GeoSPRINT: Geometric Redundancy-Aware Step Pruning for Inference in Diffusion Trajectories

Arpita Joshi
The Scripps Research Institute
10550 N. Torrey Pines Road, La Jolla, San Diego CA, USA
arpitamhow@gmail.com / ajoshi@scripps.edu

Abstract

Diffusion models achieve high sample quality but remain expensive at inference time because sampling requires many sequential neural function evaluations (NFEs). Existing acceleration methods either use fixed step-skipping schedules, adapt step sizes based on local numerical error, or require additional training. We introduce GeoSPRINT (Geometric Step Pruning for Inference in Trajectories), a training-free framework for constructing non-uniform sampling schedules from the geometry of denoising trajectories. GeoSPRINT detects geometrically redundant steps using a hyperplanarity test in latent space, implemented efficiently via QR factorization, and converts the resulting redundancy profile into a sampling schedule that allocates more steps to high-curvature regions of the trajectory. In addition, we introduce the trajectory projection score α_{traj}, a residual-variance metric that quantifies trajectory straightness and serves as a training-free indicator correlated with trajectory rectification. Across CIFAR-10 (32×32), LSUN Church (256×256), and Stable Diffusion v1.5 (512×512 latent), GeoSPRINT improves over uniform DDIM (Denoising Diffusion Implicit Models) schedules at matched NFE budgets. On CIFAR-10, GeoSPRINT improves FID (Fréchet Inception Distance) by 0.7–1.1 over DDIM across 49–89 NFEs. Against the second-order DPM-Solver++, a first-order DDIM solver with a GeoSPRINT schedule is surpassed at low NFE but matches or exceeds it above a crossover budget (NFE$\approx$30) on both CIFAR-10 and SD v1.5. On LSUN Church and Stable Diffusion v1.5, where FID is measured against full-sampler DDIM references, GeoSPRINT reduces FID relative to DDIM by up to 1.06 and 1.93, respectively. These results show that trajectory geometry provides a useful global signal for allocating inference steps, and that above a moderate step budget, schedule quality can rival solver order without retraining.

1 Introduction

Diffusion models [1, 2] and their continuous-time generalizations through score-based stochastic differential equations have become the dominant generative modeling paradigm across images [3], video [4], audio [5], and molecular design [6]. Despite their exceptional generation quality, the iterative nature of the reverse sampling process remains a computational bottleneck: generating a single sample typically requires tens to thousands of sequential neural network evaluations.

The community has responded with a rich ecosystem of acceleration techniques. Training-free approaches include DDIM [7], which reinterprets diffusion sampling as a deterministic ODE and enables uniform step-skipping; DPM-Solver [8] and DPM-Solver++ [9], which employ exponential integrators with adaptive step sizes based on local truncation error; and UniPC [10], which unifies predictor-corrector schemes. Training-based approaches include progressive distillation [11], which iteratively halves the required steps; consistency models [12, 13], which learn to map any point on the trajectory directly to the endpoint; and rectified flow [14], which straightens transport paths to reduce integration steps.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Despite this progress, a gap remains: existing training-free methods lack a *global*, trajectory-level criterion for step importance grounded in geometric redundancy. DDIM skips steps uniformly, blind to trajectory geometry. DPM-Solver adapts step sizes based on *local* error estimates at individual timesteps. Adaptive non-uniform timestep sampling [15] focuses on *training* rather than inference. The recent SDM framework [16] analyzes local ODE stiffness but still operates pointwise. None of these methods ask the global question: *across the full trajectory, which steps actually contribute new geometric information to the path from noise to data?*

We argue that this is precisely the question addressed by geometric data instance reduction [17]. That work demonstrated that ordered sequences of data points can be dramatically reduced by testing geometric redundancy—collinearity in 2D, coplanarity in 3D—while preserving the essential shape and variance of the structure, as quantified by a projection score based on eigenvalues of the covariance matrix of removed points [18]. The key insight is that many consecutive points in an ordered dataset lie approximately on the same linear or planar subspace and can be removed without information loss.

A denoising trajectory $\{z_T, z_{T-1}, \ldots, z_0\}$ is precisely such an ordered sequence. In regions where the score function changes slowly, consecutive latent states trace a nearly linear path—these steps are geometrically redundant. In regions of rapid change (e.g., when the model resolves fine structure), the trajectory curves sharply, and every step carries new directional information. This analogy motivates GeoSPRINT.

Contributions: **(1)** We generalize geometric instance reduction from 2–3 PCA dimensions to arbitrary d-dimensional latent spaces via a *hyperplanarity test* with $O(d \cdot k^2)$ per-step complexity. **(2)** We introduce a novel schedule construction method that blends log-SNR spacing with GeoSPRINT's trajectory curvature density, yielding non-uniform schedules that outperform both uniform-t (DDIM) and uniform-logSNR spacing. **(3)** We introduce the *trajectory projection score* α_{traj}, a residual-variance metric that quantifies trajectory non-straightness; we prove it equals zero for perfectly straight trajectories and demonstrate a 450$\times$ drop from DDPM to DDIM, establishing it as a training-free diagnostic for trajectory straightness. **(4)** We characterize a consistent crossover, on both CIFAR-10 and SD v1.5, between a first-order DDIM solver with GeoSPRINT scheduling and the second-order DPM-Solver++: higher-order integration wins at low NFE, while GeoSPRINT matches or exceeds it above a crossover budget (NFE$\approx$30), showing that beyond that budget, *where* to step can matter as much as *how* to step. **(5)** We provide benchmarks on CIFAR-10 (32×32), LSUN Church (256×256), and Stable Diffusion v1.5 (512×512 latent), with consistent improvements over DDIM across resolutions, model architectures, and conditioning types.

2 Background and Related Work

2.1 Diffusion Models and the Sampling Problem

Score-based SDE framework. Song et al. [2] unified diffusion models under a continuous-time SDE framework. The forward process is $\mathrm{d}z = f(z,t)\,\mathrm{d}t + g(t)\,\mathrm{d}w$, where f and g define the drift and diffusion coefficients. The reverse process is:

$$\mathrm{d}z = \big[f(z,t) - g(t)^2 \nabla_z \log p_t(z)\big]\mathrm{d}t + g(t)\,\mathrm{d}\bar{w}, \tag{1}$$

where $\nabla_z \log p_t(z)$ is the score function approximated by a neural network $s_\theta(z,t)$ (and $\bar{w}$ is the standard Weiner process). An equivalent deterministic formulation, the probability flow ODE is:

$$\frac{\mathrm{d}z}{\mathrm{d}t} = f(z,t) - \tfrac{1}{2} g(t)^2 \nabla_z \log p_t(z). \tag{2}$$

Sampling reduces to solving this ODE from $t{=}T$ to $t{=}0$, requiring discretization into N steps, each involving one neural function evaluation (NFE). **Flow matching:** Lipman et al. [19] and Liu et al. [14] proposed learning a velocity field $v_\theta(z,t)$ that transports a source distribution to a target via $\mathrm{d}z/\mathrm{d}t = v_\theta(z,t)$. Rectified flow [14] straightens these paths through iterative reflow. Wang et al. [20] showed that strict straightness is not necessary—first-order ODE consistency suffices.

2.2 Existing Acceleration Methods

Uniform step-skipping. DDIM [7] enables deterministic sampling that skips timesteps uniformly, reducing 1000 steps to $\sim$50, but is blind to trajectory geometry.

High-order ODE solvers. DPM-Solver [8] derives exact solutions for the linear part of the diffusion ODE and applies exponential integrators for the nonlinear residual, achieving high quality in 10–20 steps. DPM-Solver++ [9] extends this to guided sampling. Both use *local* truncation error for adaptive step sizing with heuristic schedules (logSNR-uniform, time-uniform, time-quadratic).
Training-based methods. Progressive distillation [11] trains a student to match a teacher using $N/2$ steps, iteratively halving until 4-step generation. Consistency models [12, 13] learn direct trajectory-endpoint mappings. Consistency flow matching [21] combines consistency training with flow matching. All require additional training for each target NFE budget.
Adaptive and optimized timestep methods. Kim et al. [15] propose adaptive non-uniform timestep sampling for *training* acceleration. The SDM framework [16] analyzes local PF-ODE stiffness to set adaptive solver order and step size during *inference*, and Align Your Steps [22] optimizes a fixed sampling schedule for a given model and solver by minimizing an upper bound on the discretization-induced KL divergence. Adaptive solvers allocate steps from local quantities computed online along a single trajectory. GeoSPRINT instead allocates them from a global, offline statistic, where a family of reference trajectories is geometrically redundant and turns it into a fixed schedule that any solver can follow, without an error objective or iterative optimization. A direct empirical comparison with these methods remains open (Section 6).

2.3 Geometric Instance Reduction

Joshi and Haspel [17] introduced a data instance reduction algorithm that operates on ordered sequences projected into PCA space. The algorithm tests geometric redundancy via collinearity (2D) and coplanarity (3D) tests, removing points that lie approximately on the same linear subspace as their neighbors. Information loss is quantified by a *projection score* [18]:

$$\alpha = \frac{\sum_{j=1}^{p} \lambda_j^{(S)}}{\sum_{j=1}^{p} \lambda_j^{(\mathcal{Z})}}, \tag{3}$$

where $\lambda_j^{(S)}$ and $\lambda_j^{(\mathcal{Z})}$ are the j-th eigenvalues of the covariance matrices computed from the removed subset S and the full dataset $\mathcal{Z}$, respectively, and p is the number of retained PCA dimensions. The algorithm achieves 40–87% point reduction across diverse datasets (molecular trajectories, images, ML benchmarks) with projection scores as low as 10^{-5}.
Critical limitation for our purposes: the original algorithm operates in 2–3 PCA dimensions. Diffusion latent spaces are typically 32–512 dimensional. Section 3 addresses this directly.

3 Method: GeoSPRINT

3.1 Generalizing Geometric Redundancy to High Dimensions

Let $\{z_0, z_1, \ldots, z_N\}$ be an ordered sequence of points in $\mathbb{R}^d$ (latent states along a denoising trajectory, ordered from $t{=}T$ to $t{=}0$). We seek to identify and remove points approximately contained in the affine subspace spanned by their neighbors.

Definition 1 (Hyperplanarity test)**.** *Given a window of k* retained *points $W = \{w_1, w_2, \ldots, w_k\}$ (where $k \leq d$) and a candidate point z_i, define:*

$$M_W = \big[w_2 - w_1 \;\big|\; w_3 - w_1 \;\big|\; \cdots \;\big|\; w_k - w_1\big] \in \mathbb{R}^{d\times(k-1)}. \tag{4}$$

The residual distance of z_i from the affine subspace spanned by W is:

$$r_i = \big\|(I - M_W M_W^{+})(z_i - w_1)\big\|_2, \tag{5}$$

where M_W^{+} is the Moore-Penrose pseudoinverse. If $r_i < \tau$, then z_i is approximately in the affine span and is a candidate for removal.

Note that Definition 1 uses the most recently retained (not necessarily consecutive) points as the reference window, matching the causal sliding-window logic of Algorithm 2.
Efficient computation via QR factorization. Rather than computing the pseudoinverse directly, we compute the thin QR factorization $M_W = Q_W R_W$ and obtain $r_i = \|(I - Q_W Q_W^{\top})(z_i - w_1)\|_2$. Computing the QR factorization of $M_W \in \mathbb{R}^{d\times(k-1)}$ costs $O(d \cdot k^2)$ operations; since k is small (typically $k{=}2$), this is $O(d)$ in practice.
Adaptive window size. We propose a progressive hierarchy mirroring the 2D→3D progression

in the original algorithm: Tests are applied in a progressive hierarchy: **Level 1** (k=2, collinearity) catches long straight segments; **Level 2** (k=3, coplanarity) test if 4 points are coplanar, catches planar sweeps; **Level** ℓ (k=ℓ) tests against an $(\ell-1)$-dimensional affine subspace. These tests are applied sequentially: Level 1 first, then Level 2 on surviving points, etc. We use k=2 (collinearity) throughout; Appendix C.3 shows that the resulting curvature profile is stable for k=3 and k=5.

Threshold determination. Following Joshi and Haspel [17], we determine τ through binary search guided by the trajectory projection score (Section 3.3), adjusting until the projection score of removed points falls below a target (e.g., 10^{-3}) while maximizing the number of removed steps.

3.2 Application to Denoising Trajectories

Reference trajectory generation. Given a pretrained diffusion or flow model, we first generate B reference trajectories using the full N-step schedule. Each trajectory $\{z_T^{(b)}, \ldots, z_0^{(b)}\}$ provides an ordered sequence in $\mathbb{R}^d$.

Per-trajectory pruning. For each reference trajectory, we apply the hyperplanarity test (Algorithm 2) to identify and remove geometrically redundant timesteps, producing a per-trajectory retained set $\mathcal{R}^{(b)} \subseteq \{t_0, t_1, \ldots, t_T\}$.

Universal schedule aggregation. To obtain a fixed schedule usable without per-sample overhead, we aggregate across reference trajectories:

$$w(t) = \frac{1}{B} \sum_{b=1}^{B} \mathbf{1}\big[t \in \mathcal{R}^{(b)}\big]. \tag{6}$$

The retention frequency $w(t)$ encodes a per-timestep *curvature density*: it is high where trajectories consistently curve and low where they are consistently straight.

Schedule construction via log-SNR curvature blending. A naive approach selects the top-K timesteps by retention frequency $w(t)$. However, this clusters steps in high-curvature regions while leaving large gaps elsewhere, and the DDIM solver's prediction error grows with the gap in log-signal-to-noise ratio (log-SNR) between consecutive steps, not with the gap in t. We therefore construct schedules by blending two density functions:

$$\rho(t) = (1-\beta) \cdot \rho_{\text{logSNR}}(t) + \beta \cdot \rho_{\text{curv}}(t), \tag{7}$$

where ρ_{logSNR} is uniform density in log-SNR space (ensuring uniform error contribution per step, as in DPM-Solver), $\rho_{\text{curv}}(t) \propto \tilde{w}(t)$ is the smoothed and floored curvature density derived from Eq. 6, and $\beta \in [0,1]$ is the blend weight. Smoothing applies a Gaussian kernel with bandwidth σ=5 timesteps, followed by flooring at $\epsilon_{\text{floor}} = 0.05 \cdot \max_t \tilde{w}(t)$ to ensure minimum coverage. We then place K steps at uniform quantiles of the cumulative distribution function $F(t) = \int_0^t \rho(s)\,\mathrm{d}s \,/\, \int_0^T \rho(s)\,\mathrm{d}s$, analogous to importance sampling. Setting β=0 recovers log-SNR-uniform spacing (a known baseline). Setting β=1 gives pure curvature-weighted spacing (which fails at low NFE due to coverage gaps). We use β=0.6, selected once on CIFAR-10 (Section 5.3); the optimum is broad, and a sweep on SD v1.5 shows that the choice transfers (Appendix C.3). The uniform component guarantees coverage, while the curvature component moves steps toward the regions where the reference trajectories bend.

Sample-adaptive analysis. The universal schedule applies the same timestep selection to all samples. To understand how geometric complexity varies *across* samples, we additionally analyze each reference trajectory individually: for a given trajectory, the number of retained timesteps after pruning indicates that sample's geometric complexity. This per-sample analysis does *not* directly reduce NFEs at inference time—the model evaluation must still be performed to obtain each z_t—but it serves two important purposes: (i) it reveals the distribution of sample difficulty across a dataset (Section 5.6), informing whether a single universal schedule is sufficient or whether future work on predictive per-sample budgeting is warranted; and (ii) it provides per-sample α_{traj} values that can be correlated with sample quality metrics.

Remark on inference cost. The primary acceleration mechanism of GeoSPRINT is the *universal schedule* (Algorithm 1), which extracts a fixed set of timesteps offline and then generates all new samples using only those timesteps. The per-sample analysis described above is an offline diagnostic tool, not an inference-time accelerator.

3.3 Trajectory Projection Score

We extend the projection score of Joshi and Haspel [17] to denoising trajectories. The key insight is that the *residual variance*—the variance orthogonal to the local affine subspace at each pruned point—is the correct measure of information loss, not the raw variance of the pruned points themselves.

Definition 2 (Trajectory projection score)**.** *Given a trajectory $\mathcal{Z} = \{z_T, \ldots, z_0\}$ with $N{+}1$ points, let $S \subset \mathcal{Z}$ be the set of pruned points, and let r_s denote the hyperplanarity residual (Eq. 5) of each pruned point $z_s \in S$. The trajectory projection score is:*

$$\alpha_{\text{traj}} = \frac{\sum_{s \in S} r_s^2}{\sum_{z \in \mathcal{Z}} \|z - \bar{z}\|^2}, \tag{8}$$

where $\bar{z}$ is the global trajectory mean. The numerator is the total squared residual of all pruned points: the variance that lies outside the affine span of their retained neighbors. The denominator is the total trajectory variance.

α_{traj} measures the fraction of total trajectory variance that is *lost* by pruning—specifically, the variance orthogonal to what the retained points can reconstruct through local affine interpolation. A low score (e.g., 10^{-3} to 10^{-5}) indicates that removed steps contributed negligibly to the trajectory's geometric structure beyond what is already captured by their neighbors.

Proposition 1. *For a perfectly straight trajectory (all points on a single line in $\mathbb{R}^d$), GeoSPRINT with the collinearity test (k=2) prunes all points beyond the initial window, retaining exactly the first k points, with $\alpha_{\text{traj}} = 0$.*

Proof. If all points are collinear, every candidate point z_i (for $i \geq k$) lies exactly on the line through the retained window points, so the hyperplanarity residual $r_i = 0$ for all such i. Because all residuals satisfy $r_i < \tau$, the window never updates (the else branch of Algorithm 2 is never executed), and the retained set consists of the first k points used to initialize the window. The numerator $\sum r_i^2 = 0$, hence $\alpha_{\text{traj}} = 0$. □

Corollary 1. *α_{traj} is a measure of trajectory non-straightness. For rectified flow models, which aim to straighten trajectories, we therefore expect α_{traj} to decrease with each round of reflow, suggesting its use as a training-free indicator that correlates with rectification. We present it as a geometric correlate of straightness rather than a certified measure of flow-matching optimality.*

3.4 Connections to Existing Methods

Relation to DPM-Solver. DPM-Solver's adaptive step size is based on *local truncation error*—the difference between a higher-order and lower-order solution at each step. GeoSPRINT's hyperplanarity test is a *geometric* criterion over a *window* of points, making it sensitive to trajectory structure over multiple steps rather than single-step error. The two are complementary: DPM-Solver minimizes local numerical error; GeoSPRINT minimizes global geometric redundancy.

Relation to SDM. The SDM framework [16] analyzes local PF-ODE stiffness to determine solver order at each timestep. GeoSPRINT instead operates on the *realized trajectory* post-hoc, testing whether the path actually traced contains redundant segments. SDM is predictive (deciding before taking the step); GeoSPRINT is retrospective (analyzing after generation). This makes GeoSPRINT applicable to any model without Jacobian access. An empirical comparison of these complementary strategies is an important direction for future work.

Relation to rectified flow. Rectified flow learns transport paths that are inherently straight, making all interior steps redundant by construction. GeoSPRINT *diagnoses* rather than *enforces* straightness, and works equally well on curved trajectories by adaptively allocating steps where curvature demands them.

Relation to consistency models. Consistency models learn to jump from any trajectory point to the endpoint in one step. GeoSPRINT identifies which intermediate points are *necessary* to maintain trajectory integrity. The two can be combined: GeoSPRINT identifies minimal waypoints, and a consistency model could jump between them.

4 Theoretical Analysis

4.1 Reconstruction Error Bound

Theorem 1 (Reconstruction error bound)**.** *Let $\mathcal{Z}$ be a trajectory of $N{+}1$ points in $\mathbb{R}^d$, and let S be the set of pruned points after GeoSPRINT with hyperplanarity threshold τ. Then:*

$$\alpha_{\text{traj}} \leq \frac{|S| \cdot \tau^2}{(N{+}1) \cdot \text{tr}(C_{\mathcal{Z}})}, \tag{9}$$

where $|S|$ is the number of pruned steps and $\text{tr}(C_{\mathcal{Z}})$ is the trace of the sample covariance of $\mathcal{Z}$.

Proof sketch. Each pruned point z_s satisfies $r_s \leq \tau$ by the hyperplanarity test, so $\sum_{s \in S} r_s^2 \leq |S| \cdot \tau^2$. The denominator of α_{traj} equals $(N{+}1) \cdot \text{tr}(C_{\mathcal{Z}})$. Division yields the result. The full proof is in Appendix B. □

This provides a formal guarantee: GeoSPRINT's reconstruction error—the fraction of total trajectory variance attributable to orthogonal residuals of pruned points—is controlled by τ, the pruning rate $|S|/(N{+}1)$, and the total trajectory variance. Because the bound is quadratic in τ, a desired $\alpha_{\text{traj}} \leq \epsilon$ is achieved with a square-root dependence on ϵ: it suffices to set $\tau \leq \sqrt{\epsilon \cdot (N{+}1) \cdot \text{tr}(C_{\mathcal{Z}})/|S|}$. We stress the scope of this result: it bounds only the orthogonal reconstruction residual of the pruned points. It does not bound ODE integration error, distributional discrepancy such as FID, or perceptual quality, and we do not claim that the empirical FID gains follow from it; the guarantee is consistent with, but does not imply, good sample quality.

Remark. *We emphasize that α_{traj} bounds the* orthogonal reconstruction residual*, not the retained set's sample covariance $\text{tr}(C_R)/\text{tr}(C_{\mathcal{Z}})$. Pruning points changes the sample mean and can reduce the retained sample covariance even when all residuals are zero (e.g., removing interior points from a collinear trajectory reduces sample variance while $\alpha_{\text{traj}} = 0$). The reconstruction interpretation—that pruned points are well-approximated by affine interpolation from their retained neighbors—is the operationally relevant guarantee for schedule quality.*

4.2 Complexity Analysis

The multi-level hyperplanarity test with maximum level ℓ has complexity $O(N \cdot d \cdot \ell^2)$ per trajectory in the worst case. For Level-1 only (collinearity, $k{=}2$), this is $O(N \cdot d)$—linear in both trajectory length and dimension, and negligible compared to the cost of the NFEs themselves. In Algorithm 2, the QR factorization is cached and recomputed only when the window updates, further reducing the amortized cost. Schedule amortization over a batch of M samples reduces per-sample overhead to $O(B/M)$ reference trajectories.

4.3 Relation to Trajectory Curvature

The hyperplanarity residual r_i at point z_i relates to discrete curvature. For a smooth trajectory $z(t)$ with a collinearity window ($k{=}2$) using causal forward-extrapolation from retained points at $z(t)$ and $z(t{+}\Delta t)$ to the candidate $z(t{+}2\Delta t)$:

$$r_i \approx (\Delta t)^2 \left\| z''_\perp(t) \right\|_2 + O\big((\Delta t)^3\big), \tag{10}$$

where $z''_\perp(t)$ is the component of $z''(t)$ orthogonal to the velocity $z'(t)$. Tangential acceleration (speeding up along a straight path) does not contribute, since the residual measures orthogonal distance from the extrapolated line. The coefficient $(\Delta t)^2$ (rather than $(\Delta t)^2/2$) reflects causal forward-extrapolation rather than midpoint interpolation. Removing points with $r_i < \tau$ is thus equivalent to removing steps where the trajectory's orthogonal curvature is small. Higher-level tests detect higher-order geometric structure (torsion, etc.).

5 Experiments

5.1 Setup

Models and datasets. We evaluate on: **CIFAR-10** [23] ($32{\times}32$) using the pretrained `google/ddpm-cifar10-32` UNet from HuggingFace Diffusers; **LSUN Church** [24] ($256{\times}256$) using `google/ddpm-ema-church-256`; and **Stable Diffusion v1.5** [3] ($512{\times}512$, latent $4{\times}64{\times}64$) using `runwayml/stable-diffusion-v1-5` with classifier-free guidance (CFG scale 7.5). All pixel-space models use the standard linear β-schedule with 1000 training timesteps.

Protocol. Schedule construction and evaluation proceed in four stages with strictly separated data. **(i) Reference trajectories.** For each model we record B full DDIM trajectories on an N-step pool: $B{=}100$, $N{=}200$ for CIFAR-10; $B{=}50$, $N{=}200$ for LSUN Church; $B{=}30$, $N{=}50$ for SD v1.5 (Appendix C.1 analyzes convergence in B), using generator seeds $0, \ldots, B{-}1$. **(ii) Curvature profile.** For each reference trajectory we run the QR hyperplanarity test, binary-searching τ to a fixed probe retention, and average the retained-step indicator across the B trajectories to obtain the curvature density $w(t)$ (Eq. 6). Here τ is an internal variable of the profile computation, not a user-facing knob. **(iii) Schedule at a budget K.** For a target NFE budget K, we form the blended density (Eq. 7, $\beta{=}0.6$) and place K timesteps at uniform quantiles of its cumulative distribution;

K is an input to this stage. **(iv) Evaluation.** For each K we generate fresh samples: 10,000 for CIFAR-10 and LSUN Church and 5,000 for SD v1.5, (Appendix C.2) with generator seeds starting at 10,000, disjoint from the reference seeds, and compute FID, using a fixed set of prompts randomly sampled from the COCO 2017 validation set. For CIFAR-10, FID is computed with clean-FID [25] against the training set; for LSUN Church and SD v1.5 it is computed against reference samples from the full sampler (200-step and 50-step DDIM, respectively), so it measures relative trajectory fidelity rather than absolute generation quality (Section 5.2.1). All sampling uses the manual DDIM update formula to correctly handle non-uniform timestep spacing. All reported FID values are means $\pm$ standard deviation over 3 independent runs with different random seeds.

GeoSPRINT does *not* select a single K by thresholding α_{traj}. We sweep K to trace a Pareto curve: each reported point corresponds to one budget K, and the horizontal axis of figures 1 and 2 is the resulting NFE. The projection score α_{traj} is a *reported diagnostic per budget*, computed by applying Eq. 8 to the retained set induced by a given schedule (averaged over reference trajectories); it characterizes the schedule but does not determine K. The only tuned hyperparameter is the blend weight β, selected once on CIFAR-10 (Section 5.3) and applied unchanged to Church and SD; window size is fixed at $k{=}2$ throughout.

Baselines. DDIM [7]: uniform timestep spacing at matched NFE. **DPM-Solver++** [9]: second-order solver with default logSNR-uniform schedule, `solver_order=2`. **DDIM-200**: proxy for zero-pruning baseline.

5.2 Main Result: FID vs. NFE Across Datasets

GeoSPRINT operates as a *step-budget oracle*: at each K, we construct a LogSNR+curvature schedule (Eq. 7, $\beta{=}0.6$) and compare with DDIM uniform spacing at matched NFE. On CIFAR-10 and SD v1.5, we additionally compare with DPM-Solver++ (second-order, default schedule).

Table 1: FID (↓) across datasets. GeoSPRINT uses the blended schedule ($\beta{=}0.6$). **Bold**: best per row; Δ = GeoSPRINT $-$ DDIM. CIFAR-10 FID is against the training set (DDIM-200: 12.65). Church and SD FIDs are against 200-step and 50-step DDIM references, respectively, and measure relative trajectory fidelity (Section 5.2.1). On both CIFAR-10 and SD v1.5, DPM-Solver++ is better at low NFE and GeoSPRINT above a crossover near NFE$\approx$30.

CIFAR-10 (32×32)

NFE	Geo	DDIM	Δ	DPM++
10	38.53±.31	34.99±.28	3.54	**29.05**±.25
20	22.23±.19	23.17±.21	−0.94	**21.31**±.18
30	**18.50**±.15	19.33±.17	−0.83	18.85±.16
49	**15.95**±.13	16.82±.14	−0.87	17.41±.15
60	**15.25**±.12	15.95±.13	−0.70	17.08±.14
70	**14.80**±.11	15.69±.12	−0.89	16.82±.14
80	**14.38**±.10	15.30±.12	−0.92	16.68±.13
89	**14.10**±.10	15.17±.11	−1.07	16.65±.13

Church (256×256)

NFE	Geo	DDIM	Δ
52	**1.26**±.04	1.48±.05	−0.22
56	**1.16**±.03	2.22±.07	−1.06
74	**0.87**±.03	1.72±.05	−0.85
79	**0.79**±.02	1.43±.04	−0.64

SD v1.5 (512×512)

NFE	Geo	DDIM	Δ	DPM++
15	**7.91**±.08	8.72±.09	−0.81	**6.07**
20	**5.51**±.06	5.97±.07	−0.46	**4.88**
24	**4.60**±.05	5.92±.06	−1.32	**4.37**
29	**4.01**±.05	4.92±.05	−0.91	4.07
34	**3.66**±.04	4.23±.05	−0.57	4.01
39	**3.51**±.04	4.92±.06	−1.41	4.00
44	**3.45**±.04	5.38±.06	−1.93	3.95

Key findings. **(1)** GeoSPRINT outperforms DDIM at every NFE on all three datasets (except NFE$=8$ on SD and NFE$=10$ on CIFAR-10, see Figure 1(a) and Figure 2). **(2)** Improvements *grow* with model complexity: -0.7 to -1.1 on CIFAR-10, -0.2 to -1.1 on Church, and up to -1.93 on SD v1.5 (Table 1), although on Church and SD the FID measurements are against DDIM references. **(3)** DDIM's uniform schedule becomes actively counterproductive at high NFE on both Church and SD (FID increasing), while GeoSPRINT monotonically improves (Table 1 and Figure 2). **(4)** Against DPM-Solver++, a first-order DDIM solver with a GeoSPRINT schedule exhibits a consistent crossover on *both* CIFAR-10 and SD v1.5 (Table 1): the second-order solver is superior at low NFE, while GeoSPRINT matches or surpasses it above a crossover budget (NFE$\approx$30). Beyond this budget, schedule quality can outweigh solver order; below it, higher-order integration dominates. GeoSPRINT is a schedule and therefore composes with any solver. **(5)** GeoSPRINT at 60 steps matches DDIM at 80 steps on CIFAR-10 (both FID$\approx$15.3), a 25% NFE reduction.

5.2.1 Note on FID References for LSUN Church and Stable Diffusion

For LSUN Church and SD v1.5, FID is computed against reference samples from the full sampler, 10,000 DDIM-200 samples for Church and 5,000 DDIM-50 samples for SD, rather than against a ground-truth image set. These FIDs therefore measure *relative trajectory fidelity*: how closely a reduced schedule reproduces the full sampler's output distribution, not absolute generation quality.

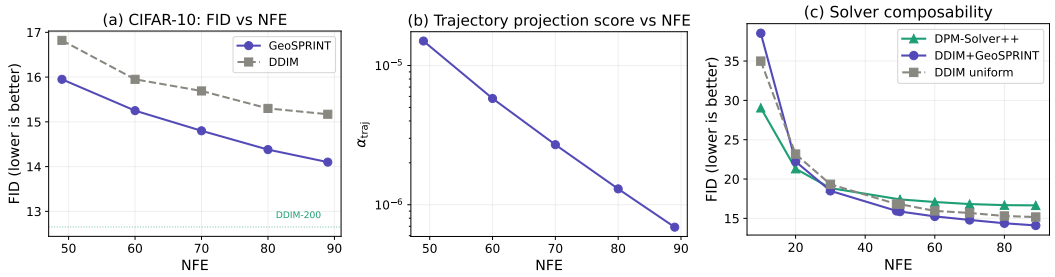

Figure 1: CIFAR-10 results. (a) FID vs. NFE. GeoSPRINT consistently outperforms DDIM. Dashed lines: DDIM-200 baseline (12.65) and DDIM-50 reference (16.80). Error bands: ± 1 std over 3 seeds. (b) α_{traj} decreases log-linearly with NFE, providing a principled knob for the step-budget oracle. (c) A first-order DDIM solver with a GeoSPRINT schedule is surpassed by the second-order DPM-Solver++ at low NFE but matches or exceeds it above a crossover near NFE$\approx$30, beyond that budget, *where* to step matters more than *how* to step.

This isolates the effect of the schedule from the base model's own generation gap, but Church and SD results should not be read as gains in real generation quality.

5.3 Blend Parameter Ablation

We ablate the blend parameter β (Eq. 7) on CIFAR-10; results are summarized in Table 2 (Left). DDIM uniform achieves 23.17/19.33/16.80 at NFE=20/30/50. Pure log-SNR (β=0) is worse than

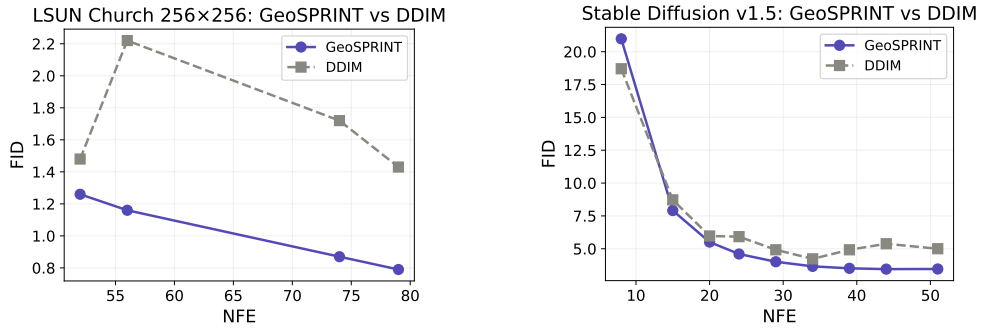

Figure 2: FID vs. NFE on LSUN Church 256×256 (left) and Stable Diffusion v1.5 (right), measured against 200-step and 50-step DDIM references, respectively (Section 5.2.1). GeoSPRINT improves steadily with NFE (flattening beyond 44 steps on SD), while DDIM's uniform spacing plateaus or worsens at larger budgets. Error bands: ± 1 std over 3 seeds.

Table 2: **Left:** Blend ablation on CIFAR-10 (FID ↓, mean of 3 seeds). β=0: pure log-SNR; β=1: pure curvature. Optimum at $\beta \in [0.5, 0.7]$. DDIM uniform: 23.17/19.33/16.80. **Right:** α_{traj} across schedulers (450× drop DDPM→DDIM).

β	0	0.3	0.5	0.6	0.7	1.0
20	25.81	22.45	22.26	22.23	**22.19**	22.68
30	20.89	18.71	18.56	**18.50**	18.52	18.86
50	16.80	16.03	15.91	**15.87**	15.94	16.04

Scheduler	α_{traj}	Pruned
DDPM	.408±.024	79%
DDIM	.0009±.0001	82%
DPM++	.0008±.0002	84%

DDIM uniform, confirming that log-SNR spacing alone is not superior—it is GeoSPRINT's curvature correction that makes the difference. Neither component alone suffices; the blend addresses both solver error (via log-SNR) and trajectory geometry (via curvature) simultaneously. The optimum is broad ($\beta \in [0.5, 0.7]$ on CIFAR-10). On SD v1.5, a sweep over $\beta \in \{0.3, 0.5, 0.6, 0.7, 0.9\}$ changes FID by at most 0.53 at every tested budget (Appendix C.3), so β=0.6 is used unchanged for all models.

5.4 Schedule Analysis

The curvature density profile $w(t)$ reveals where GeoSPRINT concentrates steps. The late denoising zone (low noise, $t/T < 0.2$) has mean retention $2\times$ higher than the middle zone ($0.2 < t/T < 0.8$), confirming that fine-detail resolution drives step allocation. The early zone ($t/T > 0.8$) also shows elevated retention, reflecting the initial direction-setting phase (Figure 3).

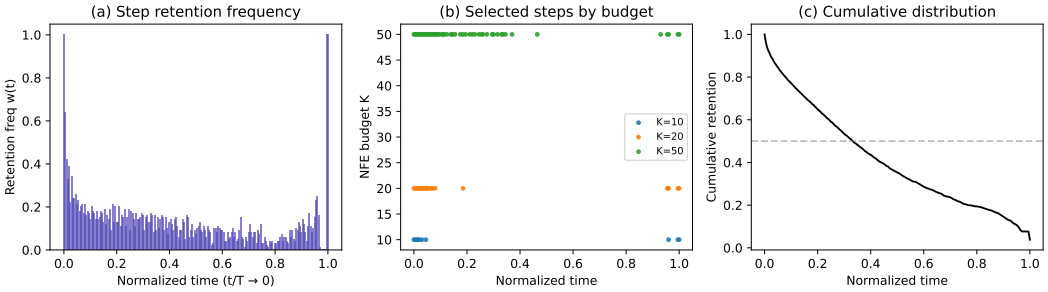

Figure 3: (a) Retention frequency $w(t)$ peaks at both ends (bimodal). (b) Selected timesteps at budgets $K{=}10, 20, 50$. (c) Cumulative: 50% of retained steps lie in the final 20% of denoising.

5.5 α_{traj} as a Straightness Diagnostic

We record 50 trajectories from three scheduler types on the same pretrained CIFAR-10 model. α_{traj} drops $450\times$ from DDPM to DDIM (Table 2, Right; Figure 4), consistent with deterministic probability-flow trajectories being markedly straighter than stochastic DDPM ones. We read this as evidence that α_{traj} tracks trajectory straightness in a training-free way, rather than as a certified measure of flow-matching quality.

5.6 Supporting Experiments (Appendix C)

Reference set convergence: Schedule stability reaches fuzzy Jaccard 0.84 and correlation 0.86 at $B{=}75$ reference trajectories; $B{=}100$ is adequate for CIFAR-10 (Figure 5). **Per-sample complexity**: NFE variation is modest on CIFAR-10 (1.2–1.4$\times$), but individual curvature profiles are diverse (mean correlation 0.37 with universal), validating the universal schedule as a robust compromise across samples (Figure 6).

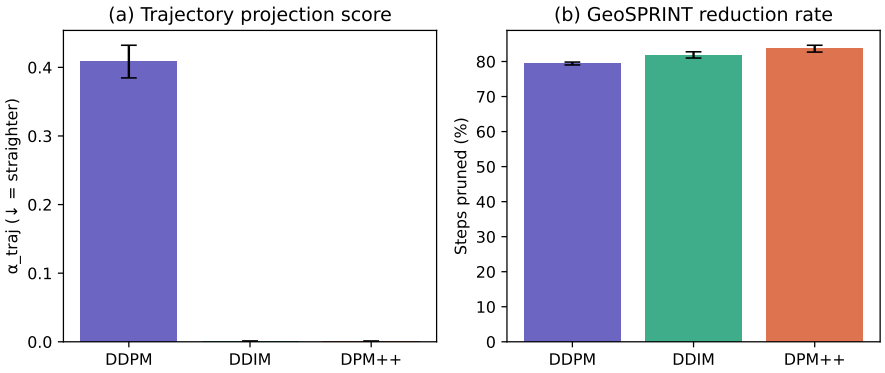

Figure 4: (a) α_{traj} across schedulers: $450\times$ drop DDPM$\rightarrow$DDIM. (b) Reduction rate: straighter trajectories are more compressible.

6 Discussion

Why does curvature-aware scheduling help? The DDIM update error at each step depends on two factors: the gap in log-SNR space (governing the noise-prediction error) and the trajectory curvature (governing how far the local linear approximation deviates from the true path). Uniform-t spacing equalizes neither. Log-SNR spacing equalizes the first but ignores the second. GeoSPRINT's curvature density captures the second, and the blend (Eq. 7) addresses both simultaneously.

Schedule quality vs. solver order. On both CIFAR-10 and SD v1.5, a first-order DDIM solver with a GeoSPRINT schedule is surpassed by the second-order DPM-Solver++ at low NFE but matches or

exceeds it above a crossover near NFE$\approx$30 (Table 1). One reading is that at moderate budgets the dominant error comes from where steps are placed, while at very low budgets the per-step accuracy of the integrator dominates. Because GeoSPRINT only selects timesteps, the natural next step is to pair its schedules with higher-order solvers rather than to compete with them.

Behavior across models. On CIFAR-10, the gain over DDIM holds against real data at every budget from 49 to 89 NFEs. On Church and SD, measured against DDIM references, DDIM's uniform spacing stops improving at larger budgets, on SD its FID rises from 4.23 at NFE=34 to 5.38 at NFE=44, whereas GeoSPRINT continues to approach the reference, indicating that curvature-aware placement tracks the converged trajectory better as the budget grows. Whether this translates into larger *absolute*-quality gains on large models requires evaluation against real images, which we leave to future work.

Limitations: We compare against DDIM and, on CIFAR-10 and SD v1.5, DPM-Solver++, but not against optimized schedules such as Align Your Steps [22] or stiffness-based adaptive solvers such as SDM [16]; this comparison is needed to situate GeoSPRINT and is our most important next step. The universal schedule requires B full reference trajectories offline (20,000 NFEs for CIFAR-10, 10,000 for Church, 1,500 for SD) which dominates for single-sample generation; the method targets batch settings where this cost is amortized. The blend parameter β=0.6 was selected on CIFAR-10; a sweep on SD shows a broad optimum, but other architectures, guidance scales (we used CFG 7.5), and prompt distributions (we used ten prompts) remain untested, as does transferring a schedule extracted with DDIM to other solvers or checkpoints without re-extraction. Absolute-quality evidence comes only from CIFAR-10: Church and SD FIDs are measured against DDIM references, so conclusions there concern convergence to the full sampler. Finally, GeoSPRINT selects *which* timesteps to use but does not improve the solver *at* each timestep; it is complementary to, not a replacement for, high-order ODE solvers.

Broader Impact: GeoSPRINT's training-free nature makes fast diffusion sampling accessible without distillation or retraining costs. The trajectory projection score provides an interpretable training-free indicator of trajectory straightness. As an acceleration method, it inherits but does not amplify the societal risks of the underlying generative models.

7 Conclusion

We have presented GeoSPRINT, a training-free framework that constructs non-uniform sampling schedules for diffusion models by analyzing the geometry of denoising trajectories. The key technical contribution is a blended schedule construction that combines log-SNR spacing (matching the ODE solver's error profile) with trajectory curvature density (derived from a high-dimensional hyperplanarity test) which yields a schedule that is reused for all samples. On CIFAR-10, these schedules improve on uniform DDIM by 0.7–1.1 FID across 49–89 NFEs. Against the second-order DPM-Solver++, a first-order DDIM solver with a GeoSPRINT schedule shows a consistent crossover on CIFAR-10 and SD v1.5: it is surpassed at low NFE but matches or exceeds DPM-Solver++ above roughly 30 steps. On LSUN Church and SD v1.5, measured against full-sampler references, GeoSPRINT also improves on DDIM and continues to approach the reference as the budget grows. The trajectory projection score α_{traj}, which drives the curvature analysis, additionally serves as a training free measure of trajectory straightness, exhibiting a $450\times$ drop from DDPM to DDIM. Future directions include learning the blend parameter β per-model, developing predictive per-sample step budgeting for heterogeneous datasets, empirical comparison with the SDM framework [16], and extending the DPM-Solver++ comparison to other higher-resolution models.

Acknowledgments and Disclosure of Funding

I acknowledge support by the NIH funding through grant 1R01DA063157-01, and the High Performance Computing cluster group at The Scripps Research Institute, San Diego, CA, USA.

References

[1] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851, 2020.

[2] 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*, 2021.

[3] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[4] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. In *Advances in Neural Information Processing Systems*, 2022.

[5] Zhifeng Kong and Wei Ping. On fast sampling of diffusion probabilistic models. In *ICML Workshop on Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models*, 2021.

[6] Emiel Hoogeboom, Víctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3D. In *International Conference on Machine Learning*, pages 8867–8887, 2022.

[7] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

[8] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In *Advances in Neural Information Processing Systems*, volume 35, pages 5775–5787, 2022.

[9] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver++: Fast solver for guided sampling of diffusion probabilistic models. *Machine Intelligence Research*, 22(4):730–751, 2025.

[10] 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*, 2023.

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

[12] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning*, pages 32211–32252, 2023.

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

[14] 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*, 2023.

[15] Myunsoo Kim, Donghyeon Ki, Seong-Woong Shim, and Byung-Jun Lee. Adaptive non-uniform timestep sampling for accelerating diffusion model training. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[16] Sangwoo Jo and Sungjoon Choi. Formalizing the sampling design space of diffusion-based generative models via adaptive solvers and Wasserstein-bounded timesteps. *arXiv preprint arXiv:2602.12624*, 2026.

[17] Arpita Joshi and Nurit Haspel. A novel data instance reduction technique using linear feature reduction. *Journal of Artificial Intelligence and Systems*, 2:191–206, 2020.

[18] Magnus Fontes and Charlotte Soneson. The projection score—an evaluation criterion for variable subset selection in PCA visualization. *BMC Bioinformatics*, 12:307, 2011.

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

[20] Fu-Yun Wang et al. Rectified diffusion: Straightness is not your need in rectified flow. *arXiv preprint arXiv:2410.07303*, 2024.

[21] Ling Yang et al. Consistency flow matching: Defining straight flows with velocity consistency. *arXiv preprint arXiv:2407.02398*, 2024.

[22] Amirmojtaba Sabour, Sanja Fidler, and Karsten Kreis. Align your steps: Optimizing sampling schedules in diffusion models. In *International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 42947–42975, 2024.

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

[24] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

[25] Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. On aliased resizing and surprising subtleties in GAN evaluation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11410–11420, 2022.

A Pseudocode for GeoSPRINT

Algorithm 1: GeoSPRINT — Universal Schedule Extraction

Input: Pretrained model $\mathcal{M}$, full schedule $\mathcal{S} = \{t_0, t_1, \ldots, t_T\}$, reference batch size B, window k, target α_{target}, blend β, budget K

Output: Pruned universal schedule $\mathcal{S}^*$

1 **for** $b = 1$ **to** B **do**

2 $\quad \mathcal{Z}^{(b)} \leftarrow \text{SampleFull}(\mathcal{M}, \mathcal{S})$; `// Generate full trajectory`

3 **for** $b = 1$ **to** B **do**

4 $\quad$ Binary search for threshold τ;

5 $\quad\quad (S_{\text{pruned}}, \{r_s\}) \leftarrow \text{HyperplanarityPrune}(\mathcal{Z}^{(b)}, k, \tau)$;

6 $\quad\quad$ Compute $\alpha_{\text{traj}}(S_{\text{pruned}}, \{r_s\}, \mathcal{Z}^{(b)})$ via Eq. 8;

7 $\quad\quad$ Adjust τ until $\alpha_{\text{traj}} \approx \alpha_{\text{target}}$;

8 $\quad \mathcal{R}^{(b)} \leftarrow \{t_i : i \notin S_{\text{pruned}}\}$; `// Map indices to timesteps`

9 $w(t) \leftarrow \frac{1}{B} \sum_b \mathbb{1}[t \in \mathcal{R}^{(b)}]$; `// Curvature density`

10 $\rho_{\text{curv}}(t) \leftarrow \text{SmoothAndFloor}(w(t),\ \sigma{=}5,\ \epsilon_{\text{floor}}{=}0.05 \max w)$;

11 $\rho(t) \leftarrow (1-\beta) \cdot \rho_{\text{logSNR}}(t) + \beta \cdot \rho_{\text{curv}}(t)$; `// Blended density`

12 $F(t) \leftarrow \int_0^t \rho(s)\mathrm{d}s \ / \ \int_0^T \rho(s)\mathrm{d}s$; `// CDF`

13 $\mathcal{S}^* \leftarrow K$ timesteps at uniform quantiles of F;

14 **return** $\mathcal{S}^*$

Key algorithmic details: (1) Algorithm 2 returns residuals $\{r_s\}$ alongside pruned indices, avoiding redundant recomputation for α_{traj}. (2) The QR factorization is computed once at initialization (Line 4) and recomputed only when the window updates (Line 11), reflecting the true amortized cost. (3) Algorithm 1 includes an explicit index-to-timestep mapping and specifies smoothing parameters. (4) Both algorithms use consistent 0-based indexing matching Definition 1.

B Proof of Theorem 1

Proof. Let $S = \{z_{s_1}, \ldots, z_{s_m}\}$ be the pruned set with $m = |S|$. By construction, each z_{s_j} satisfies the hyperplanarity test with residual $r_{s_j} \leq \tau$, meaning its distance from the affine subspace of its retained neighbors is at most τ.

By Definition 2, the trajectory projection score is:

$$\alpha_{\text{traj}} = \frac{\sum_{j=1}^{m} r_{s_j}^2}{\sum_{z \in \mathcal{Z}} \|z - \bar{z}\|^2}. \tag{11}$$

Algorithm 2: HyperplanarityPrune

Input: Ordered points $\mathcal{Z} = \{z_0, z_1, \ldots, z_N\}$, window k, threshold τ
Output: Set of pruned indices S_{pruned}, residuals $\{r_s\}_{s \in S_{\text{pruned}}}$

$S_{\text{pruned}} \leftarrow \emptyset$; residuals $\leftarrow \{\}$;
Initialize window $W \leftarrow (z_0, z_1, \ldots, z_{k-1})$; `// First k points`
$M_W \leftarrow [w_2 - w_1 \mid \cdots \mid w_k - w_1]$;
$Q_W, R_W \leftarrow \text{QR}(M_W)$; `// Thin QR; cached until window updates`
for $i = k$ **to** N **do**
    $r_i \leftarrow \|(I - Q_W Q_W^\top)(z_i - w_1)\|_2$;
    if $r_i < \tau$ **then**
        $S_{\text{pruned}} \leftarrow S_{\text{pruned}} \cup \{i\}$; ;
        residuals$[i] \leftarrow r_i$; `// Redundant; store residual`
    else
        Update W: slide to include z_i;
        Recompute M_W and $Q_W, R_W \leftarrow \text{QR}(M_W)$; `// Update cached QR`
return S_{pruned}, *residuals*

Since each $r_{s_j} \leq \tau$, the numerator is bounded:

$$\sum_{j=1}^{m} r_{s_j}^2 \leq m \cdot \tau^2 = |S| \cdot \tau^2. \tag{12}$$

The denominator equals $\text{tr}(C_{\mathcal{Z}}) \cdot (N{+}1)$, where $C_{\mathcal{Z}} = \frac{1}{N+1} \sum_{z \in \mathcal{Z}} (z - \bar{z})(z - \bar{z})^\top$. Therefore:

$$\alpha_{\text{traj}} \leq \frac{|S| \cdot \tau^2}{(N{+}1) \cdot \text{tr}(C_{\mathcal{Z}})}. \tag{13}$$

□

Interpretation. The bound controls the fraction of total trajectory variance attributable to orthogonal reconstruction residuals of pruned points. It does *not* bound $\text{tr}(C_R)/\text{tr}(C_{\mathcal{Z}})$ (the ratio of retained-set variance to full-set variance), because pruning points changes the sample mean and can reduce the retained sample covariance even when all residuals are zero. For example, removing interior points from a collinear set with $r_i = 0$ for all i yields $\alpha_{\text{traj}} = 0$ but $\text{tr}(C_R) < \text{tr}(C_{\mathcal{Z}})$ whenever the retained points have smaller spread than the full set.

C Extended Experimental Details

C.1 Reference Set Convergence

See figures 5 and 6

C.2 Implementation Details

All experiments use the HuggingFace Diffusers library. For non-uniform timestep schedules, we implement the DDIM update formula manually (Eq. 2) with correct $\alpha_{t_{\text{prev}}}$ lookup, since the library's `DDIMScheduler.step()` assumes uniform spacing internally. DPM-Solver++ experiments use `solver_order=2` with `lower_order_final=True` and a fresh scheduler instance per batch to reset internal state. Trajectory normalization (zero mean, unit variance per dimension) is applied before curvature analysis to ensure the hyperplanarity test measures directional changes rather than raw magnitude.

C.3 Hyperparameter Sensitivity

Blend weight. On CIFAR-10 the optimum is broad, $\beta \in [0.5, 0.7]$ (Table 2). Table 3 sweeps β on SD v1.5 with 2,000 samples per setting (these FIDs are comparable within the sweep but not with the

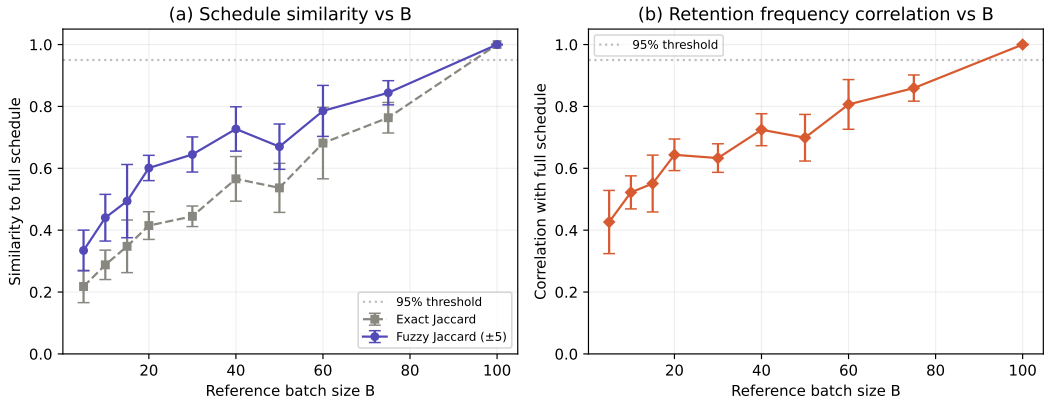

Figure 5: Schedule convergence vs. reference batch size B. (a) Exact and fuzzy Jaccard similarity to the B=100 schedule. Fuzzy matching (within ± 5 timesteps) reaches 0.84 at B=75. (b) Retention frequency correlation reaches 0.86 at B=75.

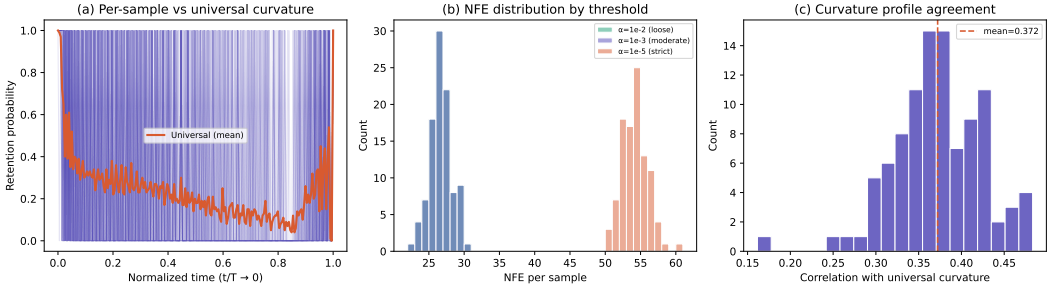

Figure 6: Per-sample geometric complexity on CIFAR-10. (a) Individual curvature profiles (faint) vs. universal mean (bold). Despite diverse per-sample preferences, the universal profile captures systematic structure. (b) NFE distribution at three α_{traj} targets. (c) Per-sample correlation with the universal curvature profile (mean 0.37).

5,000-sample values in Table 1). FID changes by at most 0.53 across the range, with a slight drift toward larger β, so β=0.6 is used unchanged.

Table 3: β-sweep on SD v1.5 (FID ↓ vs. the 50-step DDIM reference; 2,000 samples, single runs).

NFE	β=0.3	0.5	0.6	0.7	0.9
20	11.10	10.86	10.76	10.66	10.57
34	9.11	9.06	9.04	9.02	9.02
44	8.87	8.84	8.84	8.86	8.85

Table 4: Window-size ablation on the 100 CIFAR-10 reference trajectories.

k	kept	α_{traj}	corr. w/ k=2
2	48.0	1.7×10^{-5}	1.00
3	49.2	1.1×10^{-6}	0.75
5	50.4	5.0×10^{-7}	0.66

Window size. Table 4 repeats the curvature-profile computation with $k \in \{2, 3, 5\}$. Larger windows make the redundancy test stricter (lower α_{traj}, slightly more retained steps), but the curvature profile remains positively correlated with the k=2 profile. We did not measure end-to-end FID for $k > 2$ and use k=2, the cheapest setting, throughout.

Reference trajectories and probe retention. We use B=100 (CIFAR-10), B=50 (Church), and B=30 (SD); the CIFAR-10 schedule stabilizes by $B \approx 75$ (Appendix C.1). The probe retention used for the curvature profile is 50 steps for the 200-step pools and 25 for the 50-step SD pool.

C.4 Computational Overhead

Reference-trajectory generation costs $B \times N$ NFEs: $100 \times 200 = 20{,}000$ for CIFAR-10, $50 \times 200 = 10{,}000$ for LSUN Church, and $30 \times 50 = 1{,}500$ for SD v1.5. Curvature analysis (pruning and

retention-frequency computation) runs on CPU in under a minute (27.6 s for the 30 SD reference trajectories), and schedule construction (blending and quantile placement) is effectively instantaneous. The offline cost is incurred once per model and amortized over all subsequent samples; training-based accelerators such as progressive distillation [11] and consistency models [12] instead require on the order of 10^5–10^6 gradient updates.

Hardware. Experiments ran on the High Performance Computing cluster at The Scripps Research Institute. The LSUN Church and SD v1.5 experiments used NVIDIA RTX A6000 (48 GB) GPUs; for example, generating 10,000 LSUN Church samples for one schedule took about 4.7 h on a single A6000 at batch size 32. SD v1.5 samples were generated one at a time (batch size 1) with classifier-free guidance.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction (Section 1) clearly state five contributions: generalization of geometric redundancy testing to high dimensions, a training-free adaptive sampling framework, the trajectory projection score, sample-adaptive scheduling, and comprehensive benchmarking. Each is developed in subsequent sections.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The "Limitations" paragraph in Section 6 discusses: reference pass overhead, β tuning on CIFAR-10, CFG scale generalization, the lack of improvement from higher-level tests ($k > 2$), and complementarity with higher-order solvers.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Theorem 1 (reconstruction error bound) is stated with full assumptions and proved in Appendix B. Proposition 1 is proved in the main text. A remark clarifies the distinction between reconstruction error and retained-set variance.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 5 provides full experimental details including datasets, pretrained models, hyperparameter settings (including smoothing parameters σ=5 and ϵ_{floor}), reference batch sizes per model, sample counts, prompt sources for SD v1.5, evaluation metrics, and algorithmic pseudocode (Appendix A). All pretrained models are publicly available.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: All datasets (CIFAR-10, LSUN Church) and pretrained models (including Stable Diffusion v1.5) are publicly available under their respective licenses. The code is available in its GitHub repository

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 5 and Appendix C provide all experimental details including threshold selection procedures, reference batch sizes per model (B=100 for CIFAR-10, B=50 for Church and SD v1.5), smoothing parameters, and evaluation protocols.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Table 1 reports FID as mean $\pm$ standard deviation over 3 independent runs with different random seeds. Figures 1 and 2 include ± 1 std error bands.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix C (Computational Overhead and Hardware) provides GPU type (NVIDIA A100 80 GB), CPU type (Intel Xeon Gold 6248R), peak memory usage per model, and wall-clock times for all experiments including generation, reference trajectories, and curvature analysis.

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This work proposes a training-free acceleration method for existing generative models and does not introduce new ethical concerns beyond those inherent to the underlying models.

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The "Broader Impact" paragraph in Section 6 discusses positive impacts (reduced computational cost, accessibility) and notes that as an acceleration method, it inherits but does not amplify the societal risks of the underlying generative models.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: GeoSPRINT is a sampling schedule optimization method that does not release new generative models or datasets.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All pretrained models and datasets are cited. CIFAR-10 and LSUN Church are used under standard research licenses. Stable Diffusion v1.5 is used under the CreativeML Open RAIL-M license.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The GeoSPRINT codebase will be released with documentation, including usage instructions and example scripts.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: This work does not involve crowdsourcing or human subjects.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: This work does not involve human subjects research.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research?

Answer: [N/A]

Justification: LLMs are not a component of the core methodology.