieth et al., 2025) is the most general first-order abstraction; our $\mathcal{H}_t = \mathcal{G}_t^2 + \partial_t \mathcal{G}_t$ extends it, making acceleration directly learnable (2V/2A, Algorithms 1–2).

Continuous-time generative modeling. Neural ODEs (Chen et al., 2018), free-form continuous flows (Grathwohl et al., 2019), and diffusion models' probability-flow ODEs (Ho et al., 2020; Song et al., 2021b) all model only first-order drift, reconstructing curvature implicitly at sampling time (Lu et al., 2022a). SoF instead makes second-order dynamics a primary target: via phase-space equivalence (Dupont et al., 2019), the acceleration field becomes a learnable output, shifting the cost of curvature from sampling-time queries to a small training-time overhead.

5 Conclusions

We have presented SoF, a flow-matching generator in which a lightweight acceleration head, derived from the squared-generator identity and a phase-space equivalence, upgrades sampling to second-order accuracy at one NFE per step with $< 2\%$ overhead. Our experiments identify and address two inference-time issues, the schedule-formula mismatch ($75.31 \rightarrow 4.53$ at 8 NFE, $16\times$) and the uniform-grid corrector distortion ($14.40 \rightarrow 4.22$ at 50 NFE); with matched schedule and grid, second-order sampling beats same-model first-order solvers at moderate-to-high budgets, reaching FID 2.41 at 100 NFE on CIFAR-10 and 2.56 on CelebA-HQ without distillation, and DPM-Solver++ (2V) surpasses the best first-order solver from 20 NFE onward. The same protocol transfers to conditional generation, where acceleration supervision consistently improves inpainting quality on CelebA-HQ and LSUN-Church. Limitations (resolution capped at 128×128, fixed $n = 2$, modest gains at very low NFE) point to distillation and adaptive schedules as natural extensions.

ACKNOWLEDGEMENTS

Omitted for double-blind review.

AI USE STATEMENT

In this work, we used generative AI tools to polish the text of this paper and reduce grammatical and stylistic errors, and to draft parts of the training and experiment scripts. All AI-edited text was carefully read and verified by the authors, and all AI-assisted scripts were reviewed, tested, and cross-checked against first-order reference implementations. We have not used generative AI tools to develop theoretical models, formulate mathematical claims, design experiments, or interpret results. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

John C. Butcher. *Numerical Methods for Ordinary Differential Equations*. Wiley, 3rd edition, 2016.

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

John R. Dormand and Peter J. Prince. A family of embedded runge-kutta formulae. *Journal of Computational and Applied Mathematics*, 6(1):19–26, 1980.

Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural ODEs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

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

William Feller. The parabolic differential equations and the associated semi-groups of transformations. *Annals of Mathematics*, 55(3):468–519, 1952.

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

Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky T. Q. Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J. Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. *arXiv preprint arXiv:2505.13447*, 2025.

Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. FFJODE: Free-form continuous dynamics for scalable reversible generative models. In *International Conference on Learning Representations (ICLR)*, 2019.

Ernst Hairer and Gerhard Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*. Springer, 2nd edition, 1996.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Peter Holderrieth, Michael Hutchinson, and Yee Whye Teh. Generator matching: Generative modeling with arbitrary markov processes. In *International Conference on Learning Representations (ICLR)*, 2025.

Olav Kallenberg. *Foundations of Modern Probability*. Springer, 3rd edition, 2021.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations (ICLR)*, 2018.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

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 (ICLR)*, 2024.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

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

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 (ICLR)*, 2023.

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 (NeurIPS)*, 2022a.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022b.

Nanye Ma, Mark Goldstein, Michael S. Albergo, Nicholas M. Boffi, Eric Vanden-Eijnden, and Saining Xie. SiT: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision (ECCV)*, 2024.

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

Neta Shaul, Juan Perez, Ricky T. Q. Chen, Ali Thabet, Albert Pumarola, and Yaron Lipman. Bespoke solvers for generative flow models. In *International Conference on Learning Representations (ICLR)*, 2024.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations (ICLR)*, 2021a.

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

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

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

Fangyikang Wang, Hubery Yin, Yuejiang Dong, Huminhao Zhu, Chao Zhang, Hanbin Zhao, Hui Qian, and Chen Li. BELM: Bidirectional explicit linear multi-step sampler for exact inversion in diffusion models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Tianwei Yin, Michael Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Frédéo Durand, and William T. Freeman. Improved distribution matching distillation for fast image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop. In *Computer Vision and Pattern Recognition (CVPR)*, 2015.

Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. UniPC: A unified predictor-corrector framework for fast sampling of diffusion models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

APPENDICES

Notation. For ease of reference, we collect the core symbols used throughout the paper.

Table 5: Core notation used throughout the paper.

Symbol	Meaning	Symbol	Meaning
Probability & Generators			
$p_{t+h\|t}$	Transition kernel of the Markov process	$\tilde{\mathcal{G}}_t$	Expanded 1st-order generator on phase space
$\mathcal{G}_t$	First-order (infinitesimal) generator	$f, \mathcal{T}$	Test function; class of test functions
$\mathcal{H}_t$	Second-order generator: $\mathcal{G}_t^2 + \partial_t \mathcal{G}_t$		
Velocity & Acceleration			
$u_t(x)$	True velocity field	v_θ, a_θ	Network velocity / acceleration predictions
$\alpha_t(x)$	Acceleration field: $(\nabla u_t)u_t + \partial_t u_t$	x_{pred}	Predictor state in 2nd-order Heun
Path & Schedule			
μ_t, σ_t	Schedule functions of the affine path	n	Polynomial degree ($\mu_t = t^n$); default $n = 2$
X_t	Affine path: $\mu_t X_1 + \sigma_t X_0$	X_0, X_1	Source (noise) / target (data) samples
$\dot{X}_t, \ddot{X}_t$	Conditional velocity / acceleration targets		
Training & Sampling			
$\mathcal{L}_{\text{vel}}, \mathcal{L}_{\text{acc}}$	Velocity / acceleration training losses	λ_{target}	Target acceleration weight; default 0.3
$\lambda(e)$	Epoch-dependent weight (cosine warmup)	E_w	Warmup period; default 15% of epochs
h	ODE step size	N	Number of ODE steps (NFE budget)

A ASSUMPTIONS

This section formalises the assumptions underlying our derivations, ensuring rigour throughout.

A.1 ASSUMPTIONS ON STATE SPACE AND MEASURABILITY

Assumption A.1 (Polish state space)**.** The state space $\mathcal{S}$ is assumed to be a Polish space, i.e., a separable completely metrizable topological space. Concretely, we fix a metric $d : \mathcal{S} \times \mathcal{S} \to \mathbb{R}_{\geq 0}$ such that $(\mathcal{S}, d)$ is:

1. Complete: every Cauchy sequence converges in $\mathcal{S}$;
2. Separable: there exists a countable dense subset of $\mathcal{S}$.

This assumption guarantees the existence of regular conditional probabilities, the validity of the Kolmogorov extension theorem for Markov kernels, and the measurability of all limit- and integral-based constructions in the paper. All practically relevant state spaces in machine learning, including Euclidean spaces, countable discrete spaces, compact Riemannian manifolds, and their finite products, satisfy this assumption.

Assumption A.2 (Reference measure and absolute continuity)**.** We assume a fixed σ-finite reference measure ν on $\mathcal{S}$ (e.g., counting measure for discrete spaces, Lebesgue measure for $\mathbb{R}^d$, or Riemannian volume measure for manifolds). All probability measures p considered in this work are assumed to be absolutely continuous with respect to ν, admitting Radon-Nikodym densities $\frac{\mathrm{d}p}{\mathrm{d}\nu} : \mathcal{S} \to \mathbb{R}_{\geq 0}$. This ensures a unified treatment of probability distributions across different state spaces via density representations.

Assumption A.3 (Borel measurability)**.** All functions, kernels, and operators considered in the paper, including transition kernels $p_{t+h|t}(\cdot \mid x)$, vector fields $u_t(x)$, and test functions $f \in \mathcal{T}$, are assumed to be Borel measurable with respect to the appropriate σ-algebras. This ensures that all integrals and expectations are well defined, and that standard measure-theoretic tools (e.g., Fubini's theorem, dominated convergence) can be applied without further justification.

A.2 ASSUMPTIONS ON TEST FUNCTIONS AND GENERATORS

Assumption A.4 (Test function space)**.** We assume a sufficiently rich space of test functions $\mathcal{T} \subseteq C_b(\mathcal{S})$, where $C_b(\mathcal{S})$ denotes the space of bounded continuous real-valued functions on $\mathcal{S}$. The space $\mathcal{T}$ is required to satisfy:

1. Denseness in $L^1(\mathcal{S}, \nu)$ with respect to the L^1-norm;
2. Closure under the action of the generator $\mathcal{G}_t$ for all $t \in [0, T]$;
3. Approximate containment of indicator functions of Borel sets (e.g., via monotone class arguments).

This richness ensures that probability distributions are completely characterized by their expectations on $\mathcal{T}$, and that the weak-form Kolmogorov forward equation uniquely determines the evolution of p_t.

Assumption A.5 (Generator regularity)**.** The time-dependent generator $\mathcal{G}_t : \mathcal{T} \to \mathcal{M}(\mathcal{S})$ where $\mathcal{M}(\mathcal{S})$ denotes measurable functions on $\mathcal{S}$ is assumed to satisfy:

1. Linearity: $\mathcal{G}_t(\alpha f + \beta g) = \alpha \mathcal{G}_t f + \beta \mathcal{G}_t g$ for all $\alpha, \beta \in \mathbb{R}$, $f, g \in \mathcal{T}$;
2. Measurable dependence: The map $(t, x) \mapsto (\mathcal{G}_t f)(x)$ is jointly measurable in t and x for each $f \in \mathcal{T}$;
3. Local boundedness: For each compact $K \subset \mathcal{S}$ and $f \in \mathcal{T}, \sup_{t \in [0,T], x \in K} |(\mathcal{G}_t f)(x)| < \infty$.

These conditions ensure the well-posedness of the Kolmogorov forward equation and the existence of a regular version of the corresponding Markov process.

Assumption A.6 (Kolmogorov forward equation)**.** For any initial distribution p_0 absolutely continuous with respect to ν, we assume the marginal distributions p_t satisfy the Kolmogorov forward equation (KFE) in weak form: for all test functions $f \in \mathcal{T}$,

$$\frac{\mathrm{d}}{\mathrm{d}t} \mathbb{E}_{x \sim p_t}[f(x)] = \mathbb{E}_{x \sim p_t}[(\mathcal{G}_t f)(x)], \tag{13}$$

where the derivative is interpreted in the sense of distributions on $[0, T]$. Equivalently, in density form (for sufficiently regular densities),

$$\frac{\partial}{\partial t} \frac{\mathrm{d}p_t}{\mathrm{d}\nu}(x) = \mathcal{G}_t^* \left(\frac{\mathrm{d}p_t}{\mathrm{d}\nu} \right)(x), \tag{14}$$

where $\mathcal{G}_t^*$ is the formal adjoint of $\mathcal{G}_t$ in $L^2(\mathcal{S}, \nu)$. This assumption establishes the fundamental link between the generator and the evolution of probability distributions.

A.3 Assumptions on Second-Order Regularity

Assumption A.7 (Second-order regularity and interchange conditions)**.** For the derivation of the second-order generator formula and related results, we impose the following regularity conditions:

1. Differentiability of transition action: For each $f \in \mathcal{T}$ and $x \in \mathcal{S}$, the function $h \mapsto \langle p_{t+h|t}, f \rangle(x)$ is twice continuously differentiable in a neighborhood of $h = 0$, with the second derivative continuous in both t and x.
2. Time differentiability of the generator: The generator $\mathcal{G}_t$ is differentiable in t (in the operator sense on $\mathcal{T}$), meaning that for each $f \in \mathcal{T}$, the limit $\partial_t[\mathcal{G}_t f] = \lim_{h \to 0} \frac{\mathcal{G}_{t+h} f - \mathcal{G}_t f}{h}$ exists in the topology of pointwise convergence, and the mapping $(t, x) \mapsto (\partial_t \mathcal{G}_t f)(x)$ is jointly measurable.
3. Interchange of limits and expectations: For each $f \in \mathcal{T}$ and $x \in \mathcal{S}$, the following interchange holds:
$$\lim_{h \to 0} \mathbb{E}\left[\frac{(\mathcal{G}_{t+h} - \mathcal{G}_t) f}{h}(X_{t+h}) \mid X_t = x \right] = \mathbb{E}\left[\lim_{h \to 0} \frac{(\mathcal{G}_{t+h} - \mathcal{G}_t) f}{h}(X_{t+h}) \mid X_t = x \right]. \tag{15}$$
This is justified by suitable dominated convergence conditions, which are assumed to hold.
4. Feller-like continuity: For each fixed t, h, and $f \in \mathcal{T}$, the transition action $\langle p_{t+h|t}, f \rangle(x)$ is continuous in x, and the convergence in the generator definition is uniform on compact subsets of $\mathcal{S}$ when required.

These conditions ensure the validity of the second-order generator expansion and the interchange of operations in proofs involving $\mathcal{H}_t$.

A.4 Assumptions on Flows and Vector Fields

Assumption A.8 (Regularity of vector fields for flows)**.** For deterministic flows on $\mathcal{S} = \mathbb{R}^d$ generated by a time-dependent vector field $u_t(x)$, we impose the following conditions:

1. Smoothness: The vector field $u : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ is continuously differentiable in both arguments, i.e., $u \in C^1([0,1] \times \mathbb{R}^d; \mathbb{R}^d)$.
2. Lipschitz continuity: u is globally Lipschitz continuous in x uniformly in t, meaning there exists a constant $L > 0$ such that for all $t \in [0,1]$ and $x, y \in \mathbb{R}^d$,
$$\|u_t(x) - u_t(y)\| \le L\|x - y\|. \tag{16}$$
3. Growth condition: There exists a constant $M > 0$ such that for all $t \in [0,1]$ and $x \in \mathbb{R}^d$,
$$\|u_t(x)\| \le M(1 + \|x\|). \tag{17}$$
4. Measurability: The vector field $u_t(x)$ is Borel measurable in both arguments, consistent with Assumption A.3.

These conditions ensure the global existence and uniqueness of solutions to the ordinary differential equation defining the flow map $\phi_{t,s}$, the differentiability of the flow with respect to initial conditions, and the well-posedness of the associated deterministic Markov process.

B Background

This section provides a self-contained treatment of continuous-time Markov processes and their infinitesimal generators, the theoretical foundation on which the second-order generator $\mathcal{H}_t$ is built. We work over general Polish state spaces $\mathcal{S}$, characterize probability distributions via test functions $\mathcal{T}$, and define the generator $\mathcal{G}_t$ as the infinitesimal evolution operator. The Kolmogorov forward equation (KFE) then links $\mathcal{G}_t$ to the macroscopic evolution of the marginal distribution p_t, with a focus on deterministic flows.

B.1 General State Spaces and Probability Measures

State space structure. Let $\mathcal{S}$ be an arbitrary state space, which may be a Euclidean space $\mathcal{S} = \mathbb{R}^d$ (for continuous data such as images or vectors), a discrete space (for categorical data such as language tokens), a Riemannian manifold (for geometric data), or a product thereof (for multimodal generation). We equip $\mathcal{S}$ with a metric $d : \mathcal{S} \times \mathcal{S} \to \mathbb{R}_{\ge 0}$, $(x, y) \mapsto d(x, y)$. Common choices include the discrete metric $d(x, y) = \mathbb{I}_{\{x \ne y\}}$ for discrete $\mathcal{S}$, and the Euclidean norm $d(x, y) = \|x - y\|$ for $\mathcal{S} = \mathbb{R}^d$. Throughout, we assume $(\mathcal{S}, d)$ is a *Polish metric space*; see Assumption A.1 for a precise statement.

Probability measures and densities. Probability distributions over $\mathcal{S}$ are described by Borel probability measures. As per Assumption A.2, we fix a σ-finite *reference measure* ν on $\mathcal{S}$ (e.g., counting measure for discrete spaces, Lebesgue measure for $\mathbb{R}^d$, or Riemannian volume measure for manifolds). All probability measures p considered in this work are assumed to be absolutely continuous with respect to ν, admitting a Radon-Nikodym density $\frac{\mathrm{d}p}{\mathrm{d}\nu} : \mathcal{S} \to \mathbb{R}_{\ge 0}$, so that for every measurable set $A \subseteq \mathcal{S}$,
$$p(A) = \int_A \frac{\mathrm{d}p}{\mathrm{d}\nu}(x)\, \nu(\mathrm{d}x). \tag{18}$$
In the discrete case, ν is the counting measure and $\frac{\mathrm{d}p}{\mathrm{d}\nu}$ is the probability mass function; for $\mathcal{S} = \mathbb{R}^d$ with Lebesgue measure, it is the usual probability density function. This unified density representation allows us to treat integrals over $\mathcal{S}$ consistently, writing expectations as
$$\mathbb{E}_{x\sim p}[f(x)] = \int_{\mathcal{S}} f(x) \frac{\mathrm{d}p}{\mathrm{d}\nu}(x)\, \nu(\mathrm{d}x), \tag{19}$$

where $f : \mathcal{S} \to \mathbb{R}$ is any measurable test function (measurability is ensured by Assumption A.3).

B.2 Continuous-Time Markov Processes as Generative Models

Definition and Markov property. Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space, and let $X_t : \Omega \to \mathcal{S}$ for $t \in [0, 1]$ be a family of $\mathcal{S}$-valued random variables. Here, $\mathcal{S}$ satisfies the Polish space assumption (Assumption A.1), which ensures the existence of regular conditional probabilities and the measurability of the process. The process $(X_t)_{0 \le t \le 1}$ is called a continuous-time Markov process (CTMP) if for every finite sequence $0 \le t_1 < \cdots < t_{n+1} \le 1$ and every measurable set $A \subseteq \mathcal{S}$,

$$\mathbb{P}[X_{t_{n+1}} \in A \mid X_{t_1}, \ldots, X_{t_n}] = \mathbb{P}[X_{t_{n+1}} \in A \mid X_{t_n}]. \tag{20}$$

The Markov property states that the future evolution depends exclusively on the current state, rendering the process memoryless.

Transition kernels. A CTMP is equivalently characterized by its transition kernels $p_{t+h|t}(\cdot \mid x)$, defined for $0 \le t < t + h \le 1$ and $x \in \mathcal{S}$ as

$$p_{t+h|t}(A \mid x) = \mathbb{P}[X_{t+h} \in A \mid X_t = x], \qquad A \subseteq \mathcal{S} \text{ measurable.} \tag{21}$$

The Borel measurability of these kernels in x is ensured by Assumption A.3. For each x, $p_{t+h|t}(\cdot \mid x)$ is a probability measure on $\mathcal{S}$. By the Markov property, the finite-dimensional distributions of the process are uniquely determined by the initial distribution p_0 and the family of transition kernels $\{p_{t+h|t}\}$. Conversely, any consistent family of transition kernels combined with an initial distribution defines a CTMP, establishing a bijective correspondence.

B.3 Test Functions and Generator Actions

Test functions as distribution probes. Since probability measures do not naturally belong to vector spaces, derivatives of transition kernels cannot be defined directly. We circumvent this by introducing a family of test functions $\mathcal{T}$, assumed to satisfy Assumption A.4. This space consists of bounded measurable functions $f : \mathcal{S} \to \mathbb{R}$ that fully characterize probability measures: for any two probability measures μ_1, μ_2 on $\mathcal{S}$,

$$\mu_1 = \mu_2 \iff \mathbb{E}_{X \sim \mu_1}[f(X)] = \mathbb{E}_{X \sim \mu_2}[f(X)] \quad \forall f \in \mathcal{T}. \tag{22}$$

The space $\mathcal{T}$ is chosen to be as regular as possible while satisfying equation 22. Common choices include $C_c^\infty(\mathbb{R}^d)$ (smooth compactly supported functions) for $\mathcal{S} = \mathbb{R}^d$, and $\mathbb{R}^{\mathcal{S}}$ (all functions, i.e., vectors) for discrete $\mathcal{S}$.

Actions of measures and kernels. For a probability measure p_t and a test function $f \in \mathcal{T}$, the marginal action is defined as

$$\langle p_t, f \rangle \coloneqq \int_{\mathcal{S}} f(x) p_t(\mathrm{d}x) = \mathbb{E}_{X \sim p_t}[f(X)]. \tag{23}$$

This integral is well-defined due to the Borel measurability assumption (Assumption A.3). For a transition kernel $p_{t+h|t}(\cdot \mid x)$, the transition action yields a function of x:

$$\langle p_{t+h|t}, f \rangle(x) \coloneqq \int_{\mathcal{S}} f(y) p_{t+h|t}(\mathrm{d}y \mid x) = \mathbb{E}[f(X_{t+h}) \mid X_t = x]. \tag{24}$$

Again, measurability of the kernel and test function is ensured by Assumption A.3. The marginal action maps a test function to a scalar, while the transition action maps it to another real-valued function. The tower property of conditional expectation implies the consistency relation

$$\langle p_t, \langle p_{t+h|t}, f \rangle \rangle = \langle p_{t+h}, f \rangle. \tag{25}$$

When a density $p_t(x) = \frac{\mathrm{d}p_t}{\mathrm{d}\nu}(x)$ exists with respect to the reference measure ν (per Assumption A.2), the marginal action takes the familiar integral form

$$\langle p_t, f \rangle = \int_{\mathcal{S}} f(x) p_t(x) \, \nu(\mathrm{d}x). \tag{26}$$

Generator as infinitesimal evolution. Using test functions, the generator $\mathcal{G}_t$ can be rigorously defined as the operator satisfying Assumption A.5. Specifically,

$$[\mathcal{G}_t f](x) = \lim_{h \to 0} \frac{\langle p_{t+h|t}, f \rangle(x) - f(x)}{h}, \tag{27}$$

provided the limit exists uniformly for f in a suitable subspace of $\mathcal{T}$. The generator encodes the local first-order behavior of the process, analogous to the velocity field of a deterministic flow.

CTMP generative model (informal). A CTMP can be interpreted as a generative model whose infinitesimal evolution is governed by its generator. Heuristically, the transition kernel admits the first-order expansion

$$p_{t+h|t}(\cdot \mid x) = \delta_x + h\mathcal{G}_t(x) + o(h), \qquad X_0 \sim p_0, \tag{28}$$

where δ_x denotes the Dirac measure at x and the remainder $o(h)$ vanishes faster than h as $h \to 0$. This expansion suggests that learning the generator $\mathcal{G}_t$ suffices to characterize the process up to first order. Deterministic flows, the primary focus of this work, correspond to the case where $\mathcal{G}_t$ acts as a first-order differential operator driven by a velocity field $u_t(x)$.

Kolmogorov forward equation (KFE). The evolution of the marginal distribution p_t is governed by the Kolmogorov forward equation (KFE), formalized in Assumption A.6. In terms of the generator, the KFE reads

$$\frac{\mathrm{d}}{\mathrm{d}t}\langle p_t, f \rangle = \langle p_t, \mathcal{G}_t f \rangle \qquad \forall f \in \mathcal{T}. \tag{29}$$

Equivalently, if p_t admits a density $p_t(x)$ with respect to the reference measure ν, then

$$\partial_t p_t(x) = \mathcal{G}_t^* p_t(x), \tag{30}$$

where $\mathcal{G}_t^*$ is the adjoint of $\mathcal{G}_t$ with respect to the $L^2(\nu)$ inner product. The KFE provides the fundamental link between the infinitesimal generator $\mathcal{G}_t$ and the macroscopic evolution of the probability distribution. In generative modeling, solving the KFE forward in time (with initial condition p_0) yields the marginal distributions p_t that define the corruption or noising process, while solving it backward in time (with terminal condition p_1) yields the generative sampling process.

C Computation and Derivation

This appendix provides the complete step-by-step derivations supporting the results in Section 2.

C.1 Proof of Second-Order Generator Formula

The second-order generator $\mathcal{H}_t$ is formally defined as the second derivative of the transition action with respect to the time increment:

$$[\mathcal{H}_t f](x) \coloneqq \frac{\partial^2}{\partial h^2}\bigg|_{h=0} \langle p_{t+h|t}, f \rangle(x), \tag{31}$$

where $\langle p_{t+h|t}, f \rangle(x) = \mathbb{E}[f(X_{t+h}) \mid X_t = x]$ denotes the conditional expectation under the Markov transition kernel. Under the regularity conditions formalized in Assumption A.7, we derive its explicit representation in terms of the first-order generator $\mathcal{G}_t$.

We proceed step by step, leveraging the Kolmogorov forward equation (KFE) and properties of conditional expectations:

$$[\mathcal{H}_t f](x) = \frac{\partial^2}{\partial h^2}\bigg|_{h=0} \langle p_{t+h|t}, f \rangle(x) \qquad \text{(Def. of } \mathcal{H}_t\text{)} \tag{32}$$

$$= \frac{\partial}{\partial h}\bigg|_{h=0} \frac{\partial}{\partial \epsilon}\bigg|_{\epsilon=0} \langle p_{t+h+\epsilon|t}, f \rangle(x) \qquad \text{(Mixed partials equal)} \tag{33}$$

$$= \frac{\partial}{\partial h}\bigg|_{h=0} \frac{\partial}{\partial \epsilon}\bigg|_{\epsilon=0} \mathbb{E}[f(X_{t+h+\epsilon}) \mid X_t = x] \qquad \text{(Kernel = cond. law)} \tag{34}$$

$$= \frac{\partial}{\partial h}\Big|_{h=0} \frac{\partial}{\partial \epsilon}\Big|_{\epsilon=0} \mathbb{E}\big[\mathbb{E}[f(X_{t+h+\epsilon}) \mid X_{t+h}] \mid X_t = x\big] \quad \text{(Tower property)} \tag{35}$$

$$= \frac{\partial}{\partial h}\Big|_{h=0} \mathbb{E}\big[(\mathcal{G}_{t+h} f)(X_{t+h}) \mid X_t = x\big] \quad \text{(Def. of } \mathcal{G}_s\text{)} \tag{36}$$

$$= \frac{\partial}{\partial h}\Big|_{h=0} \langle p_{t+h|t}, \mathcal{G}_{t+h} f\rangle(x) \quad \text{(Duality)} \tag{37}$$

$$= \frac{\partial}{\partial h}\Big|_{h=0} \langle p_{t+h|t}, \mathcal{G}_t f\rangle(x) \quad \text{(Add/subtract } \mathcal{G}_t f\text{)}$$

$$+ \frac{\partial}{\partial h}\Big|_{h=0} \langle p_{t+h|t}, (\mathcal{G}_{t+h} - \mathcal{G}_t) f\rangle(x) \tag{38}$$

$$= [\mathcal{G}_t^2 f](x) + \lim_{h\to 0} \frac{\langle p_{t+h|t}, (\mathcal{G}_{t+h} - \mathcal{G}_t) f\rangle(x)}{h} \quad \text{(KFE; derivative def.)} \tag{39}$$

$$= [\mathcal{G}_t^2 f](x) + \lim_{h\to 0} \mathbb{E}\left[\frac{(\mathcal{G}_{t+h} - \mathcal{G}_t) f}{h}(X_{t+h}) \mid X_t = x\right] \quad \text{(Rewriting)} \tag{40}$$

$$= [\mathcal{G}_t^2 f](x) + \mathbb{E}\big[(\partial_t \mathcal{G}_t f)(X_t) \mid X_t = x\big] \quad \text{(A.7: limit/expect. swap)} \tag{41}$$

$$= [\mathcal{G}_t^2 f](x) + [\partial_t \mathcal{G}_t f](x). \quad \text{(Cond. on } X_t = x\text{)} \tag{42}$$

The derivation relies on several technical conditions formalized in Assumption A.7. These ensure the existence of second-order derivatives, the validity of interchanging limits and expectations, and the Feller-like continuity properties of the process. This representation shows that $\mathcal{H}_t$ captures both the compounded first-order evolution $\mathcal{G}_t^2$ and the explicit time variation of the generator itself, providing a complete second-order description of the Markov dynamics.

C.2 Second-Order Generator for Deterministic Flows

Definition. Let $\mathcal{S} = \mathbb{R}^d$ and let $u \in C^1([0,1]\times\mathbb{R}^d;\mathbb{R}^d)$ be a Lipschitz-continuous time-dependent vector field, $(t,x) \mapsto u_t(x)$, ensuring global existence and uniqueness of solutions. This regularity condition is formalized in Assumption A.8. The associated (measure-preserving) flow map $\phi_{t,s} : \mathbb{R}^d \to \mathbb{R}^d$ is defined as the unique classical solution to the non-autonomous linear ordinary differential equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\phi_{t,s}(x) = u_t\big(\phi_{t,s}(x)\big), \qquad \phi_{s,s}(x) = x, \tag{43}$$

for all $0 \le s \le t \le 1$ and $x \in \mathbb{R}^d$. The state space $\mathbb{R}^d$ satisfies the Polish space assumption (Assumption A.1). The pushforward $\phi_{t,s\#}\mu$ of any initial measure μ under this deterministic diffeomorphism defines the transition semigroup of a continuous-time deterministic Markov process, with transition kernel

$$p_{t|s}(\cdot \mid x) = \delta_{\phi_{t,s}(x)}(\cdot), \tag{44}$$

a Dirac measure supported at the trajectory endpoint $\phi_{t,s}(x)$. The first-order Euler discretization of the integral curve yields the infinitesimal propagation

$$X_{t+h} = X_t + h u_t(X_t) + o(h) \quad \text{as } h \to 0^+, \tag{45}$$

corresponding to a pure deterministic drift without stochastic or discontinuous contributions.

Second-Order Generator. Let $\mathcal{T} = C_c^\infty(\mathbb{R}^d)$ be the space of infinitely differentiable functions with compact support (test functions in $\mathbb{R}^d$), which satisfies Assumption A.4. Then the second-order generator can be derived from the infinitesimal evolution of expectations under the flow as follows. In addition to Assumption A.8, we assume for this calculation that $u_t(x)$ is twice continuously differentiable in x and once in t to ensure the existence of all derivatives involved.

$$[\mathcal{H}_t f] = [\mathcal{G}_t^2 f] + \partial_t[\mathcal{G}_t f] \quad \text{(Def. of } \mathcal{H}_t\text{, A.7)} \tag{46}$$

$$= \mathcal{G}_t[\mathcal{G}_t f] + \partial_t(\nabla f^\top u_t) \quad \text{(Use } \mathcal{G}_t g = \nabla g^\top u_t\text{, A.8)} \tag{47}$$

$$= u_t^\top \nabla(\nabla f^\top u_t) + \nabla f^\top \partial_t u_t \quad \text{(Apply } \mathcal{G}_t \text{ to } \nabla f^\top u_t\text{)} \tag{48}$$

$$= [\nabla(\nabla f^\top u_t)]^\top u_t + \nabla f^\top \partial_t u_t \qquad \text{(Scalar = transpose)} \tag{49}$$

$$= [\nabla(\nabla f)^\top u_t + \nabla(u_t^\top)\nabla f]^\top u_t + \nabla f^\top \partial_t u_t \qquad \text{(Product rule for gradients)} \tag{50}$$

$$= [(\nabla^2 f)u_t + (\nabla u_t)^\top \nabla f]^\top u_t + \nabla f^\top \partial_t u_t \qquad \text{(Identify Hessian)} \tag{51}$$

$$= u_t^\top (\nabla^2 f)u_t + \nabla f^\top (\nabla u_t)u_t + \nabla f^\top \partial_t u_t \qquad \text{(Expand inner product)} \tag{52}$$

$$= u_t^\top \nabla^2 f u_t + \nabla f^\top [(\nabla u_t)u_t + \partial_t u_t] \qquad \text{(Group terms)} \tag{53}$$

$$= u_t^\top \nabla^2 f u_t + \nabla f^\top \alpha_t, \qquad \alpha_t := (\nabla u_t)u_t + \partial_t u_t. \qquad \text{(Def. acceleration field } \alpha_t) \tag{54}$$

C.3 Phase-Space Equivalent Flow

Motivation. The second-order generator $\mathcal{H}_t$ derived in Theorem 2.1 involves the composition $\mathcal{G}_t^2$ and the time derivative $\partial_t \mathcal{G}_t$, which operate as second-order differential operators on test functions. Direct parameterization of $\mathcal{H}_t$ would require neural networks to output Hessian-like quantities, which complicates training and increases computational cost.

To circumvent this challenge while preserving the expressive power of second-order dynamics, we employ a *lifting* strategy: we augment the state space to include velocity as an independent variable, thereby transforming the second-order dynamics in $\mathbb{R}^d$ into a first-order flow in the expanded phase space $\mathbb{R}^{2d}$. This approach maintains the algorithmic simplicity of first-order Flow Matching while capturing the curvature effects encoded in $\mathcal{H}_t$.

Phase-Space Lifting. Let the expanded state space be $\mathcal{Y} = \mathbb{R}^d \times \mathbb{R}^d$ with elements $y = (x, v)$, where $x \in \mathbb{R}^d$ is position and $v \in \mathbb{R}^d$ is velocity. Define a time-dependent vector field $\tilde{u}_t : \mathcal{Y} \to \mathbb{R}^{2d}$ on this space as:

$$\tilde{u}_t(x, v) = \begin{pmatrix} v \\ \alpha_t(x) \end{pmatrix}, \tag{55}$$

where $\alpha_t(x) = (\nabla u_t(x))u_t(x) + \partial_t u_t(x)$ is the acceleration field defined in equation 54.

The corresponding first-order generator $\tilde{\mathcal{G}}_t$ acting on test functions $g \in C_c^\infty(\mathbb{R}^{2d})$ takes the standard flow generator form:

$$\tilde{\mathcal{G}}_t g(x, v) = \nabla_x g(x, v)^\top v + \nabla_v g(x, v)^\top \alpha_t(x). \tag{56}$$

Proceed with direct computation of $\tilde{\mathcal{G}}_t^2 \bar{f}$. First, apply $\tilde{\mathcal{G}}_t$ to $\bar{f}$:

$$\tilde{\mathcal{G}}_t \bar{f}(x, v) = \nabla_x \bar{f}(x, v)^\top v + \nabla_v \bar{f}(x, v)^\top \alpha_t(x) = \nabla f(x)^\top v, \tag{57}$$

using $\nabla_x \bar{f} = \nabla f$ and $\nabla_v \bar{f} = 0$ (since $\bar{f}$ is independent of v). Let $h(x, v) := \tilde{\mathcal{G}}_t \bar{f}(x, v) = \nabla f(x)^\top v$. Compute the gradients of h:

$$\nabla_x h(x, v) = \nabla_x(\nabla f(x)^\top v) = \nabla^2 f(x) v,$$
$$\nabla_v h(x, v) = \nabla_v(\nabla f(x)^\top v) = \nabla f(x). \tag{58}$$

Apply $\tilde{\mathcal{G}}_t$ again:

$$\tilde{\mathcal{G}}_t^2 \bar{f}(x, v) = \tilde{\mathcal{G}}_t h(x, v) \qquad \text{(Define } h = \tilde{\mathcal{G}}_t \bar{f}) \tag{59}$$

$$= \nabla_x h(x, v)^\top v + \nabla_v h(x, v)^\top \alpha_t(x) \qquad \text{(Apply def. of } \tilde{\mathcal{G}}_t) \tag{60}$$

$$= (\nabla^2 f(x) v)^\top v + \nabla f(x)^\top \alpha_t(x) \qquad \text{(Substitute } \nabla_x h = \nabla^2 f v, \nabla_v h = \nabla f) \tag{61}$$

$$= v^\top \nabla^2 f(x) v + \nabla f(x)^\top \alpha_t(x) \qquad \text{(Transpose scalar: } (AB)^\top = B^\top A^\top) \tag{62}$$

Finally, restrict to the constraint manifold $v = u_t(x)$:

$$\tilde{\mathcal{G}}_t^2 \bar{f}(x, v)\Big|_{v=u_t(x)} = u_t(x)^\top \nabla^2 f(x) u_t(x) + \nabla f(x)^\top \alpha_t(x) = \mathcal{H}_t f(x). \tag{63}$$

D Method Details

This section details the technical components of our framework: the polynomial convex scheduler, the velocity-acceleration conversion, the acceleration-enhanced sampling algorithms, the implicit/explicit curvature equivalence, and the dual-head UNet architecture.

D.1 Scheduler and Training Details

Polynomial scheduler analysis. We analyze the polynomial convex scheduler $\mu_t = t^n$, $\sigma_t = 1 - t^n$ for various $n \geq 2$ and justify the choice $n = 2$ as the best trade-off among signal uniformity, numerical stability, and path smoothness within the polynomial family.

Table 6: FID (↓) vs. polynomial degree n on CIFAR-10 (10k samples).

n	100	200	300	400	500
1	9.80	7.50	6.93	6.69	6.48
1.5	8.71	7.16	6.76	6.60	6.50
2	8.63	7.14	6.76	6.54	6.44
3	8.70	7.30	6.89	6.90	6.76
4	9.24	7.81	7.28	7.26	7.10



Figure 9: **Generated samples for different polynomial degrees n.** CIFAR-10 at 500 epochs. $n = 2$ (left of dashed line) produces the sharpest and most diverse samples; other values either lack acceleration signal ($n = 1$) or suffer from imbalanced supervision ($n \neq 2$).

Justification of $n = 2$. The second derivative of the signal coefficient is $\ddot{\mu}_t = n(n-1)t^{n-2}$. We examine three regimes:

- $n = 2$: $\ddot{\mu}_t = 2$ (constant), giving uniform acceleration supervision over $t \in [0, 1]$ with no endpoint singularities and stable gradient propagation.
- $n > 2$: $\ddot{\mu}_t = n(n-1)t^{n-2} \to 0$ as $t \to 0$, so the acceleration signal vanishes in the early-time region and capacity is wasted on negligible curvature.
- $n < 2$: for $n = 1$, $\ddot{\mu}_t = 0$ recovers the CondOT degeneracy (Proposition 2.1); for non-integer $n \in (1, 2)$, $\ddot{\mu}_t$ diverges as $t \to 0$, yielding infinite targets near the origin. Empirically (Table 6), both extremes degrade performance.

Thus $n = 2$ is the unique integer yielding constant, non-zero second derivatives without singularities: uniformly strong supervision, a C^∞ path, and excellent numerical conditioning. Table 6 confirms that $n = 2$ achieves the lowest FID across epochs, and Figure 9 gives a qualitative comparison.

Path dynamics intuition. With $n = 2$, the path exhibits a "slow start, fast finish" dynamic:

- Near $t \approx 0$: $\mu_t \approx 0$, so $X_t \approx X_0$ (noise-dominated, nearly stationary); the small initial velocity ($\dot{\mu}_0 = 0$) prevents abrupt transitions.
- Near $t \approx 1$: $\dot{\mu}_t = 2t \approx 2$, so the path converges rapidly to X_1. Velocity increases linearly from 0 to 2, as under constant force, giving a smooth yet efficient source-to-target transition.

Velocity-acceleration conversion. Given the affine path $X_t = \mu_t X_1 + \sigma_t X_0$ and a velocity prediction $v_\theta(X_t, t) \approx \dot{X}_t = \dot{\mu}_t X_1 + \dot{\sigma}_t X_0$, we recover X_1 and X_0 from (X_t, v_θ), which is useful for generating acceleration targets from a pre-trained velocity model without re-training.

From the linear system

$$\begin{cases} X_t = \mu_t X_1 + \sigma_t X_0, \\ v_\theta = \dot{\mu}_t X_1 + \dot{\sigma}_t X_0, \end{cases} \tag{64}$$

Eliminating X_0 (multiply the first equation by $\dot{\sigma}_t$, the second by σ_t, and subtract) gives

$$\dot{\sigma}_t X_t - \sigma_t v_\theta = (\dot{\sigma}_t \mu_t - \sigma_t \dot{\mu}_t) X_1. \tag{65}$$

Hence

$$X_1 = \frac{\dot{\sigma}_t X_t - \sigma_t v_\theta}{\dot{\sigma}_t \mu_t - \sigma_t \dot{\mu}_t}, \qquad X_0 = \frac{X_t - \mu_t X_1}{\sigma_t}. \tag{66}$$

For the polynomial scheduler with $n = 2$ ($\mu_t = t^2$, $\sigma_t = 1 - t^2$, $\dot{\mu}_t = 2t$, $\dot{\sigma}_t = -2t$),

$$\dot{\sigma}_t \mu_t - \sigma_t \dot{\mu}_t = (-2t)(t^2) - (1 - t^2)(2t) = -2t^3 - 2t + 2t^3 = -2t, \tag{67}$$

which is non-zero for $t \in (0, 1]$ (at $t = 0$ the path is degenerate, but training samples are drawn from $t > 0$), so the conversion is well-defined.

With X_0 and X_1 recovered, the acceleration target is

$$a_t = \ddot{\sigma}_t X_0 + \ddot{\mu}_t X_1. \tag{68}$$

For $n = 2$, $\ddot{\mu}_t = 2$ and $\ddot{\sigma}_t = -2$, so $a_t = 2(X_1 - X_0)$: the constant acceleration of the quadratic path. The conversion is exact when v_θ matches the true conditional velocity and otherwise provides a consistent supervisory signal.

D.2 Sampling Algorithms and Error Analysis

Sampling algorithm details. We present the complete formulations of the acceleration-enhanced ODE solvers: the predicted acceleration $a_\theta(x, t)$ corrects the intermediate positions of multi-stage methods, improving subsequent velocity evaluations without extra model calls.

Second-order RK4. The four-stage Runge-Kutta algorithm with acceleration-enhanced intermediate positions, given step size h and state (x, t):

$$\begin{aligned} (v_1, a_1) &= \text{model}(x, t), \\ x_{\text{mid1}} &= x + \frac{h}{2} v_1 + \frac{1}{2} \left(\frac{h}{2}\right)^2 a_1, \\ (v_2, a_2) &= \text{model}(x_{\text{mid1}}, t + h/2), \\ x_{\text{mid2}} &= x + \frac{h}{2} v_2 + \frac{1}{2} \left(\frac{h}{2}\right)^2 a_2, \\ (v_3, a_3) &= \text{model}(x_{\text{mid2}}, t + h/2), \\ x_{\text{end}} &= x + h v_3 + \frac{1}{2} h^2 a_3, \\ (v_4, \cdot) &= \text{model}(x_{\text{end}}, t + h). \end{aligned} \tag{69}$$

The final update combines only velocities, as in classical RK4:

$$x_{t+h} = x + \frac{h}{6}(v_1 + 2v_2 + 2v_3 + v_4). \tag{70}$$

The acceleration fields improve the intermediate positions $x_{\text{mid1}}, x_{\text{mid2}}, x_{\text{end}}$ via second-order Taylor corrections, so v_2, v_3, v_4 are queried at more accurate points, yielding higher accuracy than standard RK4 at no additional NFE (error analysis: Appendix D.2).

DOPRI5 (velocity-only). The Dormand-Prince 5(4) method is an adaptive step-size Runge-Kutta scheme with an embedded order-5/4 error estimator; with Butcher coefficients a_{ij}, c_i, b_i (fifth order) and $\hat{b}_i$ (fourth order), each stage evaluates the velocity field alone:

$$X_i = x + h\sum_{j=1}^{i-1} a_{ij}k_j, \qquad k_i = v_\theta(X_i, t + c_i h), \qquad i = 1, \ldots, 7, \tag{71}$$

and the step-size controller uses the embedded fourth-order error estimate:

$$\text{err} = \|x^{(5)} - x^{(4)}\|, \qquad h_{\text{new}} = h \cdot \text{safety} \cdot \left(\frac{\text{tol}}{\text{err}}\right)^{1/5}, \tag{72}$$

with safety ≈ 0.9. We deliberately do not inject Taylor corrections $\frac{1}{2}(c_i h)^2 a_{\text{prev}}$ into the intermediate stages X_i: the estimator $\|x^{(5)} - x^{(4)}\|$ assumes both flows arise from the same Butcher tableau applied to $\dot{x} = v_\theta$, and $\mathcal{O}(h^2)$ stage perturbations inflate err and trigger systematic step rejection; in practice, acceleration-augmented DOPRI5 diverges or produces pure-noise samples. The acceleration signal is therefore reserved for the fixed-stage schemes (Euler, Heun, RK4), where it improves intermediate positions without interacting with an error controller.

Computational budgets. Under a fixed budget of 100 NFE, the effective step counts are:

- Second-order Euler: 100 steps (1 NFE per step);
- Second-order Heun: 50 steps (2 NFE per step);
- Second-order RK4: 25 steps (4 NFE per step);
- Velocity-only DOPRI5: adaptive, up to 100 useful steps depending on tolerance (rejected steps incur extra NFE; with tol = 10^{-3}, typically 80–90 accepted steps).

Empirically, the higher per-step accuracy of higher-order methods more than compensates for their reduced step count at fixed NFE.

Why acceleration improves intermediate point accuracy in RK4. We show that acceleration-enhanced intermediate positions improve per-stage accuracy from $\mathcal{O}(h^2)$ to $\mathcal{O}(h^3)$, yielding more faithful velocity evaluations and tighter global error bounds for RK4.

Setup. Consider the true ODE $\frac{\mathrm{d}x}{\mathrm{d}t} = v(x, t)$ with exact solution $x(t)$. The acceleration field is defined as the material derivative of the velocity:

$$a(x, t) = \frac{\mathrm{d}v}{\mathrm{d}t} = \partial_t v + (\nabla v)v. \tag{73}$$

For a time step δ, the true trajectory admits the Taylor expansion

$$x(t + \delta) = x(t) + \delta v(x(t), t) + \frac{\delta^2}{2}a(x(t), t) + \mathcal{O}(\delta^3). \tag{74}$$

Standard RK4: first-order midpoint prediction. In the standard (velocity-only) RK4 scheme, the first midpoint is estimated using a forward Euler half-step:

$$x_{\text{mid1}} = x + \frac{h}{2}v_1, \qquad v_1 = v(x, t). \tag{75}$$

Comparing with the exact expansion equation 74 evaluated at $\delta = h/2$:

$$x(t + h/2) = x + \frac{h}{2}v_1 + \frac{1}{2}\left(\frac{h}{2}\right)^2 a_1 + \mathcal{O}(h^3), \qquad a_1 = a(x, t), \tag{76}$$

the position error of the standard prediction is

$$e_{\text{mid1}}^{(\text{std})} = x_{\text{mid1}}^{(\text{std})} - x(t + h/2) = -\frac{h^2}{8}a_1 + \mathcal{O}(h^3) = \mathcal{O}(h^2). \tag{77}$$

Acceleration-enhanced prediction: second-order midpoint. Using the learned acceleration field, the midpoint is predicted by the second-order Taylor expansion:

$$x_{\text{mid1}}^{(\text{2nd})} = x + \frac{h}{2}v_1 + \frac{1}{2}\left(\frac{h}{2}\right)^2 a_1. \tag{78}$$

Subtracting the exact expansion gives

$$x_{\text{mid1}}^{(\text{2nd})} - x(t + h/2) = \mathcal{O}(h^3), \tag{79}$$

a full order of improvement over the standard scheme.

Error propagation to velocity evaluations. The midpoint velocity is $v_2 = v(x_{\text{mid1}}, t + h/2)$. With true midpoint $x^* = x(t + h/2)$ and true velocity $v^* = v(x^*, t + h/2)$, a first-order expansion gives

$$v_2 = v^* + (\nabla v)(x_{\text{mid1}} - x^*) + \mathcal{O}(\|x_{\text{mid1}} - x^*\|^2). \tag{80}$$

so the velocity evaluation error inherits the order of the position error:

Method	Position error $\|e_{\text{mid}}\|$	Velocity error $\|v_2 - v^*\|$
Standard RK4	$\mathcal{O}(h^2)$	$\mathcal{O}(h^2)$
Acceleration-enhanced RK4	$\mathcal{O}(h^3)$	$\mathcal{O}(h^3)$

Impact on the final RK4 update. The classical RK4 combination reads

$$x_{t+h} = x + \frac{h}{6}(v_1 + 2v_2 + 2v_3 + v_4). \tag{81}$$

Let e_{mid} denote the intermediate-position error; standard error analysis gives:

- *Standard RK4*: $e_{\text{mid}} = \mathcal{O}(h^2) \implies$ local truncation error $\mathcal{O}(h^5)$, global error $\mathcal{O}(h^4)$.
- *Acceleration-enhanced RK4*: $e_{\text{mid}} = \mathcal{O}(h^3) \implies$ potential for local truncation error $\mathcal{O}(h^6)$ and global error $\mathcal{O}(h^5)$.

A subtlety: the classical weights $\left(\frac{1}{6}, \frac{1}{3}, \frac{1}{3}, \frac{1}{6}\right)$ assume $\mathcal{O}(h^2)$-accurate intermediate states, and genuine fifth-order global convergence would additionally require verifying compatibility of the higher-accuracy intermediates with the quadrature weights. The main text therefore states conservatively that acceleration "improves estimation accuracy," without claiming strict fifth-order convergence.

Implicit vs. explicit curvature: detailed derivation. This appendix formalizes the "implicit vs. explicit curvature" argument of Section 2: (i) Theorem 2.1 implies that α_t is fully determined by the velocity field; (ii) Heun's corrector reconstructs this implicit α_t via two velocity queries, while an explicit α_t head attains the same accuracy at half the cost.

Algebraic view: second-order information is encoded in the first-order generator. Theorem 2.1 states that, under the continuity and interchange assumptions,

$$\mathcal{H}_t = \mathcal{G}_t^2 + \partial_t \mathcal{G}_t. \tag{82}$$

This is an *operator identity*: whenever $\mathcal{G}_t$ is sufficiently regular in space and time, $\mathcal{H}_t$ is uniquely determined by $\mathcal{G}_t$ and its temporal derivative. For deterministic flows $\dot{x} = u_t(x)$, where $[\mathcal{G}_t f](x) = \nabla f(x)^\top u_t(x)$, applying $\mathcal{G}_t$ twice (Appendix C.2) yields

$$[\mathcal{H}_t f](x) = u_t^\top \nabla^2 f u_t + \nabla f^\top \left(\partial_t u_t + (\nabla u_t) u_t\right). \tag{83}$$

A velocity-only model thus has, in principle, full access to α_t; leaving it implicit means any numerical scheme must *recover* it from velocity queries.

Numerical view: Heun's method recovers α_t at the cost of 2 NFE. For a C^1-smooth learned velocity field $u_t(x)$, one Heun step (2 NFE) from (t, x_t) with step size h reads:

$$x_{\text{pred}} = x_t + h u_t(x_t), \qquad x_{t+h} = x_t + \frac{h}{2}\left(u_t(x_t) + u_{t+h}(x_{\text{pred}})\right). \tag{84}$$

No explicit acceleration appears. Expanding the second velocity evaluation to first order about (t, x_t) gives

$$\begin{aligned}
u_{t+h}(x_t + h u_t) &= u_t(x_t + h u_t) + h \partial_t u_t(x_t + h u_t) + \mathcal{O}(h^2) && \text{(Temporal Taylor about } t\text{)} && (85)\\
&= \left[u_t(x_t) + h(\nabla u_t)(x_t) u_t(x_t) + \mathcal{O}(h^2)\right] \\
&\quad + h\left[\partial_t u_t(x_t) + \mathcal{O}(h)\right] + \mathcal{O}(h^2) && \text{(Spatial Taylor about } x_t\text{)} && (86)\\
&= u_t(x_t) + h\left(\partial_t u_t + (\nabla u_t) u_t\right)(x_t) + \mathcal{O}(h^2) && \text{(Collect } \mathcal{O}(h) \text{ terms)} && (87)\\
&= u_t(x_t) + h\alpha_t(x_t) + \mathcal{O}(h^2). && \text{(Def. acceleration field } \alpha_t\text{)} \\
& && && (88)
\end{aligned}$$

Substituting into the Heun update yields

$$x_{t+h} = x_t + \frac{h}{2}\left(u_t(x_t) + \left[u_t(x_t) + h\alpha_t(x_t) + \mathcal{O}(h^2)\right]\right) = x_t + h u_t(x_t) + \frac{h^2}{2}\alpha_t(x_t) + \mathcal{O}(h^3). \tag{89}$$

which coincides, up to $\mathcal{O}(h^3)$ local truncation error, with the *explicit* second-order Taylor step

$$x_{t+h} = x_t + h v_1 + \frac{h^2}{2} a_1 \tag{90}$$

of our second-order Euler sampler, with $(v_1, a_1) = \text{model}(x_t, t)$.

Two conclusions.

1. *Implicit recovery.* A first-order velocity field encodes α_t implicitly: Heun's two queries recover the $\frac{h^2}{2}\alpha_t$ Taylor term exactly.
2. *Cost asymmetry.* Implicit recovery costs 2 NFE/step; an explicit α_t head attains the same $\mathcal{O}(h^3)$ accuracy with 1 NFE (second-order Euler), accounting for the 1.3–2$\times$ NFE savings in Figure 3.

Higher-order velocity-only solvers (RK4, DOPRI5) recover further Taylor coefficients at proportionally greater NFE cost: first-order generator matching encodes curvature *implicitly*, and our framework makes it *explicit*, yielding strictly better accuracy per NFE.

D.3 Architecture and NFE-Regime Analysis

Model architecture.

Dual-head UNet. We employ a UNet backbone with a shared encoder and middle block, augmented by a second output head for acceleration prediction. Two configurations are considered:

- *Shared decoder* ($\approx 1.00\times$ baseline parameters): a single decoder processes the shared features, with separate final convolutions for velocity and acceleration; parameter-efficient when both outputs share low-level representations.

- *Independent decoder* ($\approx 1.60\times$ baseline): fully separate decoding paths, offering greater flexibility for datasets where the two outputs require distinct feature hierarchies.

Both heads use zero-module initialization: all weights and biases of the acceleration head start at zero, so $a_\theta(x,t) \equiv 0$ initially and the velocity head converges undisturbed in early training.

Classifier-free guidance (CFG). CFG is applied independently to both outputs. Let y denote the conditioning information (e.g., class label) and $\varnothing$ the unconditional input (null token). Then:

$$v_{\text{guided}} = (1+w)v_\theta(x,t,y) - wv_\theta(x,t,\varnothing), \tag{91}$$

$$a_{\text{guided}} = (1+w)a_\theta(x,t,y) - wa_\theta(x,t,\varnothing), \tag{92}$$

where w is the guidance scale (typically $w \in [1,3]$). The same w is applied to both fields for consistency, and the guided fields replace the raw outputs at inference.

NFE-regime: remainder-stability bound. This appendix gives the formal counterpart of the NFE-regime discussion in Section 2.6: why first-order solvers can match or surpass the second-order Taylor scheme at very low NFE.

Truncation error decomposition. Let $x(t)$ be the exact solution of $\dot{x} = v(x,t)$ on $[0,T]$, $h = T/N$ the step size, and assume $x \in C^4$. The exact one-step update admits the expansion

$$x(t+h) = x(t) + hv + \frac{1}{2}h^2 a + \frac{1}{6}h^3 \dddot{x}(t) + \frac{1}{24}h^4 x^{(4)}(\xi), \tag{93}$$

with $\xi \in (t, t+h)$. Subtracting the second-order Taylor step $\tilde{x}_{t+h} = x(t) + hv + \frac{1}{2}h^2 a$ gives the local truncation error

$$\tau_{2\text{nd}}(h) = \frac{1}{6}h^3 \dddot{x}(t) + \frac{1}{24}h^4 x^{(4)}(\xi) = \mathcal{O}(h^3), \tag{94}$$

whereas the first-order Euler step yields $\tau_{1\text{st}}(h) = \frac{1}{2}h^2 a + \mathcal{O}(h^3) = \mathcal{O}(h^2)$. Standard accumulation arguments turn these into global errors $\mathcal{O}(h^2)$ and $\mathcal{O}(h)$, respectively.

Where the asymptotic ranking breaks. The second-order scheme wins only when

$$\tfrac{1}{6}|\dddot{x}|\,h + \tfrac{1}{24}\|x^{(4)}\|_\infty h^2 < \tfrac{1}{2}|a|, \tag{95}$$

which holds for small h but fails at large h ($N \le 4$) when higher derivatives dominate $|a|$, precisely the endpoint-concentrated curvature regime of flow matching.

Stability-region argument. Linearising $\dot{x} = \lambda x$ ($z := \lambda h$) gives stability functions

$$R_{1\text{st}}(z) = 1 + z, \qquad R_{2\text{nd}}(z) = 1 + z + \tfrac{1}{2}z^2. \tag{96}$$

The curvature term raises consistency order without enlarging the absolute stability region $\mathcal{S} = \{z : |R(z)| \le 1\}$ (Hairer & Wanner, 1996): near the boundary $\frac{1}{2}z^2$ amplifies rather than damps perturbations, so $|R_{2\text{nd}}|$ can exceed $|R_{1\text{st}}|$ away from the origin. Together with equation 95, this predicts the observed crossover: first-order at ≤ 4 NFE, curvature strictly beneficial at moderate-to-high NFE (Section 3).

D.4 DPM-Solver++ for Flow Matching: 2V and 2A Variants and Diagnostics

We adapt DPM-Solver++ (Lu et al., 2022a) to flow matching by recasting our polynomial ($n = 2$) path as a noise schedule, yielding two variants for the SoF dual-head model: 2V ($\lambda_{\text{acc}} = 0$, velocity only) and 2A (velocity + analytic acceleration). Here λ_t denotes log-SNR, distinct from the acceleration weight $\lambda(e)$ in the main text.

Bridging the polynomial schedule and the diffusion form. The polynomial interpolant $X_t = t^2 X_1 + (1-t^2)X_0$ corresponds to $\alpha_t = t^2$, $\sigma_t = 1 - t^2$, whose log-SNR is

$$\lambda_t = \log\frac{\alpha_t}{\sigma_t} = \log\frac{t^2}{1-t^2}, \tag{97}$$

mapping $t \in (0,1)$ monotonically onto $\lambda \in (-\infty, +\infty)$, with $\frac{\mathrm{d}\lambda}{\mathrm{d}t} = \frac{2}{t(1-t^2)}$, i.e., $\frac{\mathrm{d}t}{\mathrm{d}\lambda} = \frac{t(1-t^2)}{2}$. The data prediction is recovered from the velocity by solving the linear system $x_t = t^2 X_1 + (1-t^2)X_0$, $\dot{X}_t = 2t(X_1 - X_0)$:

$$\hat{x}_1(x_t, t) = x_t + \frac{1-t^2}{2t}\, v_\theta(x_t, t), \tag{98}$$

which plugs directly into the DPM-Solver++ variation-of-constants formula

$$x_t = \frac{\sigma_t}{\sigma_s} x_s + \alpha_t \int_{\lambda_s}^{\lambda_t} e^{\tau - \lambda_t} \hat{x}_1(\tau)\, \mathrm{d}\tau. \tag{99}$$

We clip the time grid to $[\epsilon, 1-\epsilon]$ with $\epsilon = 10^{-3}$ to avoid λ-overflow at the endpoints, and use a cosine-spaced grid identical to that of our fixed-step solvers. **Caution.** Directly transplanting the CondOT-formula data prediction $\hat{x}_1 = x_t + (1-t)v_\theta$ in place of equation 98 is a natural but incorrect choice for polynomial schedules: the mismatch degrades the local order from $\mathcal{O}(h^3)$ to $\mathcal{O}(h^{1.1\text{–}1.3})$ and inflates FID at low NFE by up to $16\times$ (Appendix D.4). All DPM rows in the main tables use the schedule-matched formula.

Zero-acceleration degenerate DPM-Solver++(2V). Setting $\lambda_{\mathrm{acc}} = 0$, $\alpha_i = t_i^2$, $\sigma_i = 1 - t_i^2$, and approximating $\hat{x}_1(\tau)$ in equation 99 by linear interpolation through the two most recent data predictions, with $h_i = \lambda_i - \lambda_{i-1}$ and $r_1 = h_{i-1}/h_i$, yields

$$D^{(i)} = \left(1 + \frac{1}{2r_1}\right)\hat{x}_1^{(i-1)} - \frac{1}{2r_1}\hat{x}_1^{(i-2)}, \qquad x_i = \frac{1-t_i^2}{1-t_{i-1}^2}\, x_{i-1} + t_i^2\left(1 - e^{-h_i}\right) D^{(i)}, \tag{100}$$

with the first step degenerating to DDIM and $\hat{x}_1$ from equation 98; the variant consumes only v_θ and serves as a zero-acceleration reference under the same backbone and grid.

Acceleration-aware variant DPM-Solver++(2A). Write $g(t) = \frac{1-t^2}{2t}$ so that $\hat{x}_1 = x_t + g(t)\, v_\theta$. Differentiating along the sampling trajectory,

$$\frac{\mathrm{d}\hat{x}_1}{\mathrm{d}t} = v_\theta + g'(t)\, v_\theta + g(t)\, \frac{\mathrm{d}v_\theta}{\mathrm{d}t} = \frac{1-t^2}{2t}\left(a_\theta - \frac{v_\theta}{t}\right), \tag{101}$$

where $\frac{\mathrm{d}v_\theta}{\mathrm{d}t}$ is the total time derivative along the trajectory, identified with the acceleration output a_θ of the dual-head model, and we used $1 + g'(t) = -\frac{1-t^2}{2t^2}$. The λ-derivative follows from $\frac{\mathrm{d}t}{\mathrm{d}\lambda} = \frac{t(1-t^2)}{2}$:

$$\dot{\hat{x}}_1 := \frac{\mathrm{d}\hat{x}_1}{\mathrm{d}\lambda} = \frac{(1-t^2)^2}{4}\left(a_\theta - \frac{v_\theta}{t}\right). \tag{102}$$

Two observations. (i) For the exact conditional field, the path identity $a = \dot{v} = v/t$ gives $\dot{\hat{x}}_1 = 0$, so the integrand in equation 99 is constant and the scheme is exact in one step. (ii) For the learned model, $\|a_\theta - v_\theta/t\|$ measures cross-head consistency, and its $\frac{1}{t}$ amplification as $t \to 0$ is the dominant low-NFE error source of 2A (Appendix D.5). Substituting the first-order expansion $\hat{x}_1(\tau) \approx \hat{x}_1^{\mathrm{curr}} + (\tau - \lambda_t)\dot{\hat{x}}_1^{\mathrm{curr}}$ into equation 99 (with $\int_{-h}^{0} e^u\, du = 1 - e^{-h}$, $\int_{-h}^{0} e^u(h+u)\, du = h - 1 + e^{-h}$) yields the single-step update

$$x_{\mathrm{next}} = \frac{1-t_{\mathrm{next}}^2}{1-t_{\mathrm{curr}}^2}\, x_{\mathrm{curr}} + t_{\mathrm{next}}^2\left[C_0(h)\, \hat{x}_1^{\mathrm{curr}} + C_1(h)\, \dot{\hat{x}}_1^{\mathrm{curr}}\right], \tag{103}$$

with $C_0(h) = 1 - e^{-h}$, $C_1(h) = h - 1 + e^{-h}$; each step costs one dual-head forward pass, as in 2V.

Properties. Expanding $C_0 \approx h - h^2/2$, $C_1 \approx h^2/2$ shows the acceleration term recovers the second-order Taylor correction, and substituting $\hat{x}_1^{(i-2)} \approx \hat{x}_1^{(i-1)} - h\dot{\hat{x}}_1^{(i-1)} + \mathcal{O}(h^2)$ into the 2V update yields the same leading order, so 2A and 2V are asymptotically equivalent for uniform steps. Two advantages distinguish 2A: it attains local $\mathcal{O}(h^3)$ from the first step (2V degenerates to DDIM), and its analytic head avoids the $\mathcal{O}(h)$ finite-difference error of 2V whenever $\|a_\theta - a^*\| < \mathcal{O}(h)\|D^2\hat{x}_1\|$. At very low NFE, however, the $1/t$ amplification in equation 102 can invert this ranking (Appendix D.5).

Algorithms. Both variants are summarized below; they share the same compute profile (one NFE per step, no extra memory beyond the current state and, for 2V, one stored prediction).

Algorithm 1 DPM-Solver++(2V): multistep second-order sampling using only v_θ.

Require: dual-head model (v_θ, a_θ) (only v_θ used), cosine-spaced grid $t_0 < \cdots < t_N$ clipped to $[\epsilon, 1-\epsilon]$, initial $x \sim \mathcal{N}(0, I)$

$\hat{x}_1^{\text{prev}} \leftarrow$ None, $\quad \lambda_{\text{prev}} \leftarrow$ None
for $i = 0, \dots, N-1$ **do**
  $v \leftarrow \text{model}_v(x, t_i)$
  $\hat{x}_1 \leftarrow x + \frac{1-t_i^2}{2t_i} v$
  $\lambda_i \leftarrow \log \frac{t_i^2}{1-t_i^2}, \quad \lambda_{i+1} \leftarrow \log \frac{t_{i+1}^2}{1-t_{i+1}^2}, \quad h \leftarrow \lambda_{i+1} - \lambda_i$
  $r_1 \leftarrow (\lambda_i - \lambda_{\text{prev}})/h; \quad D \leftarrow \hat{x}_1$ *if* $\hat{x}_1^{\text{prev}} =$ None *else* $(1 + \frac{1}{2r_1})\hat{x}_1 - \frac{1}{2r_1}\hat{x}_1^{\text{prev}}$
  $x \leftarrow \frac{1-t_{i+1}^2}{1-t_i^2} x + t_{i+1}^2 (1 - e^{-h}) D$
  $x \leftarrow \text{clamp}(x, -5, 5); \quad \hat{x}_1^{\text{prev}} \leftarrow \hat{x}_1; \quad \lambda_{\text{prev}} \leftarrow \lambda_i$
end for
return x

Algorithm 2 DPM-Solver++(2A): single-step second-order sampling with acceleration head.

Require: dual-head model (v_θ, a_θ), cosine-spaced grid $t_0 < \cdots < t_N$ clipped to $[\epsilon, 1-\epsilon]$, initial $x \sim \mathcal{N}(0, I)$

for $i = 0, \dots, N-1$ **do**
  $(v, a) \leftarrow \text{model}(x, t_i)$
  $\hat{x}_1 \leftarrow x + \frac{1-t_i^2}{2t_i} v, \quad \dot{\hat{x}}_1 \leftarrow \frac{(1-t_i^2)^2}{4} \left(a - \frac{v}{t_i} \right)$
  $\lambda_i \leftarrow \log \frac{t_i^2}{1-t_i^2}, \quad \lambda_{i+1} \leftarrow \log \frac{t_{i+1}^2}{1-t_{i+1}^2}, \quad h \leftarrow \lambda_{i+1} - \lambda_i$
  $C_0 \leftarrow 1 - e^{-h}, \quad C_1 \leftarrow h - 1 + e^{-h}$
  $x \leftarrow \frac{1-t_{i+1}^2}{1-t_i^2} x + t_{i+1}^2 (C_0 \hat{x}_1 + C_1 \dot{\hat{x}}_1)$
  $x \leftarrow \text{clamp}(x, -5, 5)$
end for
return x

Role in our comparison. In Table 2, 2V and 2A share the same backbone, grid, and one-NFE-per-step budget, so their FID differences isolate the value of analytic acceleration; the formula-mismatched CondOT counterparts are in Appendix D.4.

Effect of the schedule-formula mismatch. Table 7 compares the mismatched CondOT formula with the schedule-matched one on the SoF dual-head model (50k, cosine grid). Unit tests on an analytic quadratic path show the matched prediction recovers the endpoint to $\sim 10^{-6}$ error, while the CondOT formula deviates by 11.3 at $t = 0.5$ (growing as $t \to 0$); the measured one-step order drops from 2.5–2.8 to 1.1–1.3. The mismatch inflates low-NFE FID by up to $16\times$ (75.31 vs. 4.53 at 8 NFE for 2V), vanishes asymptotically, and implies that transplanting CondOT formulas onto a polynomial-schedule model reports spuriously weak DPM rows.

D.5 Uniform-grid corrector distortion in first-order Heun.

The first-order Heun baseline of Table 2 originally used a *uniform* grid, giving FID 14.40 at 50 NFE, an order of magnitude worse than first-order Euler (reproduced: 14.37). With a uniform grid the Heun corrector evaluates v_θ at the predicted endpoint $x_{\text{pred}} = x_t + h v_t$, and near $t \to 0, 1$ the large, unrefined steps push x_{pred} outside the locally linear region of the velocity field, distorting the trapezoidal average; the cosine grid refines steps at both endpoints with the same step count and restores FID to 4.22 (Table 8), consistent with Appendix D.3. The effect is not schedule-specific: a polynomial-schedule velocity-only model shows the same pathology (22.46 → 2.58). The anomaly

is thus a property of the *grid*, not of the schedule or model family; all multi-step solver numbers in the revised tables use the cosine grid.

Table 7: CondOT formula vs. schedule-matched polynomial formula on CIFAR-10 (SoF dual-head model, 50k, cosine grid).

Method	8	20	50	100 NFE
DPM++ (2V), CondOT formula	75.31	3.28	2.46	2.44
DPM++ (2V), matched	**4.53**	**2.67**	**2.45**	2.44
DPM++ (2A), CondOT formula	34.46	7.28	3.60	2.85
DPM++ (2A), matched	**13.54**	**6.25**	**3.52**	**2.84**

Table 8: First-order Heun on CIFAR-10 (cifar10_first, cfg=0.3, 50k samples): inference-grid comparison across NFE budgets.

Time grid	20	50	100 NFE
Uniform (original Table 2)	54.43	14.40 (rep. 14.37)	4.77
Cosine (endpoint-refined)	**7.72**	**4.22**	**4.20**
Karras	24.25	4.80	4.31
Logit-normal	115.15	78.76	60.32

Diagnosis of 2A degradation at low NFE. With the schedule-matched formula, 2A still trails 2V at low NFE (13.54 vs. 4.53 at 8 NFE on CIFAR-10, 50k). Per-step probing ($\|\hat{x}_1\|$, $\|\dot{\hat{x}}_1\|$, and the $C_1\dot{\hat{x}}_1$ share of each update) identifies the mechanism: equation 102 contains the factor $1/t$, so cross-head residuals $a_\theta - v_\theta/t$ are amplified as $t \to 0$, and at large steps this amplified correction dominates the update. Four diagnostics corroborate the picture (measured in the CondOT-formula regime):

- a logit-normal grid, concentrating steps near $t \to 0$, is catastrophic (192 at 8 NFE);
- widening the clip to $\epsilon = 10^{-2}$ improves 8 NFE ($37.7 \to 23.5$) but degrades 20/50 NFE, an ϵ sweet spot rather than a cure;
- clamping the acceleration norm to $c = 2$ cuts 8-NFE FID from 37.7 to 7.75;
- a looser clamp ($c = 4$) worsens it to 92.7, so the fix is sensitivity-bounded, not monotonic.

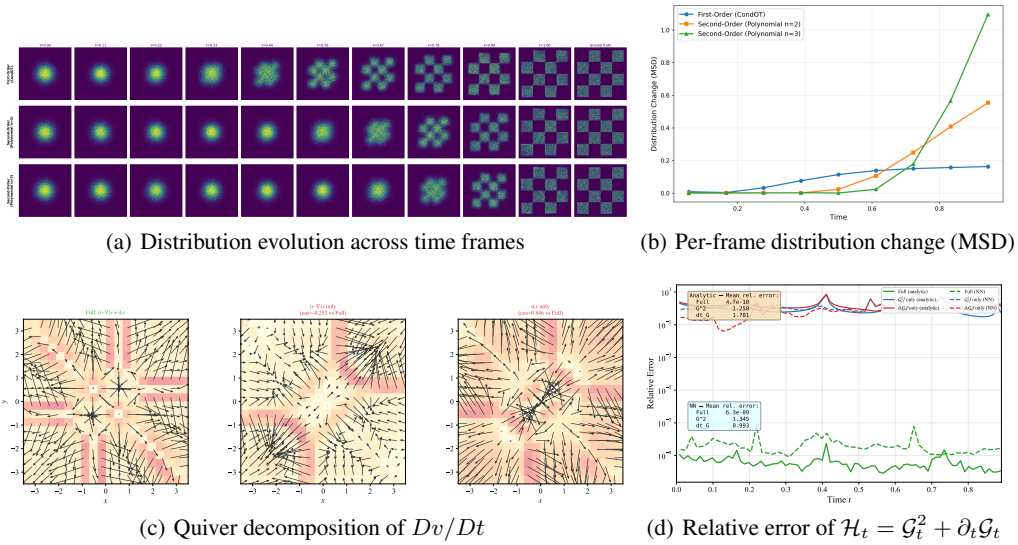
Crucially, the fix does not transfer across formula regimes: with the schedule-matched formula, $c = 2$ clamping *degrades* 8-NFE FID to 201.23 versus 13.54 unclamped; we leave this interaction to future work. We therefore report 2A *without* clamping and recommend 2V at very low NFE.

Alternative: sampling-time automatic differentiation. A natural alternative is to recover $\alpha_t = \partial_t v_\theta + (\nabla v_\theta) v_\theta$ at sampling time via Jacobian-vector products (Taylor-mode automatic differentiation), requiring no change to training. We implemented an AD-based second-order Euler step on the velocity-only baseline; two findings argue against this route:

- *Cost*: each step costs 2330.9 ms versus 197.4 ms for the dual-head forward pass (RTX 3090, batch 64), an $11.8\times$ overhead that cancels the NFE savings, whereas the dual head adds $< 2\%$ FLOPs (Table 1).
- *Accuracy*: AD-Taylor attains FID $10.77/8.64/8.35$ at $8/20/50$ NFE on the same first-order baseline where the trained dual head reaches ≈ 2.5 with second-order Euler,[1] since differentiating an imperfectly learned velocity field amplifies its errors, whereas supervising a_θ directly regularizes the curvature estimate.

[1]The AD-Taylor FID uses a 10k diagnostic subset; the dual-head reference uses the standard 50k protocol. A same-protocol 50k re-evaluation is in progress.

E SUPPLEMENTARY EXPERIMENTS



(a) Distribution evolution across time frames

(b) Per-frame distribution change (MSD)

(c) Quiver decomposition of Dv/Dt

(d) Relative error of $\mathcal{H}_t = \mathcal{G}_t^2 + \partial_t \mathcal{G}_t$

Figure 10: **Polynomial schedule & generator decomposition verification.** (a) Density snapshots: CondOT distributes transport uniformly while polynomial schedules ($n = 2, 3$) delay mass movement until $t > 0.5$, concentrating transformation as $t \to 1$ and producing non-degenerate $\ddot{X}_t \neq 0$. (b) MSD confirms the late-burst pattern with a sharp peak in the final 20% of time. (c) Quiver decomposition of $\frac{Dv}{Dt} = (v \cdot \nabla)v + \partial_t v$ at $t = 0.5$: convective and local-time-derivative components point in markedly different directions ($\cos \ll 1$), so neither alone captures the total acceleration. (d) Relative error of $\mathcal{H}_t f = \mathcal{G}_t^2 f + \partial_t \mathcal{G}_t f$: the full formula achieves near-machine precision ($\sim 10^{-15}$ analytic, $\sim 10^{-4}$ NN), whereas either term alone incurs $\mathcal{O}(1)$ error.

Ablation on the acceleration weight. Figure 7 (main text) sweeps $\lambda_{\text{target}} \in \{0, 0.1, 0.3, 0.5\}$ for the three fixed-step second-order solvers at 100 NFE against adaptive Dopri5. Even at $\lambda_{\text{target}} = 0$, 2nd-Heun and 2nd-RK4 match Dopri5; with acceleration supervision all fixed-step solvers surpass it in FID and IS, peaking at $\lambda_{\text{target}} = 0.3$ (our default) and regressing mildly at 0.5.

Sensitivity to velocity vs. acceleration perturbation. We extend the qualitative comparison of Figure 11 to a quantitative sweep, injecting additive Gaussian noise of magnitude $\varepsilon \in \{0.1, 0.3, 1, 3\}$ into either head during sampling (second-order Euler, 50 NFE; Table 9). The acceleration head is effectively immune: FID remains 7.81 across all ε, whereas identical noise on the velocity head degrades FID monotonically to 114.3 at $\varepsilon = 3$. Trajectory profiling explains the asymmetry: the acceleration correction $\|\frac{h^2}{2} a_\theta\|$ accounts for only $\sim 0.2\%$ of the velocity step $\|h v_\theta\|$ on average. We therefore characterize the learned acceleration as a *low-risk, fine-grained curvature correction*: it systematically improves per-step accuracy (Section 2.6) without introducing a new failure mode, and its supervision acts as a regularizer on the curvature estimate rather than a dominant trajectory driver.

Table 9: FID under per-head Gaussian perturbation ε (second-order Euler, 50 NFE, CIFAR-10, 10k subset).

Perturbed head	$\varepsilon = 0.1$	0.3	1.0	3.0
Velocity v_θ	7.78	7.54	9.57	114.30
Acceleration a_θ	7.81	7.81	7.81	7.81

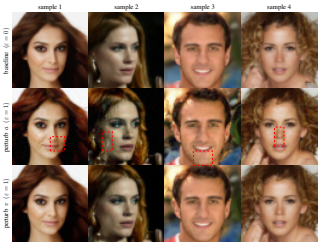


Figure 11: **Robustness to perturbations of v vs. a.** Top row: clean baseline ($\varepsilon = 0$). Middle row: acceleration head perturbed by Gaussian noise of magnitude $\varepsilon = 1$. Bottom row: identical perturbation applied to the velocity head. Red dashed boxes mark the few local artifacts under a-perturbation; v-perturbation causes markedly stronger global distortions.

Table 10: **Ablation Results for SoF.** FID (↓) across datasets and NFE budgets. Notation: C = conditional; S = shared decoder; Z = zero acceleration. Best per-column in **bold**, second-best underlined; "–" = not applicable (RK4 requires multiples of 4; "50" NFE for RK4 corresponds to 48 NFE). For CIFAR-10, models are trained at 32×32 resolution for 900 epochs and evaluated on 50k generated samples; CelebA-HQ does not include conditional generation experiments.

Sampling (Model)	CIFAR-10 (32×32, 900 epochs, 50k samples)						CelebA-HQ (64×64, 500 epochs, 30k samples)					
	2 NFE	4 NFE	8 NFE	20 NFE	50 NFE	100 NFE	2 NFE	4 NFE	8 NFE	20 NFE	50 NFE	100 NFE
1st Euler							357.22	244.43	92.66	37.78	5.89	**2.50**
1st Euler (C)	304.01	163.34	49.06	6.82	2.67	2.76						
1st Euler (C, S)	309.96	168.59	52.03	6.97	**2.52**	2.68						
2nd Euler							**95.58**	**45.68**	**17.75**	**6.51**	**3.46**	2.75
2nd Euler (Z)							357.27	330.77	137.35	52.90	11.85	4.02
2nd Euler (C)	134.62	46.45	18.10	6.67	3.66	2.96						
2nd Euler (C, Z)	304.01	283.32	93.21	14.09	3.24	2.57						
2nd Euler (C, S)	**125.97**	**44.13**	**17.83**	**6.66**	3.61	2.71						
2nd Euler (C, S, Z)	309.63	285.65	95.43	15.16	3.24	**2.51**						
1st Heun							**396.32**	**321.69**	**34.44**	48.55	32.03	15.80
1st Heun (C)	**344.46**	**258.66**	72.81	17.26	22.95	8.80						
1st Heun (C, S)	366.94	266.14	108.54	16.52	21.24	8.22						
2nd Heun							439.83	330.92	112.61	**24.47**	3.40	**2.56**
2nd Heun (Z)							396.37	322.19	140.58	30.34	**3.37**	2.60
2nd Heun (C)	475.82	298.66	123.75	14.06	2.63	2.51						
2nd Heun (C, Z)	**344.46**	**258.66**	74.36	21.51	2.56	2.50						
2nd Heun (C, S)	466.67	298.46	118.86	**13.29**	**2.48**	**2.41**						
2nd Heun (C, S, Z)	366.76	265.90	**70.96**	21.06	2.49	2.44						
1st RK4							–	302.69	32.94	**14.39**	10.47	4.45
1st RK4 (C)	–	277.03	104.65	**4.36**	4.04	3.03						
1st RK4 (C, S)	–	288.45	115.17	7.03	4.32	3.93						
2nd RK4							–	**236.18**	184.86	30.58	3.30	**2.46**
2nd RK4 (Z)							–	323.45	**22.68**	27.39	**2.93**	**2.46**
2nd RK4 (C)	–	265.90	134.49	18.42	2.72	2.53						
2nd RK4 (C, Z)	–	**233.42**	**103.72**	14.19	2.58	2.50						
2nd RK4 (C, S)	–	269.84	145.71	17.06	**2.57**	**2.43**						
2nd RK4 (C, S, Z)	–	246.25	120.67	13.75	2.58	2.46						

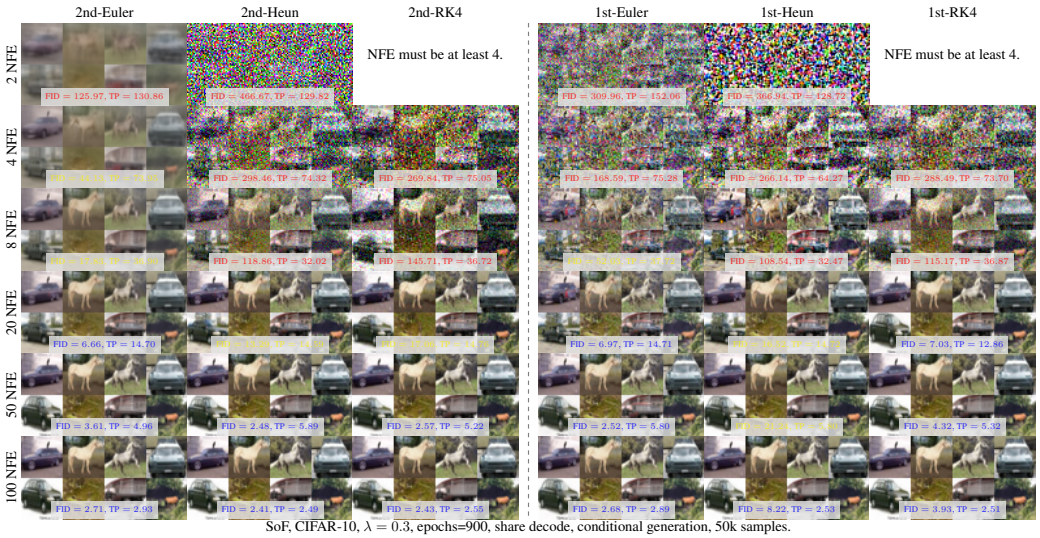
Table 10 confirms that the full second-order pipeline, comprising conditional generation, shared decoder, and zero-module initialization, consistently yields the lowest FID across both datasets and all NFE budgets, with 2nd-order Heun (C, S) attaining the best or second-best score in nearly every column; removing any component degrades performance, particularly at low NFE where first-order baselines remain uncompetitive.

Figures 12–15 provide qualitative sample comparisons across solvers on CIFAR-10 and at 128×128 resolution on CelebA-HQ, LSUN-Church, and LSUN-Classroom; FID color coding follows the main text.



FM, CIFAR-10, epochs=900, 50k samples.

FM, CelebA-HQ (64×64), epochs=500, 30k samples.



SoF, CIFAR-10, $\lambda = 0.3$, epochs=900, share decode, conditional generation, 50k samples.



SoF, CelebA-HQ (64×64), $\lambda = 0.3$, epochs=500, unshare decode, unconditional generation, 30k samples.

Figure 12: **Qualitative comparison across solvers.** FID color: red $\geq$ 100, yellow 10–100, blue $<$ 10.

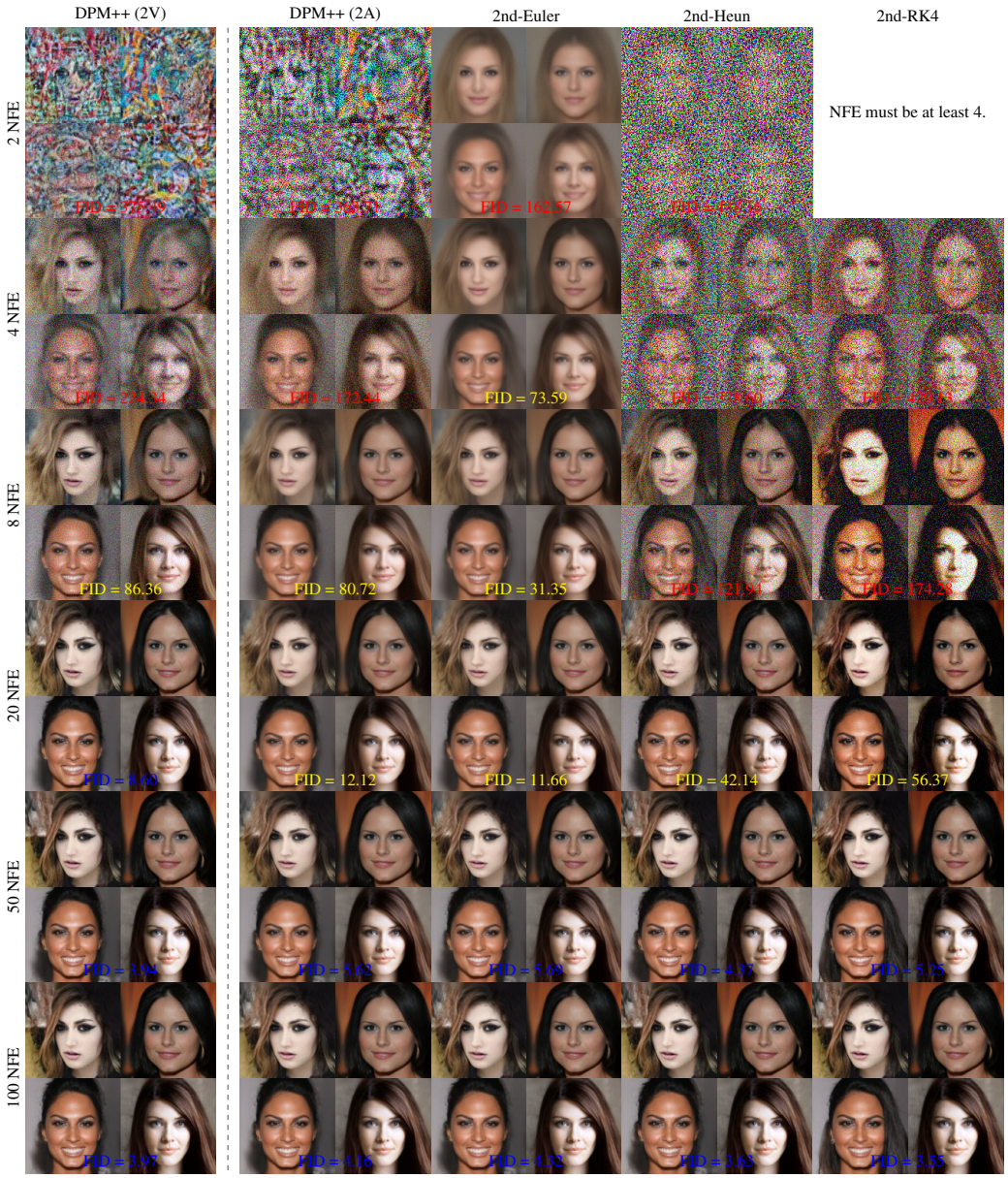


Figure 13: **SoF, CelebA-HQ (128 × 128), epochs=500, unshare decode, unconditional generation, 30k samples.** $\lambda = 0.3$ for all solvers except DPM++ (2V) which uses $\lambda = 0$. FID color: red ≥ 100, yellow 10–100, blue < 10.

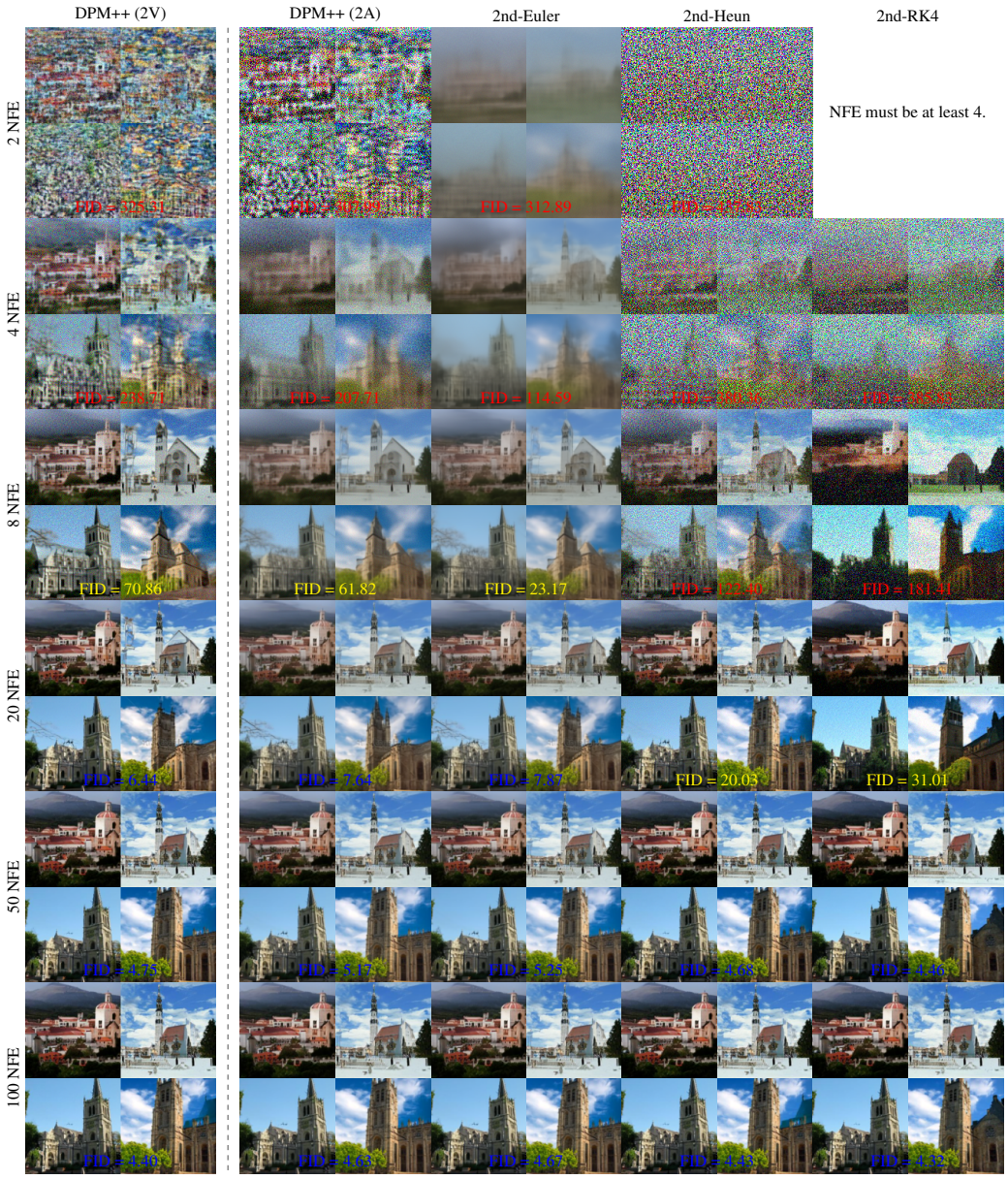


Figure 14: **SoF, LSUN-Church (128 × 128), epochs=500, unshare decode, unconditional generation, 30k samples.** $\lambda = 0.3$ for all solvers except DPM++ (2V) which uses $\lambda = 0$. FID color: red ≥ 100, yellow 10–100, blue < 10.

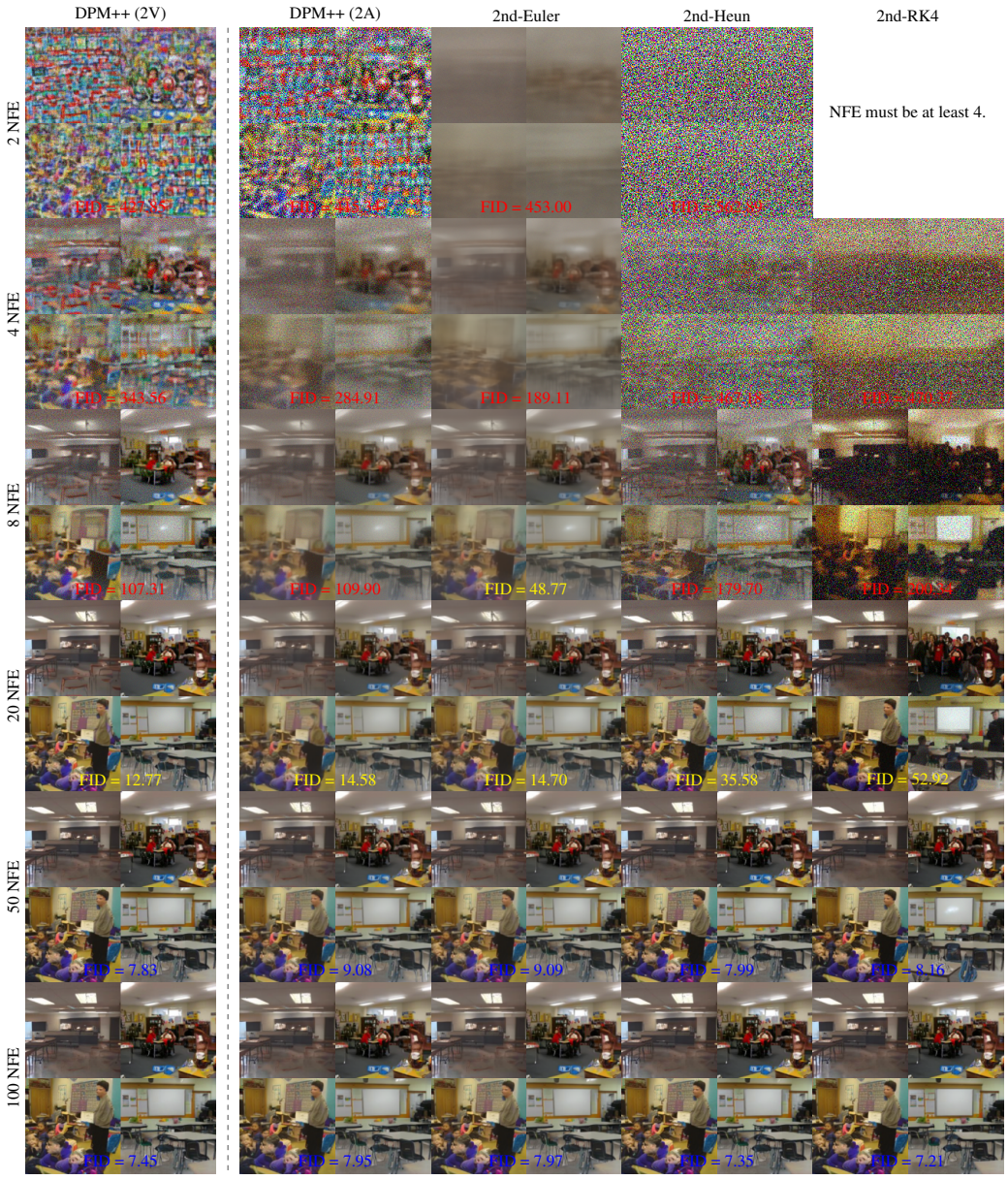


Figure 15: **SoF, LSUN-Classroom (128 × 128), epochs=500, unshare decode, unconditional generation, 30k samples.** $\lambda = 0.3$ for all solvers except DPM++ (2V) which uses $\lambda = 0$. FID color: red ≥ 100, yellow 10–100, blue < 10.