On the Trajectory Regularity of ODE-based Diffusion Sampling

Defang Chen [1 2] **Zhenyu Zhou** [1 2] **Can Wang** [1 2] **Chunhua Shen** [3] **Siwei Lyu** [4]

Abstract

Diffusion-based generative models use stochastic differential equations (SDEs) and their equivalent ordinary differential equations (ODEs) to establish a smooth connection between a complex data distribution and a tractable prior distribution. In this paper, we identify several intriguing trajectory properties in the ODE-based sampling process of diffusion models. We characterize an implicit denoising trajectory and discuss its vital role in forming the coupled sampling trajectory with a strong shape regularity, regardless of the generated content. We also describe a dynamic programming-based scheme to make the time schedule in sampling better fit the underlying trajectory structure. This simple strategy requires minimal modification to any given ODE-based numerical solvers and incurs negligible computational cost, while delivering superior performance in image generation, especially in $5 \sim 10$ function evaluations.

1. Introduction

Diffusion-based generative models (Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Ho et al., 2020; Song et al., 2021c) have gained significant attention and achieved remarkable results in image (Dhariwal & Nichol, 2021; Rombach et al., 2022), audio (Kong et al., 2021), video (Ho et al., 2022; Blattmann et al., 2023), and notably in text-to-image synthesis (Saharia et al., 2022; Ruiz et al., 2023; Podell et al., 2024; Esser et al., 2024). These models introduce noise into data through a *forward process* and subsequently generate data by sampling via a *backward process*. Both processes are characterized and modeled using stochastic differential equations (SDEs) (Song et al., 2021c). In diffusion-based generative models, the pivotal element is the score function, defined as the gradient of the log data density *w.r.t.* the input (Hyvärinen, 2005; Lyu, 2009; Raphan & Simoncelli, 2011; Vincent, 2011), irrespective of the model's specific configurations. Training such a model involves learning the score function, which is achievable by developing a noise-dependent denoising model. This model is trained to minimize the mean square error in data reconstruction for the data-noise pairings generated during the forward process (Kingma et al., 2021; Karras et al., 2022). To generate data, diffusion-based generative models solve the acquired score-based backward SDE through a numerical solver. Recent research has shown that the backward SDE can be effectively replaced by an equivalent probability flow ordinary differential equation (PF-ODE), preserving identical marginal distributions (Song et al., 2021c;a; Lu et al., 2022a; Zhang & Chen, 2023; Zhou et al., 2023). This deterministic ODE-based generation reduces the need for stochastic sampling to just the randomness in the initial sample selection, thereby simplifying and granting more control over the entire generative process (Song et al., 2021a;c). Under the PF-ODE formulation, starting from white Gaussian noise, the *sampling trajectory* is formed by running a numerical solver with discretized time steps. These steps collectively constitute the *time schedule* used in sampling.

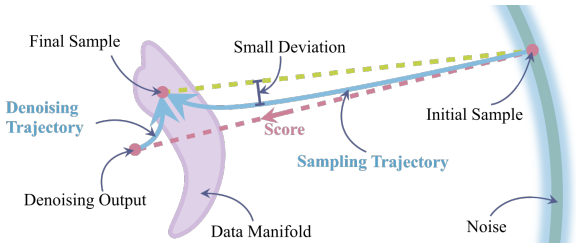


Figure 1. A geometric picture of ODE-based sampling in diffusion models. Each initial sample (from the noise distribution) starts from a big sphere and converges to the final sample (in the data manifold) along a regular sampling trajectory, which is controlled by an implicit denoising trajectory.

Despite the impressive generative capabilities exhibited by diffusion-based models, many mathematical and statistical aspects of these models remain veiled in mystery, largely due to the complex nature of SDEs and neural network models, and the substantial data dimensions involved. In particular, empirical studies indicate an intriguing regularity in the sampling trajectories of PF-ODE based diffusion

[1]State Key Laboratory of Blockchain and Data Security, Zhejiang University, China [2]Hangzhou High-Tech Zone (Binjiang) Blockchain and Data Security Research Institute, China [3]Zhejiang University, China [4]University at Buffalo, USA. Correspondence to: Can Wang <wcan@zju.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

models (Chen et al., 2023a), *i.e.*, the tendency of sample paths to exhibit a "boomerang" shape, or specifically, a linear-nonlinear-linear structure as depicted in Figure 1. In addition, we observe that each sampling trajectory barely strays from the straight line joining its beginning and end points, a deviation that can be effectively approximated using two or three orthogonal bases (Section 3). This pattern appears consistently in different trajectories, irrespective of their initial random samples or the corresponding real data samples (see Figure 4). This simple structure guarantees the common use of a shared time schedule for synthesizing different samples, and enable us to safely adopt large sampling steps without incurring much truncation error (Song et al., 2021a; Karras et al., 2022; Lu et al., 2022a), especially at the first step (Dockhorn et al., 2022; Zhou et al., 2023).

We hypothesize that this regularity reflects some underlying geometric structures of the sampling trajectories. This work aims to elucidate this phenomenon. We start by simplifying the ODE-based sampling trajectory, which reveals an implicit *denoising trajectory*. The denoising trajectory corresponds to a rotation of each point on the sampling trajectory and thus determines the curvature of the sampling trajectory (Section 4.1). Built upon this insight, we show that the denoising trajectory affords a closed-form solution when we use a kernel density estimation (KDE) of varying widths to approximate the original data distribution from training samples. This is analogous to the classical mean-shift algorithm (Fukunaga & Hostetler, 1975; Cheng, 1995; Comaniciu & Meer, 2002), albeit an important difference is that we use time-varying width in KDE (Section 4.2). Though not feasible for the practical solution of the sampling trajectories, the KDE-based solution converges to the optimal solution based on the real data distribution asymptotically. Its closed form lends itself to theoretical analysis. We show that the linear-nonlinear-linear structure follows naturally from this interpretation of the PF-ODE. This trajectory regularity unifies prior observations and clarifies many existing heuristics to expedite diffusion model sampling. Using the shape regularity of the sampling trajectories, we introduce an efficient and effective accelerated sampling approach based on dynamic programming to determine the optimal time schedule. Experimental results demonstrate that trajectory regularity-based accelerated sampling can significantly improve the performance of diffusion-based generative models in a few (≤ 10) function evaluations. Our main contributions are summarized as follows[1]:

- We describe and demonstrate a strong shape regularity of trajectories of ODE-based diffusion sampling, *i.e.*, the sampling trajectories approximately follow a similar shape with a linear-nonlinear-linear structure.
- We explain this regularity through the closed form of the denoising trajectory under a KDE-based data modeling with the time-varying bandwidth.
- We develop a dynamic programming-based approach that leverages the trajectory regularity to find the optimal time schedule of the sampling steps. It introduces minimal overhead and yields improved image quality.

[1]The unpublished, early manuscript of this paper can be found in arXiv (Chen et al., 2023a).

2. Preliminaries

For successful generative modeling, it is essential to connect the data distribution p_d with a non-informative, manageable distribution p_n. Diffusion models fulfill this objective by incrementally introducing white Gaussian noise into the data, effectively obliterating its structures, and subsequently reconstructing the synthesized data from noise samples via a series of denoising steps. The forward step can be modeled as a diffusion process $\{\mathbf{x}_t\}_{t=0}^T$ starting from $\mathbf{x}_0 \sim p_d$, which is the solution of a linear Itô stochastic differential equation (SDE) (Song et al., 2021c; Karras et al., 2022)

$$\mathrm{d}\mathbf{x}_t = f(t)\mathbf{x}_t \mathrm{d}t + g(t)\mathrm{d}\mathbf{w}_t, \tag{1}$$

where $\mathbf{w}_t$ denotes the Wiener process; $f(t)\mathbf{x}_t$ and $g(t)$ are the drift and diffusion coefficients, respectively. The marginal distribution $p_t(\mathbf{x}_t)$ evolves according to the well-known Fokker-Planck equation given the initial condition $p_0(\mathbf{x}_0) = p_d(\mathbf{x}_0)$ (Oksendal, 2013). By properly setting the drift and diffusion coefficients, the data distribution is smoothly transformed to the (approximate) noise distribution $p_T(\mathbf{x}_T) \approx p_n$ in a forward manner. The transition kernel in this context is always a Gaussian distribution, *i.e.*, $p_{0t}(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; s(t)\mathbf{x}_0, s^2(t)\sigma^2(t)\mathbf{I})$, where $s(t) = \exp\left(\int_0^t f(\xi)\mathrm{d}\xi\right)$, $\sigma(t) = \sqrt{\int_0^t \left[g(\xi)/s(\xi)\right]^2 \mathrm{d}\xi}$, and we denote them as s_t, σ_t hereafter for notation simplicity. The signal-to-noise ratio (SNR) is defined as $1/\sigma_t^2$ (Karras et al., 2022). More details are provided in Section A.1.

In the literature, two forms of SDEs are commonly used, namely, the variance-preserving (VP) SDE and the variance-exploding (VE) SDE (Song et al., 2021c), and they are both widely used in large-scale generative models (Ramesh et al., 2022; Rombach et al., 2022; Balaji et al., 2022; Yuan et al., 2023). Our analysis will be based on VE-SDEs and the results can be easily extended to VP-SDEs. In fact, we can safely remove the drift term in (1) without changing the essential characteristics of the underlying diffusion model.

Remark 2.1 (Proofs in Section A.1). Linear diffusion processes sharing the same SNR of transition kernels are equivalent according to Itô's lemma (Oksendal, 2013).

Because of this, we only consider a standardized VE-SDE

$$\mathrm{d}\mathbf{x}_t = \sqrt{\mathrm{d}\sigma_t^2/\mathrm{d}t}\,\mathrm{d}\mathbf{w}_t, \quad \sigma_t : \mathbb{R} \to \mathbb{R}, \tag{2}$$

where σ_t is a pre-defined increasing noise schedule.

The reverse of the forward SDE, as expressed in (2), is represented by another SDE which facilitates the synthesis of data from noise through a backward sampling (Feller, 1949; Anderson, 1982). Notably, a probability flow ordinary differential equation (PF-ODE) exists and maintains the same marginal distributions $\{p_t(\mathbf{x}_t)\}_{t=0}^T$ as the SDE at each time step throughout the diffusion process (Song et al., 2021c):

$$\mathrm{d}\mathbf{x}_t = -\sigma_t \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) \mathrm{d}\sigma_t. \tag{3}$$

The deterministic nature of ODE offers several benefits in generative modeling, including efficient sampling, unique encoding, and meaningful latent manipulations (Song et al., 2021c;a). We thus choose this formula to analyze the sampling behavior of diffusion models throughout this paper.

Training. Simulating the above PF-ODE (3) requests having the score function $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$ (Hyvärinen, 2005; Lyu, 2009), which is typically estimated with *denoising score matching* (Vincent, 2011). Thanks to a profound connection between the score function and the posterior expectation, *i.e.*, $\mathbb{E}(\mathbf{x}_0|\mathbf{x}_t) = \mathbf{x}_t + \sigma_t^2 \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$ (Robbins, 1956; Efron, 2011; Raphan & Simoncelli, 2011), we can also train a *denoising autoencoder* (DAE) (Vincent et al., 2008; Bengio et al., 2013), denoted as $r_{\boldsymbol{\theta}}$, to estimate the conditional expectation $\mathbb{E}(\mathbf{x}_0|\mathbf{x}_t)$, and then convert it to the score function. The objective function of training such a neural network across different noise levels with the weighting function $\lambda(t)$ is $\mathcal{L}_{\mathrm{DAE}}(\boldsymbol{\theta}; \lambda(t)) :=$

$$\int_0^T \lambda(t) \mathbb{E}_{\mathbf{x}_0 \sim p_d} \mathbb{E}_{\mathbf{x}_t \sim p_{0t}(\mathbf{x}_t|\mathbf{x}_0)} \|r_{\boldsymbol{\theta}}(\mathbf{x}_t; \sigma_t) - \mathbf{x}_0\|_2^2 \mathrm{d}t. \tag{4}$$

The optimal estimator $r^\star_{\boldsymbol{\theta}}(\mathbf{x}_t; \sigma_t)$, also known as Bayesian least squares estimator, equals $\mathbb{E}(\mathbf{x}_0|\mathbf{x}_t)$. We thus have $r^\star_{\boldsymbol{\theta}}(\mathbf{x}_t; \sigma_t) = \mathbf{x}_t + \sigma_t^2 \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$. In practice, it is assumed that this connection approximately holds for a converged model ($r_{\boldsymbol{\theta}}(\mathbf{x}_t; \sigma_t) \approx r^\star_{\boldsymbol{\theta}}(\mathbf{x}_t; \sigma_t)$[2]), and we can plug it into (3) to derive the *empirical* PF-ODE as follows

$$\mathrm{d}\mathbf{x}_t = \frac{\mathbf{x}_t - r_{\boldsymbol{\theta}}(\mathbf{x}_t; \sigma_t)}{\sigma_t} \mathrm{d}\sigma_t = \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t; \sigma_t) \mathrm{d}\sigma_t. \tag{5}$$

The noise-prediction model $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\cdot, \cdot)$ introduced above are used in some previous works (Ho et al., 2020; Song et al., 2021a; Nichol & Dhariwal, 2021; Bao et al., 2022).

Sampling. Given the empirical PF-ODE (5), we can synthesize novel samples by first drawing pure noises $\hat{\mathbf{x}}_{t_N} \sim p_n$ as the initial condition, and then numerically solving this equation backward with N steps to obtain a sequence $\{\hat{\mathbf{x}}_{t_n}\}_{n=0}^N$ with a certain time schedule $\Gamma = \{t_0 = \epsilon, \cdots, t_N = T\}$. We adopt the notation $\hat{\mathbf{x}}_{t_n}$ to denote the generated sample by numerical methods, which differs from the exact solutions denoted as $\mathbf{x}_{t_n}$. The final sample $\hat{\mathbf{x}}_{t_0}$ is considered to approximately follow the data distribution p_d. We designate this sequence as a *sampling trajectory*. In practice, there exists various sampling strategies inspired from the classic numerical methods to solve the backward PF-ODE (5), including Euler (Song et al., 2021a), Heun's (Karras et al., 2022), Runge-Kutta (Song et al., 2021c; Liu et al., 2022; Lu et al., 2022a), and linear multistep methods (Liu et al., 2022; Lu et al., 2022b; Zhang & Chen, 2023; Zhao et al., 2023).

[2]We slightly abuse the notation and still denote the converged model as $r_{\boldsymbol{\theta}}(\cdot; \cdot)$ hereafter.

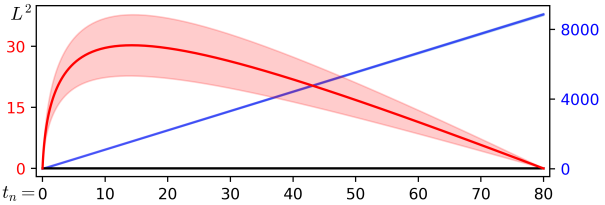


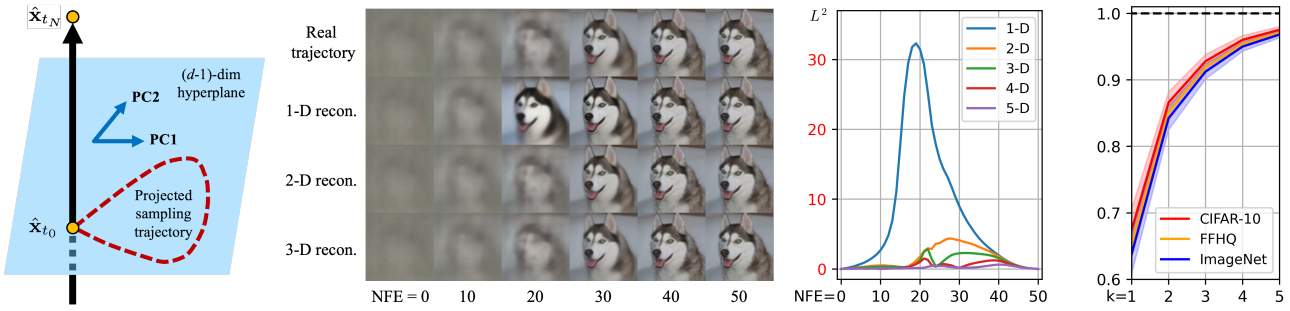
Figure 2. The sampling trajectory exhibits a very small trajectory deviation (red curve) compared to the sample distance (blue curve) in the sampling process starting from $t_N = 80$ to $t_0 = 0.002$.

3. Regularity of PF-ODE Sampling Trajectory

As mentioned in Section 1, the sampling trajectories within a PF-ODE framework of the diffusion model exhibit a certain regularity in their shapes, regardless of the specific content generated. To better demonstrate this concept, we undertake a series of empirical studies.

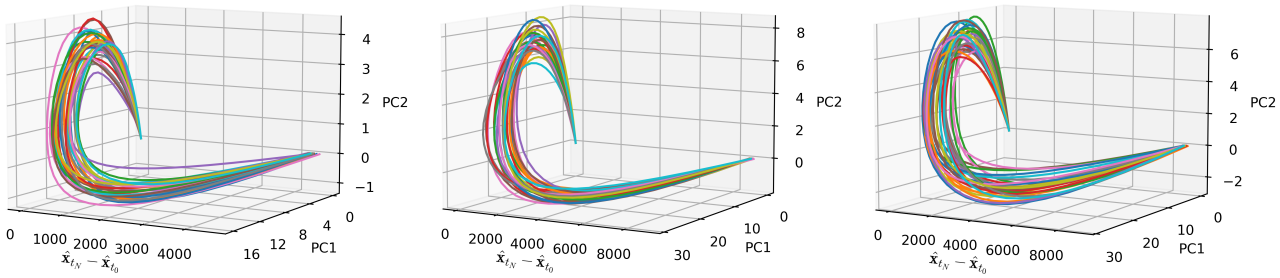
1-D Projections. Visualizing the entire sampling trajectory and analyzing its geometric characteristics in the original high-dimensional space is intractable. To address this, we first examine the *trajectory deviation* from the direct line connecting the two endpoints, which helps assess the trajectory's linearity (see Figure 1). This approach allows us to align and collectively observe the general behaviors of all sampling trajectories. Specifically, we determine the trajectory deviation as the perpendicular L^2 distance from each intermediate sample $\hat{\mathbf{x}}_{t_n}$ to the vector $\hat{\mathbf{x}}_{t_N} - \hat{\mathbf{x}}_{t_0}$, depicted by the red curve with a "boomerang" shape in Figure 2. Additionally, we calculate the L^2 distance between $\hat{\mathbf{x}}_{t_n}$ and the final sample $\hat{\mathbf{x}}_{t_0}$ in the trajectory as $\|\hat{\mathbf{x}}_{t_n} - \hat{\mathbf{x}}_{t_0}\|_2$, depicted by the blue monotone curve in Figure 2.

From Figure 2, we observe that the sampling trajectory's deviation gradually increases from $t = 80$ to approximately $t = 10$, then swiftly diminishes as it approaches the final samples. This pattern suggests that initially, each sample might be influenced by various modes, experiencing significant impact, but later becomes strongly guided by its specific mode after a certain turning point. This behavior supports the approach of arranging time intervals more densely near the minimum timestamp and sparsely towards the maximum



(a) Trajectory projection. (b) Visual Comparison. (c) L^2 error. (d) PCA ratio.

Figure 3. (a) We adopt d-dimensional vector $\hat{\mathbf{x}}_{t_N} - \hat{\mathbf{x}}_{t_0}$ and several top principal components (PCs) on its $(d-1)$-dimensional orthogonal complement to approximate the original d-dimensional sampling trajectory. (b) The visual comparison of trajectory reconstruction on Imagenet 64×64. We reconstruct the real sampling trajectory (top row) using $\hat{\mathbf{x}}_{t_N} - \hat{\mathbf{x}}_{t_0}$ (1-D recon.) along with its top 1 or 2 principal components (2-D or 3-D recon.). To amplify the visual difference, we present the denoising outputs of these trajectories. (c) We calculate the L^2 distance between the real trajectory and the reconstructed trajectories up to 5-D reconstruction. (d) The variance explained by the top k principal components. We report the ratio of the summation of the top k eigenvalues to the summation of all eigenvalues.



(a) CIFAR-10. (b) FFHQ. (c) ImageNet 64 × 64.

Figure 4. We project 30 high-dimensional sampling trajectories generated on three different datasets into 3-D subspace. These trajectories are first aligned to the direction of $\hat{\mathbf{x}}_{t_N} - \hat{\mathbf{x}}_{t_0}$ (this direction is different for each sample), and then projected to the top 2 principal components in the orthogonal space to $\hat{\mathbf{x}}_{t_N} - \hat{\mathbf{x}}_{t_0}$. See texts for more details.

one (Song et al., 2021a; Lu et al., 2022a; Karras et al., 2022; Song et al., 2023). However, when we consider the ratio of the maximum deviation to the endpoint distance in Figure 2, we find that the trajectory deviation is remarkably slight (approximately $30/8868 \approx 0.0034$), indicating a pronounced straightness. Additionally, the generated samples along the sampling trajectory tend to move monotonically from their initial points toward their final points.

The trajectory deviation mathematically equals the reconstruction error if we project all d-dimensional points of the sampling trajectory onto the vector $\hat{\mathbf{x}}_{t_N} - \hat{\mathbf{x}}_{t_0}$. As demonstrated in Figure 3b-3c, the 1-D approximation proves inadequate, leading to a significant deviation from the actual trajectory. These observations imply that while all sampling trajectories share a similar macro-structure, 1-D projection cannot accurately capture such trajectory details.

Multi-D Projections. Moreover, we implement *principal component analysis* (PCA) on the orthogonal complement to the vector $\hat{\mathbf{x}}_{t_N} - \hat{\mathbf{x}}_{t_0}$, which assists in assessing the trajectory's rotational properties. As illustrated in Figure 3a, we begin by projecting each d-D sampling trajectory into its $(d-1)$-D orthogonal space relative to the associated $\hat{\mathbf{x}}_{t_N} - \hat{\mathbf{x}}_{t_0}$ vector, followed by conducting PCA. Figure 3b-3c show that the 2-D approximation using $\hat{\mathbf{x}}_{t_N} - \hat{\mathbf{x}}_{t_0}$ and the first principal component markedly narrows the visual discrepancy with the real trajectory, thereby reducing the L^2 reconstruction error. This finding suggests that all points in each d-D sampling trajectory diverge slightly from a 2-D plane. Consequently, the tangent and normal vectors of the trajectory can be effectively characterized in this manner.

By incorporating an additional principal component, we enhance our ability to capture the torsion of the sampling trajectory, thereby increasing the total explained variance to approximately 85% (see Figure 3d). This improvement allows for a more accurate approximation of the actual trajectory and further reduces the L^2 reconstruction error (see Figure 3b-3c). In practical terms, this level of approximation effectively captures all the visually pertinent information, with the deviation from the real trajectory being nearly in-

distinguishable. Consequently, we can confidently utilize a 3-D subspace, formed by two principal components and the vector $\hat{\mathbf{x}}_{t_N} - \hat{\mathbf{x}}_{t_0}$, to understand the geometric structure of high-dimensional sampling trajectories.

Expanding on this understanding, we present a visualization of 30 randomly chosen sampling trajectories created by a diffusion model trained on CIFAR-10 (Krizhevsky & Hinton, 2009), FFHQ (Karras et al., 2019), or ImageNet 64×64 (Russakovsky et al., 2015) in Figure 4. It is important to note that the scale along the axis of $\hat{\mathbf{x}}_{t_N} - \hat{\mathbf{x}}_{t_0}$ is significantly larger than that of the other two principal components by orders of magnitude. As a result, the trajectory remains very close to the straight line connecting its endpoints, corroborating our findings from the trajectory deviation experiment (see Figure 2). Furthermore, Figure 4 accurately depicts the sampling trajectory's behavior, showing its gradual departure from the osculating plane during the sampling process. Interestingly, each trajectory displays a simple linear-nonlinear-linear structure and shares a similar shape. This consistency reveals a strong regularity in all sampling trajectories with different initial starting points, independent of the specific content generated.

4. Understanding the Trajectory Regularity

We next attempt to explain the trajectory regularity observed in the previous section. We first show that there exists an implicit denoising trajectory, which controls the rotation of the sampling trajectory and determines the subsequent points in a convex combination way (Section 4.1). We then establish a connection between the sampling trajectory and KDE approximation of data distribution (Section 4.2), which is the linchpin to understanding the observed regularity.

4.1. Implicit Denoising Trajectory

Given a parametric diffusion model with the *denoising output* $r_{\boldsymbol{\theta}}(\cdot;\cdot)$, the sampling trajectory is simulated by numerically solving the empirical PF-ODE (5), and meanwhile, an implicitly coupled sequence $\{r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}, \sigma_{t_n})\}_{n=1}^N$ is formed as a by-product. We designate this sequence, or simplified to $\{r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})\}_{n=1}^N$ if there is no ambiguity, as a *denoising trajectory*. It follows a PF-ODE $\mathrm{d}r_{\boldsymbol{\theta}}(\mathbf{x}_t;\sigma_t)/\mathrm{d}\sigma_t = -\sigma_t \left[\mathrm{d}^2\mathbf{x}_t/\mathrm{d}\sigma_t^2\right]$ and actually encapsulates the curvature information of the associated sampling trajectory. The following proposition reveals how these two trajectories are inherently related.

Proposition 4.1. *Given the probability flow ODE (5) and a current position $\hat{\mathbf{x}}_{t_{n+1}}$, $n \in [0, N-1]$ in the sampling trajectory, the next position $\hat{\mathbf{x}}_{t_n}$ predicted by a k-order Taylor expansion with the time step size $\sigma_{t_{n+1}} - \sigma_{t_n}$ equals*

$$\hat{\mathbf{x}}_{t_n} = \frac{\sigma_{t_n}}{\sigma_{t_{n+1}}}\hat{\mathbf{x}}_{t_{n+1}} + \frac{\sigma_{t_{n+1}} - \sigma_{t_n}}{\sigma_{t_{n+1}}}\mathcal{R}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}), \tag{6}$$

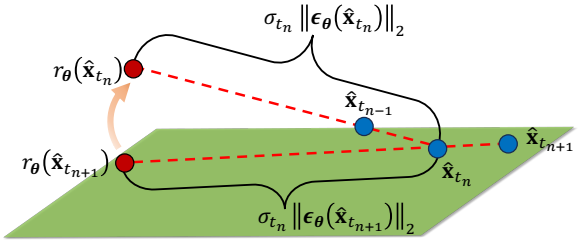


Figure 5. An illustration of two consecutive Euler steps, starting from a current sample $\hat{\mathbf{x}}_{t_{n+1}}$. A single Euler step in the ODE-based sampling is a convex combination of the denoising output and the current position to determine the next position. Blue points form a piecewise linear sampling trajectory, while red points form the denoising trajectory governing the rotation direction.

which is a convex combination of $\hat{\mathbf{x}}_{t_{n+1}}$ and the generalized denoising output $\mathcal{R}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) =$

$$r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - \sum_{i=2}^{k} \frac{1}{i!} \frac{\mathrm{d}^{(i)}\mathbf{x}_t}{\mathrm{d}\sigma_t^{(i)}}\bigg|_{\hat{\mathbf{x}}_{t_{n+1}}} \sigma_{t_{n+1}}(\sigma_{t_n} - \sigma_{t_{n+1}})^{i-1}. \tag{7}$$

We have $\mathcal{R}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) = r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})$ for Euler method ($k = 1$), and $\mathcal{R}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) = r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) + \frac{\sigma_{t_n} - \sigma_{t_{n+1}}}{2} \frac{\mathrm{d}r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{\mathrm{d}\sigma_t}$ for second-order numerical methods ($k = 2$).

Corollary 4.2. *The denoising output $r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})$ reflects the prediction made by a single Euler step from $\hat{\mathbf{x}}_{t_{n+1}}$ with the time step size $\sigma_{t_{n+1}}$.*

Corollary 4.3. *Each previously proposed second-order ODE-based accelerated sampling method corresponds to a specific first-order finite difference of $\mathrm{d}r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})/\mathrm{d}\sigma_t$.*

Proofs and more discussions are provided in Section A.2. The ratio $\sigma_{t_n}/\sigma_{t_{n+1}}$ in (6) reflects our inclination to maintain the current position rather than transitioning to the generalized denoising output at t_{n+1}. In this context, different time-schedule functions, such as uniform, quadratic, and polynomial schedules (Song et al., 2021a; Lu et al., 2022a; Karras et al., 2022), essentially represent various weighting functions. We primarily focus on the Euler method to simplify subsequent discussions, though these insights can be readily extended to higher-order methods. The behavior of the trajectory in the continuous scenario is similarly discernible by examining the sampling process with an infinitesimally small Euler step. A graphical representation of two successive Euler steps is depicted in Figure 5.

We further deduce that, approximately, each intermediate point $\hat{\mathbf{x}}_{t_n}$, $n \in [1, N-1]$ in the piecewise linear sampling trajectory is determined by the selected time schedule, given that $\|r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - \hat{\mathbf{x}}_{t_n}\|_2 = (\sigma_{t_n}/\sigma_{t_{n+1}})\|r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - \hat{\mathbf{x}}_{t_{n+1}}\|_2 = \sigma_{t_n}\|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})\|_2 \approx \sigma_{t_n}\|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})\|_2 = \|r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) - \hat{\mathbf{x}}_{t_n}\|_2$. In this scenario, the denoising output $r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})$ appears to be oscillating toward $r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})$ around

$\hat{\mathbf{x}}_{t_n}$, akin to a simple gravity pendulum (Young et al., 1996). The pendulum length effectively shortens by the coefficient $\sigma_{t_n}/\sigma_{t_{n+1}}$ in each sampling step, starting from roughly $\sigma_T\sqrt{d}$. This specific structure exits in all trajectories. Theoretical justification and empirical evidence are detailed in Section A.3.3. Practically, the magnitude of each oscillation is extremely small ($\approx 0°$), and the entire sampling trajectory only marginally deviates from a 2-D plane. Such deviations can be further represented using a few orthogonal bases as discussed in Section 3.

4.2. Theoretical Analysis of the Trajectory Structure

Next we show that using a Gaussian kernel density estimate (KDE) with training data samples, the denoising trajectory has a closed-form solution.

Given a dataset $\mathcal{D} \coloneqq \{\mathbf{y}_i \in \mathbb{R}^d\}_{i\in\mathcal{I}}$ containing $|\mathcal{I}|$ i.i.d. data points drawn from the unknown data distribution p_d, the marginal density at each time of the forward diffusion process (2) becomes a Gaussian KDE with a bandwidth σ_t^2, *i.e.*, $\hat{p}_t(\mathbf{x}_t) = \int p_{0t}(\mathbf{x}_t|\mathbf{y})\hat{p}_d(\mathbf{y}) = \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}\mathcal{N}\left(\mathbf{x}_t;\mathbf{y}_i,\sigma_t^2\mathbf{I}\right)$, where the empirical data distribution $\hat{p}_d(\mathbf{y})$ is a summation of multiple *Dirac delta functions*, *i.e.*, $\hat{p}_d(\mathbf{y}) = \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}\delta(\|\mathbf{y}-\mathbf{y}_i\|)$. In this case, the optimal denoising output of training a denoising autoencoder becomes a convex combination of original data points, $r^\star_{\boldsymbol{\theta}}(\mathbf{x}_t;\sigma_t) =$

$$\sum_i \frac{\exp\left(-\|\mathbf{x}_t-\mathbf{y}_i\|_2^2/2\sigma_t^2\right)}{\sum_j \exp\left(-\|\mathbf{x}_t-\mathbf{y}_j\|_2^2/2\sigma_t^2\right)}\mathbf{y}_i = \sum_i u_i\mathbf{y}_i, \qquad (8)$$

where each weight u_i is calculated based on the time-scaled and normalized L^2 distance between the input $\mathbf{x}_t$ and $\mathbf{y}_i$ belonging to the dataset $\mathcal{D}$, and $\sum_i u_i = 1$. The proof is provided in Appendix A.3.1. The above equation appears to be highly similar to the iterative formula used in mean shift, which is a well-known non-parametric mode-seeking algorithm via iteratively gradient ascent with adaptive step sizes (Fukunaga & Hostetler, 1975; Cheng, 1995; Comaniciu & Meer, 2002; Yamasaki & Tanaka, 2020). In particular, the time-decreasing bandwidth ($\sigma_t \to 0$ as $t \to 0$) in (8) is strongly reminiscent of *annealed mean shift*, or *multi-bandwidth mean shift* (Shen et al., 2005), which was developed as a metaheuristic algorithm to escape local maxima, where classical mean shift is susceptible to stuck, by monotonically decreasing the bandwidth in iterations.

Based on this connection, we characterize the *local behavior* of diffusion sampling, *i.e.*, each sampling trajectory monotonically converges in terms of the sample likelihood, and its coupled denoising trajectory always achieves higher likelihood (see the proof in Appendix A.3.4). We also characterize the *global behavior* of diffusion sampling as a linear-nonlinear-linear mode-seeking path. In the optimal case, the denoising output, or annealed mean vector, starts from a spurious mode (dataset mean), *i.e.*, $r^\star_{\boldsymbol{\theta}}(\mathbf{x}_t;\sigma_t) \approx \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}\mathbf{y}_i$ with a sufficiently large bandwidth σ_t. Meanwhile, the sampling trajectory is initially located in an approximately uni-modal Gaussian distribution with a *linear* score function $\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t) = \left(r^\star_{\boldsymbol{\theta}}(\mathbf{x}_t;\sigma_t)-\mathbf{x}_t\right)/\sigma_t^2 \approx -\mathbf{x}_t/\sigma_t^2$. The approximation is valid as the norm of dataset mean is very close to zero compared with the norm of $\mathbf{x}_t$ due to the concentration of measure (see Lemma A.11). As σ_t monotonically decreases in the sampling process, the mode numbers of $\hat{p}_t(\mathbf{x}_t)$ increase (Silverman, 1981), and the simple distribution surface gradually shifts to the complex multi-modal one. The score function appears *nonlinear* in the medium σ_t stage since multiple data points contribute a non-negligible effect. Finally, the sampling trajectory is attracted by a certain real-data mode with a sufficiently small bandwidth σ_t, and the score function appears *linear* again, *i.e.*, $\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t) \approx (\mathbf{y}_k-\mathbf{x}_t)/\sigma_t^2$, where $\mathbf{y}_k$ denotes the nearest data point to $\mathbf{x}_t$. In this way, the global mode is hopefully sought by the sampling trajectory, as annealed mean shift (Shen et al., 2005). Intriguingly, we can provide a guarantee that the whole sampling trajectory length is around $\sigma_T\sqrt{d}$ (see Section A.3.3). The above analysis implies that the optimal sampling trajectories simply replay the dataset, but in practice, a slight score deviation ensures the generative ability of diffusion models (see Section A.3.5).

5. Geometry-Inspired Time Scheduling

An ODE-based numerical solver such as Euler (Song et al., 2021c) or Runge-Kutta (Liu et al., 2022; Zhang & Chen, 2023) relies on a pre-defined time schedule $\Gamma = \{t_0 = \epsilon, \cdots, t_N = T\}$ in the sampling process. Typically, given the initial time t_N and the final time t_0, the intermediate time steps t_1 to t_{N-1} are determined by heuristic strategies such as uniform, quadratic (Song et al., 2021a), log-SNR (Lu et al., 2022a;b), and polynomial functions (Karras et al., 2022; Song et al., 2023). In fact, the time schedule reflects our prior knowledge of the sampling trajectory shape. Under the constraint of the total *number of score function evaluations* (NFEs), an improved time schedule can reduce the local truncation error in each numerical step, and hopefully minimize the global truncation error. In this way, the sample quality generated by numerical methods could approach that of the exact solutions of the given empirical PF-ODE (5).

Our previous discussions in Section 3 identified each sampling trajectory as a simple low-dimensional "boomerang" curve. We can thus leverage this geometric structure to re-allocate the intermediate timestamps according to the principle that assigning a larger time step size when the trajectory exhibits a relatively small curvature while assigning a smaller time step size when the trajectory exhibits a relatively large curvature. Additionally, we have demonstrated that different trajectories share almost the same shape, which helps us estimate the common structure of the sampling tra-

jectory by using just a few "warmup" samples. We name our approach to achieve the above goal as *geometry-inspired time scheduling* (GITS) and elaborate the details as follows. We will also show that GITS is adaptable, easy to implement, and introduces negligible computing overheads.

The allocation of the intermediate timestamps can be formulated as an integer programming problem and solved by dynamic programming to search for optimal time scheduling (Cormen et al., 2022). We first define a searching space denoted as Γ_g, which is a fine-grained grid including all possible intermediate timestamps. Then, we measure the trajectory curvature by the local truncation errors. Specifically, we define the cost from the current position $\mathbf{x}_{t_i}$ to the next position $\mathbf{x}_{t_j}$ as the difference between an Euler step and the ground-truth prediction, *i.e.*, $c_{t_i \to t_j} := \mathcal{D}(\hat{\mathbf{x}}_{t_i \to t_j}, \mathbf{x}_{t_i \to t_j})$, where t_i and t_j are two intermediate timestamps from Γ_g and $t_i > t_j$. According to the empirical PF-ODE (5), the ground-truth prediction is calculated as $\mathbf{x}_{t_i \to t_j} = \int_{t_i}^{t_j} \mathbf{x}_{t_i} + \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t)\sigma_t' \mathrm{d}t$, and the Euler prediction is calculated as $\hat{\mathbf{x}}_{t_i \to t_j} = \mathbf{x}_{t_i} + (\sigma_{t_j} - \sigma_{t_i})\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_i})\sigma_{t_i}'$. The cost function $\mathcal{D}$ can be defined as the L^2 distance in the original pixel space, *i.e.*, $\mathcal{D}(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2$, or any other user-specified metric. Given all computed pair-wise costs, which form a cost matrix, this becomes a standard minimum-cost path problem and can be solved with dynamic programming. The algorithm and details are provided in Appendix A.4.1.

Still, there also exist many different ways to determine the time schedule (*e.g.*, using a trainable neural network) by leveraging our discovered trajectory regularity.

6. Experiments

We adhere to the setup and experimental designs of the EDM framework (Karras et al., 2022; Song et al., 2023), with $f(t) = 0$, $g(t) = \sqrt{2t}$, and $\sigma_t = t$. Under this parameterization, the forward VE-SDE is expressed as $\mathrm{d}\mathbf{x}_t = \sqrt{2t}\,\mathrm{d}\mathbf{w}_t$, while the corresponding empirical PF-ODE is formulated as $\mathrm{d}\mathbf{x}_t = \frac{\mathbf{x}_t - r_{\boldsymbol{\theta}}(\mathbf{x}_t; t)}{t}\mathrm{d}t$. The temporal domain is segmented using a polynomial function $t_n = (t_0^{1/\rho} + \frac{n}{N}(t_N^{1/\rho} - t_0^{1/\rho}))^\rho$, where $t_0 = 0.002$, $t_N = 80$, $n \in [0, N]$, and $\rho = 7$. In the visual analysis about 1-D projections (Figure 2) and Multi-D projections (Figure 4) detailed in Section 3, we simulate each sampling trajectory employing the Euler method with 100 score function evaluations. The mean and standard deviation for trajectory deviation showcased in Figure 2 are derived from $5,000$ synthesized samples on ImageNet 64×64, and the ratios of explained variance for PCA presented in Figure 3d are based on 1,000 synthesized samples.

We initiate the dynamic programming experiments with 256 "warmup" samples randomly selected from Gaussian noise to create a more refined grid, and then calculate the cost matrix. The number of "warmup" samples is not a critical hyper-parameter, but reducing it generally increases the variance, as shown in the Table 6. Due to subtle differences exist among sampling trajectories (see Figure 4), we recommend utilizing a reasonable number of "warmup" samples to determine the optimal time schedule, such that this time schedule hopefully works well for all the generated samples. The ground-truth predictions are generated by iPNDM (Zhang & Chen, 2023) (employing a fourth-order multistep Runge-Kutta method with a lower-order warming start) using the polynomial time schedule specified in EDM (Karras et al., 2022) with 60 NFEs, resulting in a grid size $|\Gamma_g| = 61$. Additional findings are available in Appendix A.4. The reported results of all com-

Table 1. Sample quality comparison in terms of Fréchet Inception Distance (FID (Heusel et al., 2017), lower is better) on four datasets (resolutions ranging from 32×32 to 256×256). †: Results reported by authors. More results are provided in Table 8.

METHOD	NFE			
	5	6	8	10
CIFAR-10 32×32 (Krizhevsky & Hinton, 2009)				
DDIM (Song et al., 2021a)	49.66	35.62	22.32	15.69
DDIM + GITS (**ours**)	28.05	21.04	13.30	10.37
DPM-Solver-2 (Lu et al., 2022a)	-	60.00	10.30	5.01
DPM-Solver++(3M) (Lu et al., 2022b)	24.97	11.99	4.54	3.00
DEIS-tAB3 (Zhang & Chen, 2023)	14.39	9.40	5.55	4.09
UniPC (Zhao et al., 2023)	23.98	11.14	3.99	2.89
AMED-Solver (Zhou et al., 2023)	-	7.04	5.56	4.14
AMED-Plugin (Zhou et al., 2023)	-	6.67	3.34	**2.48**
iPNDM (Zhang & Chen, 2023)	13.59	7.05	3.69	2.77
iPNDM + GITS (**ours**)	**8.38**	**4.88**	**3.24**	**2.49**
FFHQ 64×64 (Karras et al., 2019)				
DDIM (Song et al., 2021a)	43.93	35.22	24.39	18.37
DDIM + GITS (**ours**)	29.80	23.67	16.60	13.06
DPM-Solver-2 (Lu et al., 2022a)	-	83.17	22.84	9.46
DPM-Solver++(3M) (Lu et al., 2022b)	22.51	13.74	6.04	4.12
DEIS-tAB3 (Zhang & Chen, 2023)	17.36	12.25	7.59	5.56
UniPC (Zhao et al., 2023)	21.40	12.85	5.50	3.84
AMED-Solver (Zhou et al., 2023)	-	10.28	6.90	5.49
AMED-Plugin (Zhou et al., 2023)	-	9.54	5.28	3.66
iPNDM (Zhang & Chen, 2023)	17.17	10.03	5.52	3.98
iPNDM + GITS (**ours**)	**11.22**	**7.00**	**4.52**	**3.62**
ImageNet 64×64 (Russakovsky et al., 2015)				
DDIM (Song et al., 2021a)	43.81	34.03	22.59	16.72
DDIM + GITS (**ours**)	24.92	19.54	13.79	10.83
DPM-Solver-2 (Lu et al., 2022a)	-	44.83	12.42	6.84
DPM-Solver++(3M) (Lu et al., 2022b)	25.49	15.06	7.84	5.67
DEIS-tAB3 (Zhang & Chen, 2023)	14.75	12.57	6.84	5.34
UniPC (Zhao et al., 2023)	24.36	14.30	7.52	5.53
RES(M)† (Zhang et al., 2023)	25.10	14.32	7.44	5.12
AMED-Solver (Zhou et al., 2023)	-	10.63	7.71	6.06
AMED-Plugin (Zhou et al., 2023)	-	12.05	7.03	5.01
iPNDM (Zhang & Chen, 2023)	18.99	12.92	7.20	5.11
iPNDM + GITS (**ours**)	**10.79**	**8.43**	**5.82**	**4.48**
LSUN Bedroom 256×256 (Yu et al., 2015) (pixel-space)				
DDIM (Song et al., 2021a)	34.34	25.25	15.71	11.42
DDIM + GITS (**ours**)	22.04	16.54	11.20	9.04
DPM-Solver-2 (Lu et al., 2022a)	-	80.59	23.26	9.61
DPM-Solver++(3M) (Lu et al., 2022b)	23.15	12.28	7.44	5.71
UniPC (Zhao et al., 2023)	23.34	11.71	7.53	5.75
AMED-Solver (Zhou et al., 2023)	-	12.75	**6.95**	5.38
AMED-Plugin (Zhou et al., 2023)	-	11.58	7.48	5.70
iPNDM (Zhang & Chen, 2023)	26.65	20.73	11.78	5.57
iPNDM + GITS (**ours**)	**15.85**	**10.41**	**7.31**	**5.28**

Table 2. Image generation on Stable Diffusion v1.4 (Rombach et al., 2022) with the default classifier-free guidance scale 7.5 (one sampling step requires two NFEs). We follow the standard FID evaluation, and use the statistics and 30k sampled captions from the MS-COCO (Lin et al., 2014) validation set provided here.

METHOD	NFE			
	10	12	14	16
DPM-Solver++(2M) (Lu et al., 2022b)	17.16	15.76	15.06	14.72
DPM-Solver++(2M) + GITS (**ours**)	**15.53**	**13.29**	**12.44**	**12.26**

pared approaches are obtained from an open-source toolbox: `https://github.com/zju-pi/diff-sampler`.

Image generation. As shown in Tables 1-2, our simple time re-allocation strategy based on iPNDM (Zhang & Chen, 2023) consistently beats all existing ODE-based accelerated sampling methods, with a significant margin especially in the few NFE cases. In particular, all time schedules in these datasets are searched based on the Euler method, *i.e.*, DDIM (Song et al., 2021a), but they are directly applicable with high-order methods such as iPNDM (Zhang & Chen, 2023). The trajectory regularity we uncovered guarantees that the schedule determined through 256 "warmup" samples is effective across all generated content. Furthermore, the experimental outcomes suggest that identifying this trajectory regularity enhances our comprehension of diffusion models' mechanisms. This understanding opens avenues for developing tailored time schedules for more efficient diffusion sampling. Note that we did not use the analytical first step (AFS) that replaces the first numerical step with an analytical Gaussian score to save one NFE, proposed in (Dockhorn et al., 2022) and later used in (Zhou et al., 2023), as we found this trick is particularly effective for datasets containing the small-resolution images. DPM-Solver-2 (Lu et al., 2022a) and AMED-Solver/Plugin (Zhou et al., 2023) are thus inapplicable with NFE$=5$ (marked as "-") in Table 1. Ablation studies on AFS and the grid size for dynamic programming are provided in Appendix A.4.

Time schedule comparison. From Table 3, we can see that compared with existing handcraft time schedules, our used schedule is expected to better fit the underlying trajectory structure in the sampling of diffusion models and achieves smaller truncation errors with improved sample quality.

Running time. Our strategy incurs a very low cost and does not require accessing the real dataset. We start by a few initial "warmup" samples and run the given ODE-solver with fine-grained and coarse-grained steps to compute the cost matrix for dynamic programming. *Such a computation is performed only once on each dataset to simultaneously find optimal time schedules for different NFEs*, thanks to the optimal substructure property (Cormen et al., 2022). This requires less than or about a minute on CIFAR-10, FFHQ, ImageNet 64×64, and 10 to 15 minutes for LSUN Bedroom and LAION (Stable Diffusion), as shown in Table 4.

Table 3. The comparison of FID results on CIFAR-10. Time schedules considerably affect the image generation performance.

TIME SCHEDULE	NFE			
	5	6	8	10
DDIM-uniform	36.98	28.22	19.60	15.45
DDIM-logsnr	53.53	38.20	24.06	16.43
DDIM-polynomial	49.66	35.62	22.32	15.69
DDIM + GITS (**ours**)	**28.05**	**21.04**	**13.30**	**10.09**
iPNDM-uniform	17.34	9.75	7.56	7.35
iPNDM-logsnr	19.87	10.68	4.74	2.94
iPNDM-polynomial	13.59	7.05	3.69	2.77
iPNDM + GITS (**ours**)	**8.38**	**4.88**	**3.24**	**2.49**

Table 4. Used time (seconds) in different stages of GITS, evaluated on an NVIDIA A100 GPU. "warmup" samples are generated by 60 NFE and the NFE budget for dynamic programming is 10.

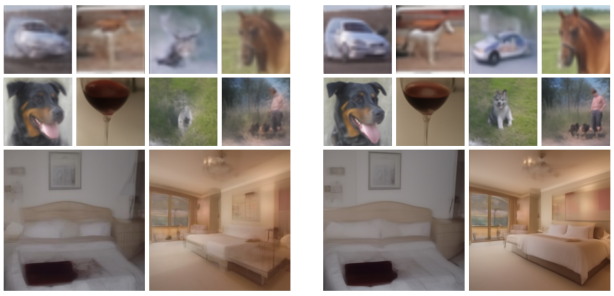
DATASET	sample generation	cost matrix	dynamic programming	total time (s)
CIFAR-10 32×32	27.47	5.29	0.015	32.78
FFHQ 64×64	51.90	10.88	0.016	62.79
ImageNet 64×64	71.77	13.28	0.016	85.07
LSUN Bedroom	517.63	122.13	0.015	639.78
LAION (sd-v1.4)	877.62	24.00	0.016	901.62

7. Related Work and Discussions

The popular variance-exploding (VE) SDEs (Song & Ermon, 2019; Song et al., 2021c) are taken as our main examples for analysis, which are equivalent to their variance-preserving (VP) counterparts according to Itô's lemma (see Remark 2.1 and Appendix A.1). The equivalence has been established in their corresponding PF-ODE (rather than SDE) forms by using the change-of-variable formula (see Proposition 1 of Song et al. (2021a) and Proposition 3 of Zhang & Chen (2023)). Karras et al. (2022) also presented a set of steps to express different specific models in a common framework.

Instead of training a noise-conditional score model with denoising score matching (Vincent, 2011; Song & Ermon, 2019; Song et al., 2021b) or training a noise-prediction model to estimate the added noise in each step (Ho et al., 2020; Song et al., 2021a; Nichol & Dhariwal, 2021; Vahdat et al., 2021; Bao et al., 2022), we follow (Kingma et al., 2021; Karras et al., 2022) and train a denoising model that predicts the reconstructed data from its corrupted version. With the help of simplified empirical PF-ODE (5), we could characterize an implicit denoising trajectory and draw inspiration from classical non-parametric mean shift (Fukunaga & Hostetler, 1975; Cheng, 1995; Comaniciu & Meer, 2002).

Denoising trajectory has been observed since the renaissance of diffusion models (see Figure 6 of (Ho et al., 2020)) and later in Figure 3 of (Kwon et al., 2023), but did not been



(a) DDIM, NFE = 5. (b) DDIM + GITS, NFE = 5.

Figure 6. The visual comparison of samples generated by DDIM and DDIM + GITS (1st row: CIFAR-10, 2nd row: ImageNet 64 × 64, 3rd row: LSUN Bedroom). See more results in Appendix A.4.

investigated, perhaps due to the indirect model parameterization. Karras et al. (2022) first stated that the denoising output reflects the tangent of the sampling trajectory as our Corollary 4.2, but neither characterizes the denoising trajectory in differential equations nor discusses how it controls the sampling trajectory. In fact, Karras et al. (2022) mentioned this property to argue the sampling trajectory of (5) is approximately linear due to the slow change in denoising output, and verified it in a 1D toy example. In contrast, we provide an in-depth analysis of the high-dimensional trajectories with real data and highlight a strong regularity.

We then describe one potential application to accelerate the sampling process. Different from most existing methods focusing on develop better ODE-solvers (Song et al., 2021a; Karras et al., 2022; Liu et al., 2022; Lu et al., 2022a; Zhang & Chen, 2023; Zhao et al., 2023) while selecting a time schedule in a handcraft or trial-and-error way, we leverage trajectory regularity to better allocate the discretized time steps. Our approach is extremely faster than those distillation-based accelerating sampling methods (Luhman & Luhman, 2021; Salimans & Ho, 2022; Zheng et al., 2023; Song et al., 2023) by several orders of magnitude. Besides, Watson et al. (2021) used dynamic programming to optimize the time schedule based on the decomposable nature of ELBO but even worsen the sample quality. There are also many theoretical studies on the convergence analysis, and score estimation of diffusion models, but none of them focus on the trajectory properties (De Bortoli, 2022; Pidstrigach, 2022; Lee et al., 2023; Chen et al., 2023b;c).

Recently, a concurrent work named AYS was proposed to optimize time schedules in sampling by minimizing the mismatch between solving the true backward SDE and its approximated linear counterpart, based on techniques from stochastic calculus (Sabour et al., 2024). In contrast, our GITS leverages the strong trajectory regularity in diffusion models and yields time schedules with dynamic programming using only a few number of "warmup" samples. Our method also gets rid of the time-consuming Monte-Carlo computation in AYS (Sabour et al., 2024) and therefore is faster by several orders of magnitude. In Figure 7, we compare samples generated by different time schedules, using the colab code from `https://research.nvidia.com/labs/toronto-ai/AlignYourSteps/` with the default setting. We also evaluate their FID performance as the procedure used in Table 2, and the results are 14.28 (uniform), 12.48 (AYS), and 12.02 (GITS).

(a) Uniform. (b) AYS. (c) GITS.

Figure 7. The visual comparison of samples generated by Stable Diffusion 1.5 with DPM-Solver++(2M), using the uniform, AYS-optimized (Sabour et al., 2024) or GITS-optimized time schedule and 10 steps. The text prompts are "a photo of an astronaut riding a horse on mars" (1st row); "a whimsical underwater world inhabited by colorful sea creatures and coral reefs" (2nd row); "a digital illustration of the Babel tower 4k detailed trending in artstation fantasy vivid colors" (3rd row).

8. Conclusion

In this paper, we illustrate the trajectory regularity that consistently appears in the ODE-based diffusion sampling, regardless of the specific content generated. We explain this regularity by characterizing and analyzing the implicit denoising trajectory, especially its behavior, under kernel density estimation-based data modeling. Such insights about the trajectory structure of diffusion-based generative models can lead to an accelerated sampling method to improve image synthesis quality with negligible computing overheads.

For future works, we would like to explore deeper shape regularities in the sampling trajectories, describing more precise geometric structures of the regular shapes, and identifying new applications of these insights.

Impact Statement

This work examines the theoretical properties of sampling trajectories in diffusion-based generative models. These models can produce images, audio, and videos indistinguishable from humans, raising concerns about their potential use in disseminating misinformation. We are fully aware of these negative impacts and, in subsequent work, will devote ourselves to developing mitigation measures, including detecting and watermarking synthetic contents.

Acknowledgements

Defang Chen would like to thank Ziying Guo for helpful discussions on the sampling trajectory length and drawing the illustrations (Figures 1, 5 and 10), and thank Yuanzhi Zhu for helpful suggestions on the early manuscript of this work, and thank Haowei Zheng for helpful discussions on the singularities of diffusion models and Remark A.12.

Defang Chen, Zhenyu Zhou and Can Wang are supported by the Starry Night Science Fund of Zhejiang University Shanghai Institute for Advanced Study, China (Grant No: SN-ZJU-SIAS-001), National Natural Science Foundation of China (Grant No: U1866602) and the advanced computing resources provided by the Supercomputing Center of Hangzhou City University.

References

Anderson, B. D. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

Balaji, Y., Nah, S., Huang, X., Vahdat, A., Song, J., Kreis, K., Aittala, M., Aila, T., Laine, S., Catanzaro, B., et al. ediffi: Text-to-image diffusion models with an ensemble of expert denoisers. *arXiv preprint arXiv:2211.01324*, 2022.

Bao, F., Li, C., Zhu, J., and Zhang, B. Analytic-dpm: an analytic estimate of the optimal reverse variance in diffusion probabilistic models. In *International Conference on Learning Representations*, 2022.

Bengio, Y., Yao, L., Alain, G., and Vincent, P. Generalized denoising auto-encoders as generative models. In *Advances in Neural Information Processing Systems*, pp. 899–907, 2013.

Blattmann, A., Rombach, R., Ling, H., Dockhorn, T., Kim, S. W., Fidler, S., and Kreis, K. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22563–22575, 2023.

Carreira-Perpinán, M. A. A review of mean-shift algorithms for clustering. *arXiv preprint arXiv:1503.00687*, 2015.

Chen, D., Zhou, Z., Mei, J.-P., Shen, C., Chen, C., and Wang, C. A geometric perspective on diffusion models. *arXiv preprint arXiv:2305.19947*, 2023a.

Chen, M., Huang, K., Zhao, T., and Wang, M. Score approximation, estimation and distribution recovery of diffusion models on low-dimensional data. In *International Conference on Machine Learning*, pp. 4672–4712, 2023b.

Chen, S., Daras, G., and Dimakis, A. Restoration-degradation beyond linear diffusions: A non-asymptotic analysis for ddim-type samplers. In *International Conference on Machine Learning*, pp. 4462–4484, 2023c.

Cheng, Y. Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):790–799, 1995.

Comaniciu, D. and Meer, P. Mean shift analysis and applications. In *Proceedings of the International Conference on Computer Vision*, pp. 1197–1203, 1999.

Comaniciu, D. and Meer, P. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

Comaniciu, D., Ramesh, V., and Meer, P. Real-time tracking of non-rigid objects using mean shift. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pp. 142–149, 2000.

Comaniciu, D., Ramesh, V., and Meer, P. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–577, 2003.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. *Introduction to algorithms*. MIT press, 2022.

De Bortoli, V. Convergence of denoising diffusion models under the manifold hypothesis. *Transactions on Machine Learning Research*, 2022.

Dhariwal, P. and Nichol, A. Diffusion models beat gans on image synthesis. In *Advances in Neural Information Processing Systems*, pp. 8780–8794, 2021.

Dockhorn, T., Vahdat, A., and Kreis, K. Genie: Higher-order denoising diffusion solvers. In *Advances in Neural Information Processing Systems*, pp. 30150–30166, 2022.

Efron, B. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011.

Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., et al. Scaling rectified flow transformers for high-resolution image synthesis. *arXiv preprint arXiv:2403.03206*, 2024.

Feller, W. On the theory of stochastic processes, with particular reference to applications. In *Proceedings of the First Berkeley Symposium on Mathematical Statistics and Probability*, pp. 403–432, 1949.

Fukunaga, K. and Hostetler, L. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21(1):32–40, 1975.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, pp. 6626–6637, 2017.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, pp. 6840–6851, 2020.

Ho, J., Salimans, T., Gritsenko, A. A., Chan, W., Norouzi, M., and Fleet, D. J. Video diffusion models. In *Advances in Neural Information Processing Systems*, pp. 8633–8646, 2022.

Hyvärinen, A. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6:695–709, 2005.

Karras, T., Laine, S., and Aila, T. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4401–4410, 2019.

Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, pp. 26565–26577, 2022.

Kingma, D. P., Salimans, T., Poole, B., and Ho, J. Variational diffusion models. In *Advances in Neural Information Processing Systems*, pp. 21696–21707, 2021.

Kong, Z., Ping, W., Huang, J., Zhao, K., and Catanzaro, B. Diffwave: A versatile diffusion model for audio synthesis. In *International Conference on Learning Representations*, 2021.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. *Technical Report*, 2009.

Kwon, M., Jeong, J., and Uh, Y. Diffusion models already have a semantic latent space. In *International Conference on Learning Representations*, 2023.

Lee, H., Lu, J., and Tan, Y. Convergence of score-based generative modeling for general data distributions. In *International Conference on Algorithmic Learning Theory*, pp. 946–985, 2023.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *Proceedings of the European Conference on Computer Vision*, pp. 740–755, 2014.

Liu, L., Ren, Y., Lin, Z., and Zhao, Z. Pseudo numerical methods for diffusion models on manifolds. In *International Conference on Learning Representations*, 2022.

Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., and Zhu, J. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. In *Advances in Neural Information Processing Systems*, pp. 5775–5787, 2022a.

Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., and Zhu, J. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022b.

Luhman, E. and Luhman, T. Knowledge distillation in iterative generative models for improved sampling speed. *arXiv preprint arXiv:2101.02388*, 2021.

Lyu, S. Interpretation and generalization of score matching. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pp. 359–366, 2009.

Nichol, A. Q. and Dhariwal, P. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pp. 8162–8171, 2021.

Oksendal, B. *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013.

Pidstrigach, J. Score-based generative models detect manifolds. In *Advances in Neural Information Processing Systems*, pp. 35852–35865, 2022.

Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., and Rombach, R. Sdxl: Improving latent diffusion models for high-resolution image synthesis. In *International Conference on Learning Representations*, 2024.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Raphan, M. and Simoncelli, E. P. Least squares estimation without priors or supervision. *Neural Computation*, 23 (2):374–420, 2011.

Robbins, H. E. An empirical bayes approach to statistics. In *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability*, 1956.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., and Aberman, K. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22500–22510, 2023.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M. S., Berg, A. C., and Li, F. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

Sabour, A., Fidler, S., and Kreis, K. Align your steps: Optimizing sampling schedules in diffusion models. *arXiv preprint arXiv:2404.14507*, 2024.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E. L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al. Photorealistic text-to-image diffusion models with deep language understanding. In *Advances in Neural Information Processing Systems*, pp. 36479–36494, 2022.

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

Särkkä, S. and Solin, A. *Applied stochastic differential equations*, volume 10. Cambridge University Press, 2019.

Shen, C., Brooks, M. J., and van den Hengel, A. Fast global kernel density mode seeking with application to localisation and tracking. In *Proceedings of the International Conference on Computer Vision*, pp. 1516–1523, 2005.

Silverman, B. W. Using kernel density estimates to investigate multimodality. *Journal of the Royal Statistical Society: Series B (Methodological)*, 43(1):97–99, 1981.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pp. 2256–2265, 2015.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, pp. 11895–11907, 2019.

Song, Y. and Ermon, S. Improved techniques for training score-based generative models. In *Advances in Neural Information Processing Systems*, pp. 12438–12448, 2020.

Song, Y., Durkan, C., Murray, I., and Ermon, S. Maximum likelihood training of score-based diffusion models. In *Advances in Neural Information Processing Systems*, pp. 1415–1428, 2021b.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021c.

Song, Y., Dhariwal, P., Chen, M., and Sutskever, I. Consistency models. In *International Conference on Machine learning*, pp. 32211–32252, 2023.

Vahdat, A., Kreis, K., and Kautz, J. Score-based generative modeling in latent space. In *Advances in Neural Information Processing Systems*, pp. 11287–11302, 2021.

Vershynin, R. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.

Vincent, P. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. Extracting and composing robust features with denoising autoencoders. In *International Conference on Machine learning*, pp. 1096–1103, 2008.

Watson, D., Ho, J., Norouzi, M., and Chan, W. Learning to efficiently sample from diffusion probabilistic models. *arXiv preprint arXiv:2106.03802*, 2021.

Whitney, H. Differentiable manifolds. *Annals of Mathematics*, pp. 645–680, 1936.

Xu, Y., Tong, S., and Jaakkola, T. S. Stable target field for reduced variance score estimation in diffusion models. In *International Conference on Learning Representations*, 2023.

Yamasaki, R. and Tanaka, T. Properties of mean shift. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(9):2273–2286, 2020.

Young, H. D., Freedman, R. A., Sandin, T., and Ford, A. L. *University physics*, volume 9. Addison-wesley Reading, MA, 1996.

Yu, F., Seff, A., Zhang, Y., Song, S., Funkhouser, T., and Xiao, J. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

Yuan, Y., Song, J., Iqbal, U., Vahdat, A., and Kautz, J. Physdiff: Physics-guided human motion diffusion model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16010–16021, 2023.

Zhang, Q. and Chen, Y. Fast sampling of diffusion models with exponential integrator. In *International Conference on Learning Representations*, 2023.

Zhang, Q., Song, J., and Chen, Y. Improved order analysis and design of exponential integrator for diffusion models sampling. *arXiv preprint arXiv:2308.02157*, 2023.

Zhao, W., Bai, L., Rao, Y., Zhou, J., and Lu, J. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. In *Advances in Neural Information Processing Systems*, pp. 49842–49869, 2023.

Zheng, H., Nie, W., Vahdat, A., Azizzadenesheli, K., and Anandkumar, A. Fast sampling of diffusion models via operator learning. In *International Conference on Machine Learning*, pp. 42390–42402, 2023.

Zhou, Z., Chen, D., Wang, C., and Chen, C. Fast ode-based sampling for diffusion models in around 5 steps. *arXiv preprint arXiv:2312.00094*, 2023.

A. Appendix

A.1. The Equivalence of Linear Diffusion Models

In this section, we show that various linear diffusion models sharing the same signal-to-noise ratio (SNR) are closely connected with Itô's lemma. In particular, all other model types (*e.g.*, variance-preserving (VP) diffusion process used in DDPMs (Ho et al., 2020)) can be transformed into the variance-exploding (VE) counterparts. Therefore, we merely focus on the mathematical properties and geometric behaviors of VE-SDEs in the main text to simplify our discussions.

A.1.1. General Notations of Linear Diffusion Models

We first recap the basic notations in diffusion models. We consider the data perturbation as a continuous stochastic process $\{\mathbf{z}_t\}_{t=0}^T$ starting from $\mathbf{z}_0 \sim p_d$, which is the solution of a linear stochastic differential equation (SDE) (Song et al., 2021c):

$$\mathrm{d}\mathbf{z}_t = f(t)\mathbf{z}_t\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}_t, \quad f(\cdot):\mathbb{R}\to\mathbb{R}, \quad g(\cdot):\mathbb{R}\to\mathbb{R}, \tag{9}$$

where $\mathbf{w}_t$ denotes the standard Wiener process; $f(\cdot)$ and $g(\cdot)$ are drift and diffusion coefficients, respectively. These two coefficients are required to be globally Lipschitz *w.r.t.* time to ensure the SDE has a unique strong solution (Oksendal, 2013). The probability density function $p_t(\mathbf{z}_t)$, starting from the initial condition $p_0(\mathbf{z}_0) = p_d(\mathbf{z}_0)$, evolves according to the well-known Fokker-Planck equation $\frac{\partial p_t(\mathbf{z}_t)}{\partial t} = -\nabla\cdot\left[p_t(\mathbf{z}_t)f(t)\mathbf{z}_t - \frac{g^2(t)}{2}\nabla_{\mathbf{z}_t}p_t(\mathbf{z}_t)\right]$. In this case, the transition kernel has the following form (Särkkä & Solin, 2019; Karras et al., 2022)

$$p_{0t}(\mathbf{z}_t|\mathbf{z}_0) = \mathcal{N}\left(\mathbf{z}_t; s(t)\mathbf{z}_0, s^2(t)\sigma^2(t)\mathbf{I}\right), \quad s(t) = \exp\left(\int_0^t f(\xi)\mathrm{d}\xi\right), \quad \text{and} \quad \sigma(t) = \sqrt{\int_0^t \left[g(\xi)/s(\xi)\right]^2\mathrm{d}\xi}. \tag{10}$$

We denote $s(t)$ and $\sigma(t)$ as s_t and σ_t respectively hereafter for notation simplicity.

Remark A.1. An important implication from (10) is that different linear diffusion processes sharing the same σ_t actually have the same signal-to-noise ratio (SNR), since SNR is defined as $s_t^2/\left[s_t\sigma_t\right]^2 = 1/\sigma_t^2$.

Therefore, we can equivalently rewrite the forward SDE (9) in terms of s_t and σ_t as follows

$$\mathrm{d}\mathbf{z}_t = \frac{\mathrm{d}\log s_t}{\mathrm{d}t}\mathbf{z}_t\,\mathrm{d}t + s_t\sqrt{\frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t}}\,\mathrm{d}\mathbf{w}_t, \quad f(t) = \frac{\mathrm{d}\log s_t}{\mathrm{d}t}, \quad \text{and} \quad g(t) = s_t\sqrt{\frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t}}. \tag{11}$$

By properly setting the coefficients s_t and σ_t, we demonstrate that the standard notations of the variance-preserving (VP) SDE and the variance-exploding (VE) SDE in the literature (Song et al., 2021c; Karras et al., 2022) can be recovered:

- VP-SDEs (Ho et al., 2020; Nichol & Dhariwal, 2021; Song et al., 2021a;c): By setting $s_t = \sqrt{\alpha_t}$, $\sigma_t = \sqrt{(1-\alpha_t)/\alpha_t}$, $\beta_t = -\mathrm{d}\log\alpha_t/\mathrm{d}t$, and $\alpha_t \in (0,1]$ as a decreasing sequence with $\alpha_0 = 1, \alpha_T \approx 0$, we have

$$\mathbf{z}_t = \sqrt{\alpha_t}\,\mathbf{z}_0 + \sqrt{1-\alpha_t}\,\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0},\mathbf{I}); \qquad \mathrm{d}\mathbf{z}_t = -\frac{1}{2}\beta_t\mathbf{z}_t\,\mathrm{d}t + \sqrt{\beta_t}\mathrm{d}\mathbf{w}_t. \tag{12}$$

- VE-SDEs (Song & Ermon, 2019; 2020; Song et al., 2021c): By setting $s_t = 1$, we have

$$\mathbf{z}_t = \mathbf{z}_0 + \sigma_t\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0},\mathbf{I}); \qquad \mathrm{d}\mathbf{z}_t = \sqrt{\frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t}}\,\mathrm{d}\mathbf{w}_t. \tag{13}$$

Lemma A.2. *Given $\mathbf{z}_0 \sim p_d$ and the transition kernel as (10), we have $\nabla_{\mathbf{z}_t}\log p_t(\mathbf{z}_t) = \left[s_t\sigma_t\right]^{-2}\left(s_t\mathbb{E}(\mathbf{z}_0|\mathbf{z}_t) - \mathbf{z}_t\right)$, or equivalently, $\nabla_{\mathbf{z}_t}\log p_t(\mathbf{z}_t) = -\left[s_t\sigma_t\right]^{-1}\mathbb{E}_{p_{t0}(\mathbf{z}_0|\mathbf{z}_t)}\boldsymbol{\epsilon}_t$, where $\boldsymbol{\epsilon}_t = \left[s_t\sigma_t\right]^{-1}\left(\mathbf{z}_t - s_t\mathbf{z}_0\right)$.*

Proof.

$$\begin{aligned}
p_t(\mathbf{z}_t) = \int p_d(\mathbf{z}_0)p_t(\mathbf{z}_t|\mathbf{z}_0)\mathrm{d}\mathbf{z}_0, \quad \rightarrow \quad \nabla_{\mathbf{z}_t}p_t(\mathbf{z}_t) &= \int \frac{(s_t\mathbf{z}_0 - \mathbf{z}_t)}{s_t^2\sigma_t^2}p_d(\mathbf{z}_0)p_{0t}(\mathbf{z}_t|\mathbf{z}_0)\mathrm{d}\mathbf{z}_0 \\
s_t^2\sigma_t^2\nabla_{\mathbf{z}_t}p_t(\mathbf{z}_t) &= \int s_t\mathbf{z}_0 p_d(\mathbf{z}_0)p_{0t}(\mathbf{z}_t|\mathbf{z}_0)\mathrm{d}\mathbf{z}_0 - \mathbf{z}_t p_t(\mathbf{z}_t) \\
s_t^2\sigma_t^2\frac{\nabla_{\mathbf{z}_t}p_t(\mathbf{z}_t)}{p_t(\mathbf{z}_t)} &= s_t\int \mathbf{z}_0 p_t(\mathbf{z}_0|\mathbf{z}_t)\mathrm{d}\mathbf{z}_0 - \mathbf{z}_t \\
\nabla_{\mathbf{z}_t}\log p_t(\mathbf{z}_t) &= \left[s_t\sigma_t\right]^{-2}\left(s_t\mathbb{E}(\mathbf{z}_0|\mathbf{z}_t) - \mathbf{z}_t\right).
\end{aligned} \tag{14}$$

We further have $\nabla_{\mathbf{z}_t} \log p_t(\mathbf{z}_t) = [s_t\sigma_t]^{-2}(s_t\mathbb{E}(\mathbf{z}_0|\mathbf{z}_t) - \mathbf{z}_t) = [s_t\sigma_t]^{-1}\mathbb{E}\left(\frac{s_t\mathbf{z}_0 - \mathbf{z}_t}{s_t\sigma_t}|\mathbf{z}_t\right) = -[s_t\sigma_t]^{-1}\mathbb{E}_{p_{t0}(\mathbf{z}_0|\mathbf{z}_t)}\boldsymbol{\epsilon}_t$ due to the linearity of expectation, where $\mathbf{z}_t = s_t\mathbf{z}_0 + s_t\sigma_t\boldsymbol{\epsilon}_t$, $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ according to the transition kernel (10). □

We can train a data-prediction model $r_{\boldsymbol{\theta}}(\mathbf{z}_t; \sigma_t)$ to approximate the posterior $\mathbb{E}(\mathbf{z}_0|\mathbf{z}_t)$, or train a noise-prediction model $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t; \sigma_t)$ to approximate $\mathbb{E}\left(\frac{s_t\mathbf{z}_0 - \mathbf{z}_t}{s_t\sigma_t}|\mathbf{z}_t\right)$ and then substitute the score with the learned model.

The probability flow ordinary differential equation (PF-ODE) of the forward SDE (9) can also be expressed with s_t and σ_t

$$\frac{\mathrm{d}\mathbf{z}_t}{\mathrm{d}t} = \frac{\mathrm{d}\log s_t}{\mathrm{d}t}\mathbf{z}_t - \frac{1}{2}s_t^2\frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t}\nabla_{\mathbf{z}_t}\log p_t(\mathbf{z}_t) = \frac{\mathrm{d}\log s_t}{\mathrm{d}t}\mathbf{z}_t - \frac{\mathrm{d}\log\sigma_t}{\mathrm{d}t}\left(s_t\mathbb{E}(\mathbf{z}_0|\mathbf{z}_t) - \mathbf{z}_t\right). \tag{15}$$

In practice, we have two formulas to calculate the *exact solution* from the current position $\mathbf{z}_{t_{n+1}}$ to the next position $\mathbf{z}_{t_n}$ ($t_{n+1} > t_n$) in the ODE-based sampling to obtain the sampling trajectory from t_N to t_0. One is

$$\mathbf{z}_{t_n} = \mathbf{z}_{t_{n+1}} + \int_{t_{n+1}}^{t_n}\frac{\mathrm{d}\mathbf{z}_t}{\mathrm{d}t}\mathrm{d}t, \tag{16}$$

and another leverages the semi-linear structure in (15) to derive the following equation (Lu et al., 2022a; Zhang & Chen, 2023) with the *variant of constants* formula

$$\begin{aligned}\mathbf{z}_{t_n} &= \exp\left(\int_{t_{n+1}}^{t_n} f(t)\mathrm{d}t\right)\mathbf{z}_{t_{n+1}} - \int_{t_{n+1}}^{t_n}\left(\exp\left(\int_t^{t_n} f(r)\mathrm{d}r\right)\frac{g^2(t)}{2}\nabla_{\mathbf{z}_t}\log p_t(\mathbf{z}_t)\right)\mathrm{d}t \\ &= \frac{s_{t_n}}{s_{t_{n+1}}}\mathbf{z}_{t_{n+1}} - s_{t_n}\int_{t_{n+1}}^{t_n}\left(s_t\sigma_t\sigma_t'\nabla_{\mathbf{z}_t}\log p_t(\mathbf{z}_t)\right)\mathrm{d}t.\end{aligned} \tag{17}$$

The above integral, whether in (16) or (17) is generally intractable. Therefore, the ODE-based sampling in diffusion models is all about how to solve the integral with numerical approximation methods in each step. Typical strategies include Euler method (Song et al., 2021a), Heun's method (Karras et al., 2022), Runge-Kutta method (Song et al., 2021c; Liu et al., 2022; Lu et al., 2022a), and linear multistep method (Liu et al., 2022; Lu et al., 2022b; Zhang & Chen, 2023; Zhao et al., 2023).

A.1.2. The Equivalence of Linear Diffusion Models

We further prove that various linear diffusion models sharing the same SNR are equivalent up to a scaling factor. We also demonstrate that how their score functions and sampling behaviors are connected.

Proposition A.3. *Any diffusion process defined as (11) can be transformed into its VE counterpart with the change-of-variables formula $\mathbf{x}_t = \mathbf{z}_t/s_t$ ($t \in (0, T]$), keeping the SNR unchanged.*

Proof. We adopt the change-of-variables formula $\mathbf{x}_t = \boldsymbol{\phi}(t, \mathbf{z}_t) = \mathbf{z}_t/s_t$ with $\boldsymbol{\phi} : (0, T] \times \mathbb{R}^n \to \mathbb{R}^n$, and we denote the i-th dimension of $\mathbf{z}_t$, $\mathbf{x}_t$ and $\mathbf{w}_t$ as $\mathbf{z}_t[i]$, $\mathbf{x}_t[i]$, and w_t respectively; $\boldsymbol{\phi} = [\phi_1, \cdots, \phi_i, \cdots, \phi_n]^T$ with a twice differentiable scalar function $\phi_i(t, z) = z/s_t$ of two real variables t and z. Since each dimension of $\mathbf{z}_t$ is independent, we can apply Itô's lemma (Oksendal, 2013) to each dimension with $\phi_i(t, \mathbf{z}_t[i])$ separately. We have

$$\frac{\partial\phi_i}{\partial t} = -\frac{z}{s_t^2}\frac{\mathrm{d}s_t}{\mathrm{d}t}, \quad \frac{\partial\phi_i}{\partial z} = \frac{1}{s_t}, \quad \frac{\partial^2\phi_i}{\partial z^2} = 0, \quad \mathrm{d}\mathbf{z}_t[i] = \frac{\mathrm{d}\log s_t}{\mathrm{d}t}\mathbf{z}_t[i]\mathrm{d}t + s_t\sqrt{\frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t}}\mathrm{d}w_t, \tag{18}$$

then

$$\begin{aligned}\mathrm{d}\phi_i(t, \mathbf{z}_t[i]) &= \left(\frac{\partial\phi_i}{\partial t} + f(t)\mathbf{z}_t[i]\frac{\partial\phi_i}{\partial z} + \frac{g^2(t)}{2}\frac{\partial^2\phi_i}{\partial z^2}\right)\mathrm{d}t + g(t)\frac{\partial\phi_i}{\partial z}\mathrm{d}w_t \\ &= \left(\frac{\partial\phi_i}{\partial t} + \frac{g^2(t)}{2}\frac{\partial^2\phi_i}{\partial z^2}\right)\mathrm{d}t + \frac{\partial\phi_i}{\partial z}\mathrm{d}\mathbf{z}_t[i] \\ \mathrm{d}\mathbf{x}_t[i] &= -\frac{\mathbf{z}_t[i]}{s_t}\frac{\mathrm{d}\log s_t}{\mathrm{d}t}\mathrm{d}t + \frac{1}{s_t}\left(\frac{\mathrm{d}\log s_t}{\mathrm{d}t}\mathbf{z}_t[i]\mathrm{d}t + s_t\sqrt{\frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t}}\mathrm{d}w_t\right) \\ \mathrm{d}\mathbf{x}_t[i] &= \sqrt{\frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t}}\,\mathrm{d}w_t, \quad \to \quad \mathrm{d}\mathbf{x}_t = \sqrt{\frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t}}\,\mathrm{d}\mathbf{w}_t,\end{aligned} \tag{19}$$

with the initial condition $\mathbf{x}_0 = \mathbf{z}_0 \sim p_d$. Since σ_t in the above VE-SDE ($\mathbf{x}$-space) is exactly the same as that used in the original SDE ($\mathbf{z}$-space, (11)), the SNR remains unchanged. □

Similarly, we provide the PF-ODE in the $\mathbf{x}$-space as follows

$$\mathrm{d}\mathbf{x}_t = -\sigma_t \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) \mathrm{d}\sigma_t, \tag{20}$$

with the score function for $t \in (0, T]$

$$\begin{aligned} \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) &= s_t \nabla_{\mathbf{z}_t} \log \int \mathcal{N}\left(\mathbf{z}_t/s_t; \mathbf{z}_0, \sigma_t^2 \mathbf{I}\right) p_d(\mathbf{z}_0) \mathrm{d}\mathbf{z}_0 \\ &= s_t \nabla_{\mathbf{z}_t} \log \int s_t^d \mathcal{N}\left(\mathbf{z}_t; s_t\mathbf{z}_0, s_t^2\sigma_t^2 \mathbf{I}\right) p_d(\mathbf{z}_0) \mathrm{d}\mathbf{z}_0 \\ &= s_t \nabla_{\mathbf{z}_t} \log p_t(\mathbf{z}_t). \end{aligned} \tag{21}$$

This equation also holds for $t = 0$ since $s_0 = 1$. Thus, $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) = s_t \nabla_{\mathbf{z}_t} \log p_t(\mathbf{z}_t)$ for $t \in [0, T]$.

Corollary A.4. *With the same numerical method, the results obtained by using (16) or (17) are not equal in general cases. But they become exactly the same for VE-SDEs in the $\mathbf{x}$-space.*

Corollary A.5. *With the same numerical method, the result obtained by using (17) in the $\mathbf{z}$-space is exactly the same as the result obtained by using (16) or (17) in the $\mathbf{x}$-space.*

Sketch of proof. Given the sample $\mathbf{z}_{t_n}$ obtained by solving the equation (17) in $\mathbf{z}$-space starting from $\mathbf{z}_{t_{n+1}}$, we prove that $\mathbf{z}_{t_n}/s_{t_n}$ is exactly equal to sampling with the equation (16) in $\mathbf{x}$-space starting from $\mathbf{x}_{t_{n+1}} = \mathbf{z}_{t_{n+1}}/s_{t_{n+1}}$ to $\mathbf{x}_{t_n}$. We have

$$\begin{aligned} \mathbf{z}_{t_n} &= s_{t_n}\left(\frac{\mathbf{z}_{t_{n+1}}}{s_{t_{n+1}}} - \int_{t_{n+1}}^{t_n} s_t \sigma_t \sigma_t' \nabla_{\mathbf{z}_t} \log p_t(\mathbf{z}_t) \mathrm{d}t\right) = s_{t_n}\left(\mathbf{x}_{t_{n+1}} + \int_{t_{n+1}}^{t_n} -\sigma_t \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) \sigma_t' \mathrm{d}t\right) \\ &= s_{t_n}\left(\mathbf{x}_{t_{n+1}} + \int_{t_{n+1}}^{t_n} \frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} \mathrm{d}t\right) = s_{t_n}\mathbf{x}_{t_n}. \end{aligned} \tag{22}$$

□

A.2. Generalized Denoising Output

All following proofs are conducted in the context of a VE-SDE $\mathrm{d}\mathbf{x}_t = \sqrt{2t}\,\mathrm{d}\mathbf{w}_t$, *i.e.*, $\sigma_t = t$ for notation simplicity, and the sampling trajectory always starts from $\hat{\mathbf{x}}_{t_N} \sim \mathcal{N}(\mathbf{0}, T^2\mathbf{I})$ and ends at $\hat{\mathbf{x}}_{t_0}$.

The PF-ODEs of the sampling trajectory and denoising trajectory are provided as follows

$$\text{sampling-ODE: } \frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} = \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t; t) = \frac{\mathbf{x}_t - r_{\boldsymbol{\theta}}(\mathbf{x}_t; t)}{t}, \quad \text{denoising-ODE: } \frac{\mathrm{d}r_{\boldsymbol{\theta}}(\mathbf{x}_t; t)}{\mathrm{d}t} = -t\frac{\mathrm{d}^2\mathbf{x}_t}{\mathrm{d}t^2}. \tag{23}$$

The denoising-ODE above is derived by simply rearranging the sampling-ODE as $r_{\boldsymbol{\theta}}(\mathbf{x}_t; t) = \mathbf{x}_t - t\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t}$, and then take the derivative of both sides, *i.e.*, $\frac{\mathrm{d}r_{\boldsymbol{\theta}}(\mathbf{x}_t;t)}{\mathrm{d}t} = \frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} - \left(\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} + t\frac{\mathrm{d}^2\mathbf{x}_t}{\mathrm{d}t^2}\right) = -t\frac{\mathrm{d}^2\mathbf{x}_t}{\mathrm{d}t^2}$.

A.2.1. Convex Combination

Proposition A.6. *Given the probability flow ODE (5) and a current position $\hat{\mathbf{x}}_{t_{n+1}}$, $n \in [0, N-1]$ in the sampling trajectory, the next position $\hat{\mathbf{x}}_{t_n}$ predicted by a k-order Taylor expansion with the time step size $t_{n+1} - t_n$ equals*

$$\hat{\mathbf{x}}_{t_n} = \frac{t_n}{t_{n+1}}\hat{\mathbf{x}}_{t_{n+1}} + \frac{t_{n+1} - t_n}{t_{n+1}} h_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}). \tag{24}$$

which is a convex combination of $\hat{\mathbf{x}}_{t_{n+1}}$ and the generalized denoising output $h_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})$,

$$h_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) = r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - \sum_{i=2}^{k} \frac{1}{i!} \frac{\mathrm{d}^{(i)}\mathbf{x}_t}{\mathrm{d}t^{(i)}}\Big|_{\hat{\mathbf{x}}_{t_{n+1}}} t_{n+1}(t_n - t_{n+1})^{i-1}. \tag{25}$$

We have $h_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) = r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})$ for Euler method ($k = 1$), and $h_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) = r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) + \frac{t_n - t_{n+1}}{2} \frac{\mathrm{d}r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{\mathrm{d}t}$ for second-order methods ($k = 2$).

Proof. The k-order Taylor expansion at $\hat{\mathbf{x}}_{t_{n+1}}$ is

$$\begin{aligned}
\hat{\mathbf{x}}_{t_n} &= \sum_{i=0}^{k} \frac{1}{i!} \frac{\mathrm{d}^{(i)}\mathbf{x}_t}{\mathrm{d}t^{(i)}}\Big|_{\hat{\mathbf{x}}_{t_{n+1}}} (t_n - t_{n+1})^i \\
&= \hat{\mathbf{x}}_{t_{n+1}} + (t_n - t_{n+1}) \frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t}\Big|_{\hat{\mathbf{x}}_{t_{n+1}}} + \sum_{i=2}^{k} \frac{1}{i!} \frac{\mathrm{d}^{(i)}\mathbf{x}_t}{\mathrm{d}t^{(i)}}\Big|_{\hat{\mathbf{x}}_{t_{n+1}}} (t_n - t_{n+1})^i \\
&= \hat{\mathbf{x}}_{t_{n+1}} + \frac{t_n - t_{n+1}}{t_{n+1}} \left(\hat{\mathbf{x}}_{t_{n+1}} - r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})\right) + \sum_{i=2}^{k} \frac{1}{i!} \frac{\mathrm{d}^{(i)}\mathbf{x}_t}{\mathrm{d}t^{(i)}}\Big|_{\hat{\mathbf{x}}_{t_{n+1}}} (t_n - t_{n+1})^i \\
&= \frac{t_n}{t_{n+1}} \hat{\mathbf{x}}_{t_{n+1}} + \frac{t_{n+1} - t_n}{t_{n+1}} h_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}),
\end{aligned} \tag{26}$$

where $h_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) = r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - \sum_{i=2}^{k} \frac{1}{i!} \frac{\mathrm{d}^{(i)}\mathbf{x}_t}{\mathrm{d}t^{(i)}}\Big|_{\hat{\mathbf{x}}_{t_{n+1}}} t_{n+1}(t_n - t_{n+1})^{i-1}$. As for first-order approximation ($k = 1$), we have $h_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) = r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})$. As for second-order approximation ($k = 2$), we have

$$h_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) = r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - \frac{t_{n+1}(t_n - t_{n+1})}{2} \frac{\mathrm{d}^2\mathbf{x}_t}{\mathrm{d}t^2}\Big|_{\hat{\mathbf{x}}_{t_{n+1}}} = r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) + \frac{t_n - t_{n+1}}{2} \frac{\mathrm{d}r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{\mathrm{d}t}. \tag{27}$$

□

Corollary A.7. *The denoising output $r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})$ reflects the prediction made by a single Euler step from $\hat{\mathbf{x}}_{t_{n+1}}$ with the time step size t_{n+1}.*

Proof. The prediction of such an Euler step equals to $\hat{\mathbf{x}}_{t_{n+1}} + (0 - t_{n+1}) \left(\hat{\mathbf{x}}_{t_{n+1}} - r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})\right) / t_{n+1} = r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})$. □

Corollary A.8. *Each previously proposed second-order ODE-based accelerated sampling method corresponds to a specific first-order finite difference of $\mathrm{d}r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})/\mathrm{d}t$.*

Proof. The proof is provided in the next section. □

A.2.2. Second-Order ODE Samplers as Finite Differences of Denoising Trajectory

To accelerate the sampling speed of diffusion models, various numerical solver-based samplers have been developed in the past several years (Song et al., 2021a;c; Karras et al., 2022; Lu et al., 2022a; Zhang & Chen, 2023). In particular, second-order ODE-based samplers are relatively promising in the practical use since they strike a good balance between fast sampling and decent visual quality (Rombach et al., 2022; Balaji et al., 2022).

We next demonstrate that each of them can be rewritten as a specific way to perform finite difference of the denoising trajectory $\mathrm{d}r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})/\mathrm{d}t$, as shown in Table 5. We assume that history points are already available for samplers in PNDM and DEIS and they are calculated with Euler method (otherwise, we need to consider Taylor expansion in terms of the data-prediction model $r_{\boldsymbol{\theta}}(\mathbf{x}_t)$ rather than the noise-prediction model $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t)$ as the implementation in the original papers).

A.2.3. EDMs (Karras et al., 2022)

EDMs employ Heun's 2nd order method, where one Euler step is first applied and followed by a second order correction, which can be written as

$$\begin{aligned}
\hat{\mathbf{x}}'_{t_n} &= \hat{\mathbf{x}}_{t_{n+1}} + (t_n - t_{n+1}) \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}), \\
\hat{\mathbf{x}}_{t_n} &= \hat{\mathbf{x}}_{t_{n+1}} + (t_n - t_{n+1}) \left(0.5 \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) + 0.5 \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}'_{t_n})\right) \\
&= \hat{\mathbf{x}}_{t_{n+1}} + (t_n - t_{n+1}) \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) + \frac{1}{2}(t_n - t_{n+1})^2 \frac{\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}'_{t_n}) - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{t_n - t_{n+1}}.
\end{aligned} \tag{28}$$

By using $r_{\boldsymbol{\theta}}(\mathbf{x}_t;t)=\mathbf{x}_t-t\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t;t)$, the sampling iteration above is equivalent to

$$\begin{aligned}\hat{\mathbf{x}}_{t_n}&=\hat{\mathbf{x}}_{t_{n+1}}+(t_n-t_{n+1})\frac{\hat{\mathbf{x}}_{t_{n+1}}-r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{t_{n+1}}+\frac{1}{2}(t_n-t_{n+1})^2\frac{\frac{\hat{\mathbf{x}}'_{t_n}-r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}'_{t_n})}{t_n}-\frac{\hat{\mathbf{x}}_{t_{n+1}}-r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{t_{n+1}}}{t_n-t_{n+1}}\\&=\frac{t_n}{t_{n+1}}\hat{\mathbf{x}}_{t_{n+1}}+\frac{t_{n+1}-t_n}{t_{n+1}}r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})-\frac{1}{2}\frac{(t_n-t_{n+1})^2}{t_{n+1}}\frac{t_{n+1}}{t_n}\frac{r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}'_{t_n})-r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{t_n-t_{n+1}}\\&=\frac{t_n}{t_{n+1}}\hat{\mathbf{x}}_{t_{n+1}}+\frac{t_{n+1}-t_n}{t_{n+1}}\left(r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})+\frac{t_n-t_{n+1}}{2}\frac{t_{n+1}}{t_n}\frac{r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}'_{t_n})-r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{t_n-t_{n+1}}\right).\end{aligned}\tag{29}$$

Compared with the generalized denoising output (27), we have $\frac{\mathrm{d}r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{\mathrm{d}t}\approx\frac{t_{n+1}}{t_n}\frac{r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}'_{t_n})-r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{t_n-t_{n+1}}$.

A.2.4. DPM-SOLVER (LU ET AL., 2022A)

According to (3.3) in DPM-Solver, the exact solution of PF-ODE in the VE-SDE setting is given by

$$\hat{\mathbf{x}}_{t_n}=\hat{\mathbf{x}}_{t_{n+1}}+\int_{t_{n+1}}^{t_n}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_t)\mathrm{d}t.\tag{30}$$

The Taylor expansion of $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_t)$ *w.r.t.* time at t_{n+1} is

$$\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_t)=\sum_{m=0}^{k-1}\frac{(t-t_{n+1})^m}{m!}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}^{(m)}(\hat{\mathbf{x}}_{t_{n+1}})+\mathcal{O}\left((t-t_{n+1})^k\right).\tag{31}$$

Then, the DPM-Solver-k sampler can be written as

$$\begin{aligned}\hat{\mathbf{x}}_{t_n}&=\hat{\mathbf{x}}_{t_{n+1}}+\sum_{m=0}^{k-1}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}^{(m)}(\hat{\mathbf{x}}_{t_{n+1}})\int_{t_{n+1}}^{t_n}\frac{(t-t_{n+1})^m}{m!}\mathrm{d}t+\mathcal{O}\left((t_n-t_{n+1})^k\right)\\&=\hat{\mathbf{x}}_{t_{n+1}}+\sum_{m=0}^{k-1}\frac{(t_n-t_{n+1})^{m+1}}{(m+1)!}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}^{(m)}(\hat{\mathbf{x}}_{t_{n+1}})+\mathcal{O}\left((t_n-t_{n+1})^{k+1}\right).\end{aligned}\tag{32}$$

Specifically, when $k=2$, the DPM-Solver-2 sampler is given by

$$\hat{\mathbf{x}}_{t_n}=\hat{\mathbf{x}}_{t_{n+1}}+(t_n-t_{n+1})\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})+\frac{1}{2}(t_n-t_{n+1})^2\frac{\mathrm{d}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{\mathrm{d}t},\tag{33}$$

The above second-order term in (Lu et al., 2022a) is approximated by

$$\frac{\mathrm{d}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{\mathrm{d}t}\approx\frac{\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{s_n})-\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{(t_n-t_{n+1})/2},\tag{34}$$

where $s_n=\sqrt{t_nt_{n+1}}$ and $\hat{\mathbf{x}}_{s_n}=\hat{\mathbf{x}}_{t_{n+1}}+(s_n-t_{n+1})\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})$. We have

$$\begin{aligned}\hat{\mathbf{x}}_{t_n}&=\hat{\mathbf{x}}_{t_{n+1}}+(t_n-t_{n+1})\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})+\frac{1}{2}(t_n-t_{n+1})^2\frac{\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{s_n})-\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{(t_n-t_{n+1})/2}\\&=\frac{t_n}{t_{n+1}}\hat{\mathbf{x}}_{t_{n+1}}+\frac{t_{n+1}-t_n}{t_{n+1}}r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})-\frac{1}{2}\frac{(t_n-t_{n+1})^2}{t_{n+1}}\frac{t_{n+1}}{s_n}\frac{r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{s_n})-r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{(t_n-t_{n+1})/2}\\&=\frac{t_n}{t_{n+1}}\hat{\mathbf{x}}_{t_{n+1}}+\frac{t_{n+1}-t_n}{t_{n+1}}\left(r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})+\frac{t_n-t_{n+1}}{2}\frac{t_{n+1}}{s_n}\frac{r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{s_n})-r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{(t_n-t_{n+1})/2}\right).\end{aligned}\tag{35}$$

Compared with the generalized denoising output (27), we have $\frac{\mathrm{d}r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{\mathrm{d}t}\approx\frac{t_{n+1}}{s_n}\frac{r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{s_n})-r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{(t_n-t_{n+1})/2}$.

Table 5. Each second-order ODE-based sampler listed below corresponds to a specific finite difference of the denoising trajectory. γ denotes a correction coefficient of forward differences. DDIM is a first-order sampler listed for comparison. GENIE trains a neural network to approximate high-order derivatives. $r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+2}})$ in S-PNDM and DEIS denotes a previous denoising output. $s_n = \sqrt{t_n t_{n+1}}$ in DPM-Solver-2. $\hat{\mathbf{x}}'_{t_n}$ in EDMs denotes the output of an intermediate Euler step.

ODE solver-based samplers	$\mathrm{d}r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})/\mathrm{d}t$	γ
DDIM (Song et al., 2021a)	None	None
GENIE (Dockhorn et al., 2022)	Neural Networks	None
S-PNDM (Liu et al., 2022)	$\gamma\left(r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+2}})\right)/(t_n - t_{n+1})$	1
DEIS (ρAB1) (Zhang & Chen, 2023)	$\gamma\left(r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+2}})\right)/(t_{n+1} - t_{n+2})$	1
DPM-Solver-2 (Lu et al., 2022a)	$\gamma\left(r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{s_n}) - r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})\right)/\left((t_n - t_{n+1})/2\right)$	t_{n+1}/s_n
EDMs (Heun) (Karras et al., 2022)	$\gamma\left(r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}'_{t_n}) - r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})\right)/(t_n - t_{n+1})$	t_{n+1}/t_n

A.2.5. PNDM (Liu et al., 2022)

Assume that the previous denoising output $r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+2}})$ is available, then one S-PNDM sampler step can be written as

$$\begin{aligned}
\hat{\mathbf{x}}_{t_n} &= \hat{\mathbf{x}}_{t_{n+1}} + (t_n - t_{n+1})\frac{1}{2}\left(3\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+2}})\right) \\
&= \hat{\mathbf{x}}_{t_{n+1}} + (t_n - t_{n+1})\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) + \frac{1}{2}(t_n - t_{n+1})^2\frac{\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+2}})}{t_n - t_{n+1}} \\
&= \frac{t_n}{t_{n+1}}\hat{\mathbf{x}}_{t_{n+1}} + \frac{t_{n+1} - t_n}{t_{n+1}}r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - \frac{1}{2}\frac{(t_n - t_{n+1})^2}{t_{n+1}}\frac{r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+2}})}{t_n - t_{n+1}} \\
&= \frac{t_n}{t_{n+1}}\hat{\mathbf{x}}_{t_{n+1}} + \frac{t_{n+1} - t_n}{t_{n+1}}\left(r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) + \frac{t_n - t_{n+1}}{2}\frac{r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+2}})}{t_n - t_{n+1}}\right).
\end{aligned} \tag{36}$$

Compared with the generalized denoising output (27), we have $\frac{\mathrm{d}r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{\mathrm{d}t} \approx \frac{r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+2}})}{t_n - t_{n+1}}$.

A.2.6. DEIS (Zhang & Chen, 2023)

In DEIS paper, the solution of PF-ODE in the VE-SDE setting is given by

$$\begin{aligned}
\hat{\mathbf{x}}_{t_n} &= \hat{\mathbf{x}}_{t_{n+1}} + \sum_{j=0}^{r} C_{(n+1)j}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1+j}}), \\
C_{(n+1)j} &= \int_{t_{n+1}}^{t_n}\prod_{k\neq j}\frac{\tau - t_{n+1+k}}{t_{n+1+j} - t_{n+1+k}}\mathrm{d}\tau.
\end{aligned} \tag{37}$$

$r = 1$ yields the ρAB1 sampler:

$$\begin{aligned}
\hat{\mathbf{x}}_{t_n} &= \hat{\mathbf{x}}_{t_{n+1}} + \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})\int_{t_{n+1}}^{t_n}\frac{\tau - t_{n+2}}{t_{n+1} - t_{n+2}}\mathrm{d}\tau + \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+2}})\int_{t_{n+1}}^{t_n}\frac{\tau - t_{n+1}}{t_{n+2} - t_{n+1}}\mathrm{d}\tau \\
&= \hat{\mathbf{x}}_{t_{n+1}} + \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})\frac{(t_n - t_{n+2})^2 - (t_{n+1} - t_{n+2})^2}{2(t_{n+1} - t_{n+2})} + \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+2}})\frac{(t_n - t_{n+1})^2}{2(t_{n+2} - t_{n+1})} \\
&= \hat{\mathbf{x}}_{t_{n+1}} + (t_n - t_{n+1})\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) + \frac{1}{2}(t_n - t_{n+1})^2\frac{\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+2}})}{t_{n+1} - t_{n+2}} \\
&= \frac{t_n}{t_{n+1}}\hat{\mathbf{x}}_{t_{n+1}} + \frac{t_{n+1} - t_n}{t_{n+1}}r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - \frac{1}{2}\frac{(t_n - t_{n+1})^2}{t_{n+1}}\frac{r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+2}})}{t_{n+1} - t_{n+2}} \\
&= \frac{t_n}{t_{n+1}}\hat{\mathbf{x}}_{t_{n+1}} + \frac{t_{n+1} - t_n}{t_{n+1}}\left(r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) + \frac{t_n - t_{n+1}}{2}\frac{r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+2}})}{t_{n+1} - t_{n+2}}\right).
\end{aligned} \tag{38}$$

Compared with the generalized denoising output (27), we have $\frac{\mathrm{d}r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})}{\mathrm{d}t} \approx \frac{r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+2}})}{t_{n+1} - t_{n+2}}$.

A.3. Theoretical Analysis of the Trajectory Structure

A.3.1. The Optimal Denoising Output

Lemma A.9. *The optimal estimator $r^\star_{\boldsymbol{\theta}}(\mathbf{x}_t;\sigma_t)$, also known as Bayesian least squares estimator, of the minimization of denoising autoencoder (DAE) objective is the conditional expectation $\mathbb{E}(\mathbf{x}_0|\mathbf{x}_t)$*

$$\mathcal{L}_{\mathrm{DAE}} = \mathbb{E}_{\mathbf{x}_0\sim p_d(\mathbf{x}_0)}\mathbb{E}_{\mathbf{x}_t\sim p_{0t}(\mathbf{x}_t|\mathbf{x}_0)}\|r_{\boldsymbol{\theta}}(\mathbf{x}_t;\sigma_t)-\mathbf{x}_0\|_2^2 = \int p_t(\mathbf{x}_t)p_{t0}(\mathbf{x}_0|\mathbf{x}_t)\|r_{\boldsymbol{\theta}}(\mathbf{x}_t;\sigma_t)-\mathbf{x}_0\|_2^2. \tag{39}$$

Proof. The solution can be easily obtained by setting the derivative of $\mathcal{L}_{\mathrm{DAE}}$ equal to zero. □

Suppose we have a training dataset $\mathcal{D}=\{\mathbf{y}_i\}_{i=1}^{|\mathcal{I}|}$ where each data $\mathbf{y}_i$ is sampled from an unknown data distribution p_d. The empirical data distribution $\hat{p}_d$ is denoted as a summation of multiple *Dirac delta functions* (*a.k.a* a mixture of Gaussian distribution): $\hat{p}_d(\mathbf{y}) = \frac{1}{|\mathcal{I}|}\sum_{i=1}^{|\mathcal{I}|}\delta(\|\mathbf{y}-\mathbf{y}_i\|)$, and the Gaussian kernel density estimate (KDE) is

$$\hat{p}_t(\mathbf{x}_t) = \int p_{0t}(\mathbf{x}_t|\mathbf{y})\hat{p}_d(\mathbf{y}) = \frac{1}{|\mathcal{I}|}\sum_i \mathcal{N}\left(\mathbf{x}_t;\mathbf{y}_i,\sigma_t^2\mathbf{I}\right). \tag{40}$$

Proposition A.10. *The optimal denoising output of training a denoising autoencoder with the empirical data distribution is a convex combination of original data points, where each weight u_i is calculated based on the time-scaled and normalized ℓ_2 distance between the input $\mathbf{x}_t$ and $\mathbf{y}_i$ belonging to the dataset $\mathcal{D}$:*

$$r^\star_{\boldsymbol{\theta}}(\mathbf{x}_t;\sigma_t) = \min_{r_{\boldsymbol{\theta}}}\mathbb{E}_{\mathbf{y}\sim\hat{p}_d}\mathbb{E}_{\mathbf{x}_t\sim p_{0t}(\mathbf{x}_t|\mathbf{y})}\|r_{\boldsymbol{\theta}}(\mathbf{x}_t;\sigma_t)-\mathbf{y}\|_2^2 = \sum_i \frac{\exp\left(-\|\mathbf{x}_t-\mathbf{y}_i\|_2^2/2\sigma_t^2\right)}{\sum_j\exp\left(-\|\mathbf{x}_t-\mathbf{y}_j\|_2^2/2\sigma_t^2\right)}\mathbf{y}_i = \sum_i u_i\mathbf{y}_i, \tag{41}$$

with the coefficients satisfying $\sum_i u_i = 1$.

This equation appears to be highly similar to the iterative formula used in mean shift (Fukunaga & Hostetler, 1975; Cheng, 1995; Comaniciu & Meer, 2002; Yamasaki & Tanaka, 2020), especially annealed mean shift (Shen et al., 2005).

Proof. Based on Lemmas A.2 and A.9, the optimal denoising output is

$$\begin{aligned}
r^\star_{\boldsymbol{\theta}}(\mathbf{x}_t;\sigma_t) = \mathbb{E}(\mathbf{y}|\mathbf{x}_t) &= \mathbf{x}_t + \sigma_t^2\nabla_{\mathbf{x}_t}\log\hat{p}_t(\mathbf{x}_t)\\
&= \mathbf{x}_t + \sigma_t^2\sum_i\frac{\nabla_{\mathbf{x}_t}\mathcal{N}\left(\mathbf{x}_t;\mathbf{y}_i,\sigma_t^2\mathbf{I}\right)}{\sum_j\mathcal{N}\left(\mathbf{x}_t;\mathbf{y}_j,\sigma_t^2\mathbf{I}\right)}\\
&= \mathbf{x}_t + \sigma_t^2\sum_i\frac{\mathcal{N}\left(\mathbf{x}_t;\mathbf{y}_i,\sigma_t^2\mathbf{I}\right)}{\sum_j\mathcal{N}\left(\mathbf{x}_t;\mathbf{y}_j,\sigma_t^2\mathbf{I}\right)}\left(\frac{\mathbf{y}_i-\mathbf{x}_t}{\sigma_t^2}\right)\\
&= \mathbf{x}_t + \sum_i\frac{\mathcal{N}\left(\mathbf{x}_t;\mathbf{y}_i,\sigma_t^2\mathbf{I}\right)}{\sum_j\mathcal{N}\left(\mathbf{x}_t;\mathbf{y}_j,\sigma_t^2\mathbf{I}\right)}\left(\mathbf{y}_i-\mathbf{x}_t\right)\\
&= \sum_i\frac{\mathcal{N}\left(\mathbf{x}_t;\mathbf{y}_i,\sigma_t^2\mathbf{I}\right)}{\sum_j\mathcal{N}\left(\mathbf{x}_t;\mathbf{y}_j,\sigma_t^2\mathbf{I}\right)}\mathbf{y}_i\\
&= \sum_i\frac{\exp\left(-\|\mathbf{x}_t-\mathbf{y}_i\|_2^2/2\sigma_t^2\right)}{\sum_j\exp\left(-\|\mathbf{x}_t-\mathbf{y}_j\|_2^2/2\sigma_t^2\right)}\mathbf{y}_i.
\end{aligned} \tag{42}$$

□

A.3.2. Theoretical Connection to Mean Shift

Mean shift is a well-known non-parametric algorithm designed to seek modes of a density function, typical a KDE, via iteratively gradient ascent with adaptive step sizes. Given a current position $\mathbf{x}$, mean shift with a Gaussian kernel and

bandwidth h iteratively adds a vector $\mathbf{m}(\mathbf{x}) - \mathbf{x}$, which points toward the maximum increase in the kernel density estimate $p_h(\mathbf{x}) = \frac{1}{|\mathcal{I}|}\sum_{i=1}^{|\mathcal{I}|}\mathcal{N}(\mathbf{x}; \mathbf{y}_i, h^2\mathbf{I})$, to itself, *i.e.*, $\mathbf{x} \leftarrow [\mathbf{m}(\mathbf{x}) - \mathbf{x}] + \mathbf{x}$. The *mean vector* is

$$\mathbf{m}(\mathbf{x}, h) = \sum_i v_i \mathbf{y}_i = \sum_i \frac{\exp\left(-\|\mathbf{x} - \mathbf{y}_i\|_2^2/2h^2\right)}{\sum_j \exp\left(-\|\mathbf{x} - \mathbf{y}_j\|_2^2/2h^2\right)}\mathbf{y}_i, \tag{43}$$

with the coefficients satisfying $\sum_i v_i = 1$. As a mode-seeking algorithm, mean shift has shown particularly successful in clustering (Cheng, 1995; Carreira-Perpinán, 2015), image segmentation (Comaniciu & Meer, 1999; 2002) and video tracking (Comaniciu et al., 2000; 2003). By treating the bandwidth σ_t in (41) as the bandwidth h in (43), we build a connection between the optimal denoising output of a diffusion model and annealed mean shift under the KDE-based data modeling.

A.3.3. The Constant Magnitude of Vector Field and Sampling Trajectory Length

Lemma A.11 (see Section 3.1 in (Vershynin, 2018))**.** *Given a high-dimensional isotropic Gaussian noise $\mathbf{z} \sim \mathcal{N}(\mathbf{0}; \sigma^2\mathbf{I}_d)$, $\sigma > 0$, we have $\mathbb{E}\left\|\mathbf{z}\right\|^2 = \sigma^2 d$, and with high probability, $\mathbf{z}$ stays within a "thin spherical shell": $\|\mathbf{z}\| = \sigma\sqrt{d} \pm O(1)$.*

Proof. We denote $\mathbf{z}_i$ as the i-th dimension of random variable $\mathbf{z}$, then the expectation and variance is $\mathbb{E}\left[\mathbf{z}_i\right] = 0$, $\mathbb{V}\left[\mathbf{z}_i\right] = \sigma^2$, respectively. The fourth central moment is $\mathbb{E}\left[\mathbf{z}_i^4\right] = 3\sigma^4$. Additionally,

$$\begin{aligned} \mathbb{E}\left[\mathbf{z}_i^2\right] = \mathbb{V}\left[\mathbf{z}_i\right] + \mathbb{E}\left[\mathbf{z}_i\right]^2 = \sigma^2, \qquad \mathbb{E}\left[\|\mathbf{z}\|^2\right] = \mathbb{E}\left[\sum_{i=1}^d \mathbf{z}_i^2\right] = \sum_{i=1}^d \mathbb{E}\left[\mathbf{z}_i^2\right] = \sigma^2 d, \\ \mathbb{V}\left[\|\mathbf{z}\|^2\right] = \mathbb{E}\left[\|\mathbf{z}\|^4\right] - \left(\mathbb{E}\left[\|\mathbf{z}\|^2\right]\right)^2 = 2d\sigma^4, \end{aligned} \tag{44}$$

Then, we have

$$\mathbb{E}\left[\|\mathbf{x} + \mathbf{z}\|^2 - \|\mathbf{x}\|^2\right] = \mathbb{E}\left[\|\mathbf{z}\|^2 + 2\mathbf{x}^T\mathbf{z}\right] = \mathbb{E}\left[\|\mathbf{z}\|^2\right] = \sigma^2 d. \tag{45}$$

Furthermore, the standard deviation of $\|\mathbf{z}\|^2$ is $\sigma^2\sqrt{2d}$, which means

$$\|\mathbf{z}\|^2 = \sigma^2 d \pm \sigma^2\sqrt{2d} = \sigma^2 d \pm O\left(\sqrt{d}\right), \qquad \|\mathbf{z}\| \ = \sigma\sqrt{d} \pm O\left(1\right), \tag{46}$$

holds with high probability. □

We denote the optimal denoising output in the KDE-based data modeling as $r^\star_{\boldsymbol{\theta}}(\mathbf{x}_t; \sigma_t)$ (see Section A.3.1). In this case, the optimal noise prediction is denoted as $\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta}}(\mathbf{x}_t; \sigma_t) = \frac{\mathbf{x}_t - r^\star_{\boldsymbol{\theta}}(\mathbf{x}_t)}{\sigma_t}$, and the optimal empirical PF-ODE in (5) becomes

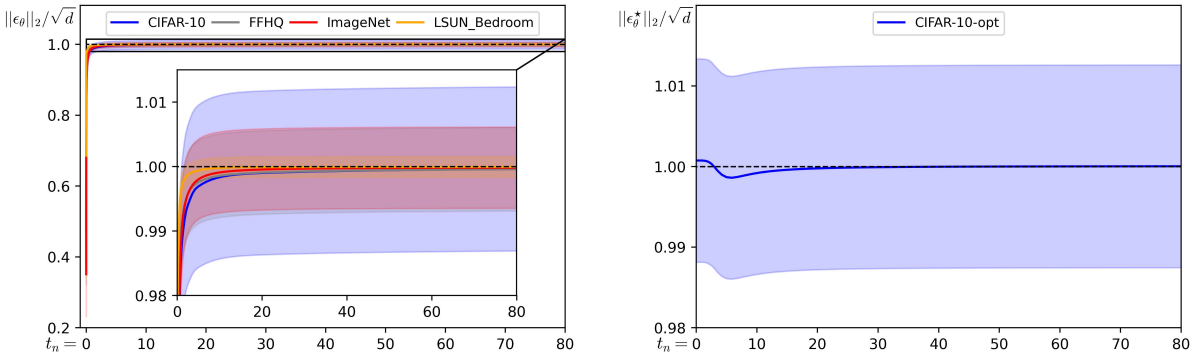
$$\mathrm{d}\mathbf{x}_t = \boldsymbol{\epsilon}^\star_{\boldsymbol{\theta}}(\mathbf{x}_t; \sigma_t)\mathrm{d}\sigma_t. \tag{47}$$

Remark A.12. Intriguingly, the magnitude of $\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta}}(\mathbf{x}_t; \sigma_t)$ approximately distributes around $\sqrt{d}$. The total trajectory length approximately equals $\sigma_T\sqrt{d}$, where d denotes the data dimension.

We next provide a sketch of proof. Suppose the data distribution lies in a smooth real low-dimensional manifold with the intrinsic dimension as m. According to the *Whitney embedding theorem* (Whitney, 1936), it can be smoothly embedded in a real $2m$ Euclidean space. We then decompose each $\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta}} \in \mathbb{R}^d$ vector as $\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta},\parallel}$ and $\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta},\perp}$, which are parallel and perpendicular to the $2m$ Euclidean space, respectively. Therefore, we have $\|\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta}}\|_2 = \|\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta},\parallel} + \boldsymbol{\epsilon}^\star_{\boldsymbol{\theta},\perp}\|_2 \geq \|\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta},\perp}\|_2 \approx \sqrt{d - 2m}$.

We provide a upper bound for the $\|\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta}}\|_2$ below

$$\begin{aligned} \mathbb{E}_{p_t(\mathbf{x}_t)}\|\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta}}\|_2 &= \mathbb{E}_{p_t(\mathbf{x}_t)}\|\frac{\mathbf{x}_t - r^\star_{\boldsymbol{\theta}}(\mathbf{x}_t)}{\sigma_t}\|_2 = \mathbb{E}_{p_t(\mathbf{x}_t)}\|\frac{\mathbf{x}_t - \mathbb{E}(\mathbf{x}_0|\mathbf{x}_t)}{\sigma_t}\|_2 \\ &= \mathbb{E}_{p_t(\mathbf{x}_t)}\|\mathbb{E}(\frac{\mathbf{x}_t - \mathbf{x}_0}{\sigma_t}|\mathbf{x}_t)\|_2 = \mathbb{E}_{p_t(\mathbf{x}_t)}\|\mathbb{E}_{p_{t0}(\mathbf{x}_0|\mathbf{x}_t)}\boldsymbol{\epsilon}\|_2 \\ &\leq \mathbb{E}_{p_t(\mathbf{x}_t)}\mathbb{E}_{p_{t0}(\mathbf{x}_0|\mathbf{x}_t)}\|\boldsymbol{\epsilon}\|_2 \\ &\leq \mathbb{E}_{p_0(\mathbf{x}_0)}\mathbb{E}_{p_{0t}(\mathbf{x}_t|\mathbf{x}_0)}\|\boldsymbol{\epsilon}\|_2 \\ &\approx \sqrt{d} \qquad \text{(concentration of measure, Lemma A.11)}. \end{aligned} \tag{48}$$



(a) The L^2 norm of $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$.

(b) The L^2 norm of $\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta}}$.

Figure 8. The optimal noise prediction $\|\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta}}\|_2 \approx \sqrt{d}$ in the whole sampling process, as the theoretical results guarantee. The actual noise prediction $\|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}\|_2 \approx \sqrt{d}$ in the most timestamps, but shrinks in the final stage (when the timestamp is very close to zero). Such norm shrinkage almost does not affect the trajectory length as the discretized time steps are very small in the final stage.

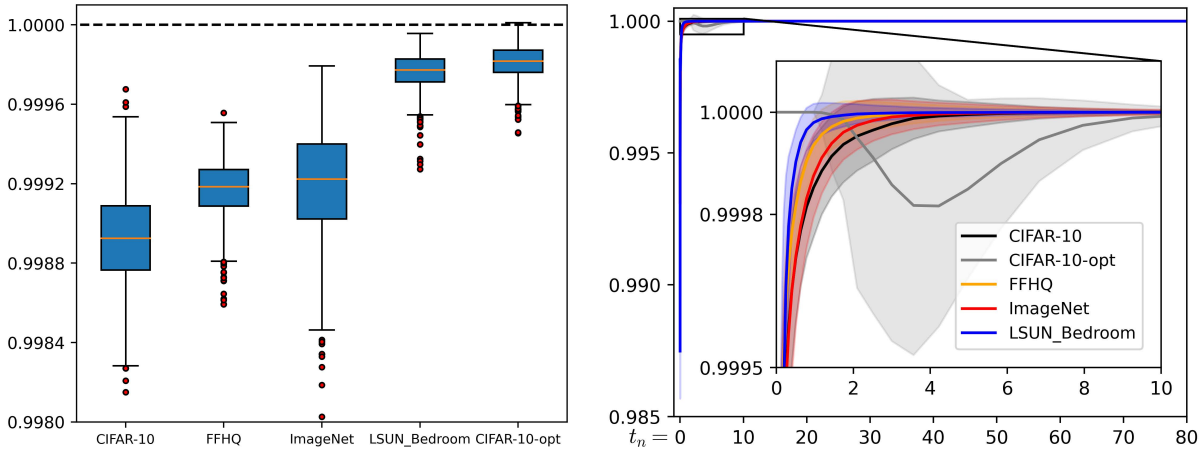


Figure 9. Left: The trajectory length is around $\sigma_T\sqrt{d}$ for both the optimal and actual diffusion models. *Right:* The cosine value between two consecutive Euler steps is very small, which indicates the magnitude of each oscillation is extremely small (around 0°).

Additionally, the variance of $\|\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta}}\|_2$ is relatively small.

$$\begin{aligned}
\mathrm{Var}_{p_t(\mathbf{x}_t)}\|\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta}}\|_2 &= \mathrm{Var}_{p_t(\mathbf{x}_t)}\|\mathbb{E}_{p_{t0}(\mathbf{x}_0|\mathbf{x}_t)}\boldsymbol{\epsilon}\|_2 = \mathbb{E}_{p_t(\mathbf{x}_t)}\|\mathbb{E}_{p_{t0}(\mathbf{x}_0|\mathbf{x}_t)}\boldsymbol{\epsilon}\|_2^2 - \left[\mathbb{E}_{p_t(\mathbf{x}_t)}\|\mathbb{E}_{p_{t0}(\mathbf{x}_0|\mathbf{x}_t)}\boldsymbol{\epsilon}\|_2\right]^2 \\
&\le \mathbb{E}_{p_t(\mathbf{x}_t)}\mathbb{E}_{p_{t0}(\mathbf{x}_0|\mathbf{x}_t)}\|\boldsymbol{\epsilon}\|_2^2 - (d-2m) = \mathbb{E}_{p_0(\mathbf{x}_0)}\mathbb{E}_{p_{0t}(\mathbf{x}_0|\mathbf{x}_t)}\|\boldsymbol{\epsilon}\|_2^2 - (d-2m) \\
&= d - (d-2m) \\
&= 2m
\end{aligned} \tag{49}$$

Therefore, the standard deviation of $\|\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta}}\|_2$ is upper bounded by $\sqrt{2m}$. Since $d \gg m$, we can conclude that in the optimal case, the magnitude of vector field is approximately constant, *i.e.*, $\|\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta}}\|_2 \approx \sqrt{d}$.

The total sampling trajectory length is $\sum_{n=0}^{N-1}(\sigma_{t_{n+1}} - \sigma_{t_n})\|\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta}}(\mathbf{x}_{t_{n+1}})\|_2 \approx \sigma_T\sqrt{d}$. Therefore, in the optimal case, we have $\|r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - \hat{\mathbf{x}}_{t_n}\|_2 = (\sigma_{t_n}/\sigma_{t_{n+1}})\|r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}}) - \hat{\mathbf{x}}_{t_{n+1}}\|_2 = \sigma_{t_n}\|\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})\|_2 \approx \sigma_{t_n}\|\boldsymbol{\epsilon}^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})\|_2 = \|r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) - \hat{\mathbf{x}}_{t_n}\|_2$. In this scenario, the denoising output $r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_{n+1}})$ appears to be oscillating toward $r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})$ around $\hat{\mathbf{x}}_{t_n}$, akin to a simple gravity pendulum (Young et al., 1996). The length of this pendulum effectively shortens by the coefficient $\sigma_{t_n}/\sigma_{t_{n+1}}$, starting from roughly $\sigma_T\sqrt{d}$. This specific structure is common to all sampling trajectories. Empirical verification of the constant magnitude of vector field is illustrated in Figure 8 and the trajectory length is illustrated in Figure 9.

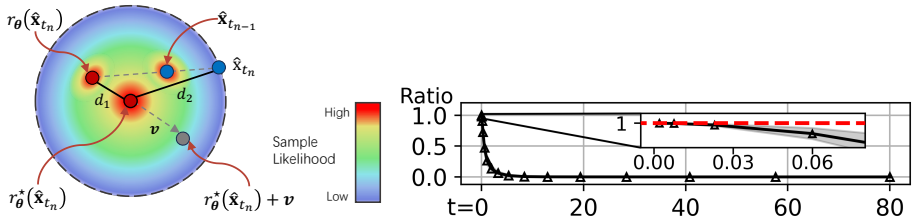


Figure 10. Left: We have three likelihood rankings in the ODE-based diffusion sampling: (1) $p_h(r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})) \geq p_h(\hat{\mathbf{x}}_{t_n})$, (2) $p_h(\hat{\mathbf{x}}_{t_{n-1}}) \geq p_h(\hat{\mathbf{x}}_{t_n})$, and (3) $p_h(r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})) \geq p_h(\hat{\mathbf{x}}_{t_n})$. This figure complements the trajectory structure shown in Figure 5. *Right:* We compute the ratios $d_1(\hat{\mathbf{x}}_{t_n})/d_2(\hat{\mathbf{x}}_{t_n})$ along 50k sampling trajectories on CIFAR-10 (black curve), and find that these ratios are consistently lower than one from $t_N = 80$ to $t_0 = 0.002$. This empirical evidence supports the validity of Assumption A.13 in practice.

A.3.4. Monotone Increase in Sample Likelihood

In this section, we characterize the local behavior of the sampling process of diffusion models. To simplify notations, we denote the deviation of denoising output from the optimal counterpart as $d_1(\hat{\mathbf{x}}_{t_n}) = \|r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) - r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})\|_2$ and the distance between the optimal denoising output and the current position as $d_2(\hat{\mathbf{x}}_{t_n}) = \|r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) - \hat{\mathbf{x}}_{t_n}\|_2$.

Assumption A.13. We assume $d_1(\hat{\mathbf{x}}_{t_n}) \leq d_2(\hat{\mathbf{x}}_{t_n})$ for all $\hat{\mathbf{x}}_{t_n}, n \in [1, N]$ in the sampling trajectory.

This assumption requires that our learned denoising output $r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})$ falls within a sphere centered at the optimal denoising output $r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})$ with a radius of $d_2(\hat{\mathbf{x}}_{t_n})$. This radius controls the maximum deviation of the learned denoising output and shrinks during the sampling process. In practice, the assumption is relatively easy to satisfy for a well-trained diffusion model. A visual illustration is provided in Figure 10.

Proposition A.14. *We have* $p_h(r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})) \geq p_h(\hat{\mathbf{x}}_{t_n})$ *and* $p_h(\hat{\mathbf{x}}_{t_{n-1}}) \geq p_h(\hat{\mathbf{x}}_{t_n})$ *in terms of the KDE* $p_h(\mathbf{x}) = \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}\mathcal{N}(\mathbf{x}; \mathbf{y}_i, h^2\mathbf{I})$ *with any positive bandwidth* h.

Proof. We first prove that given a random vector $\mathbf{v}$ falling within a sphere centered at the optimal denoising output $r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})$ with a radius of $\|r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) - \hat{\mathbf{x}}_{t_n}\|_2$, *i.e.*, $\|r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) - \hat{\mathbf{x}}_{t_n}\|_2 \geq \|\mathbf{v}\|_2$, the sample likelihood is non-decreasing from $\hat{\mathbf{x}}_{t_n}$ to $r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) + \mathbf{v}$, *i.e.*, $p_h(r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) + \mathbf{v}) \geq p_h(\hat{\mathbf{x}}_{t_n})$. Then, we provide two settings for $\mathbf{v}$ to finish the proof.

The increase of sample likelihood from $\hat{\mathbf{x}}_{t_n}$ to $r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) + \mathbf{v}$ in terms of $p_h(\mathbf{x})$ is

$$
\begin{aligned}
p_h(r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) + \mathbf{v}) - p_h(\hat{\mathbf{x}}_{t_n}) &= \frac{1}{|\mathcal{I}|}\sum_i \left[\mathcal{N}\left(r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) + \mathbf{v}; \mathbf{y}_i, h^2\mathbf{I}\right) - \mathcal{N}\left(\hat{\mathbf{x}}_{t_n}; \mathbf{y}_i, h^2\mathbf{I}\right)\right] \\
&\overset{\text{(i)}}{\geq} \frac{1}{2h^2|\mathcal{I}|}\sum_i \mathcal{N}\left(\hat{\mathbf{x}}_{t_n}; \mathbf{y}_i, h^2\mathbf{I}\right)\left[\|\hat{\mathbf{x}}_{t_n} - \mathbf{y}_i\|_2^2 - \|r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) + \mathbf{v} - \mathbf{y}_i\|_2^2\right] \\
&= \frac{1}{2h^2|\mathcal{I}|}\sum_i \mathcal{N}\left(\hat{\mathbf{x}}_{t_n}; \mathbf{y}_i, h^2\mathbf{I}\right)\left[\|\hat{\mathbf{x}}_{t_n}\|_2^2 - 2\hat{\mathbf{x}}_{t_n}^T\mathbf{y}_i - \|r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) + \mathbf{v}\|_2^2 + 2\left(r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) + \mathbf{v}\right)^T\mathbf{y}_i\right] \\
&\overset{\text{(ii)}}{=} \frac{1}{2h^2|\mathcal{I}|}\sum_i \mathcal{N}\left(\hat{\mathbf{x}}_{t_n}; \mathbf{y}_i, h^2\mathbf{I}\right)\left[\|\hat{\mathbf{x}}_{t_n}\|_2^2 - 2\hat{\mathbf{x}}_{t_n}^T r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) - \|r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) + \mathbf{v}\|_2^2 + 2\left(r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) + \mathbf{v}\right)^T r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})\right] \\
&= \frac{1}{2h^2|\mathcal{I}|}\sum_i \mathcal{N}\left(\hat{\mathbf{x}}_{t_n}; \mathbf{y}_i, h^2\mathbf{I}\right)\left[\|\hat{\mathbf{x}}_{t_n}\|_2^2 - 2\hat{\mathbf{x}}_{t_n}^T r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) + \|r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})\|_2^2 - \|\mathbf{v}\|_2^2\right] \\
&= \frac{1}{2h^2|\mathcal{I}|}\sum_i \mathcal{N}\left(\hat{\mathbf{x}}_{t_n}; \mathbf{y}_i, h^2\mathbf{I}\right)\left[\|r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) - \hat{\mathbf{x}}_{t_n}\|_2^2 - \|\mathbf{v}\|_2^2\right] \geq 0,
\end{aligned}
\tag{50}
$$

where (i) uses the definition of convex function $f(\mathbf{x}_2) \geq f(\mathbf{x}_1) + f'(\mathbf{x}_1)(\mathbf{x}_2 - \mathbf{x}_1)$ with $f(\mathbf{x}) = \exp\left(-\frac{1}{2}\|\mathbf{x}\|_2^2\right)$, $\mathbf{x}_1 = \left(\hat{\mathbf{x}}_{t_n} - \mathbf{y}_i\right)/h$ and $\mathbf{x}_2 = \left(r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) + \mathbf{v} - \mathbf{y}_i\right)/h$; (ii) uses the relationship between two consecutive steps $\hat{\mathbf{x}}_{t_n}$ and $r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})$ in mean shift with the Gaussian kernel (see (43))

$$
r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) = \mathbf{m}(\hat{\mathbf{x}}_{t_n}) = \sum_i \frac{\exp\left(-\|\hat{\mathbf{x}}_{t_n} - \mathbf{y}_i\|_2^2/2h^2\right)}{\sum_j \exp\left(-\|\hat{\mathbf{x}}_{t_n} - \mathbf{y}_j\|_2^2/2h^2\right)}\mathbf{y}_i, \tag{51}
$$

which implies the following equation also holds

$$\sum_i \mathcal{N}\left(\hat{\mathbf{x}}_{t_n}; \mathbf{y}_i, h^2\mathbf{I}\right) \mathbf{x}_i = \sum_i \mathcal{N}\left(\hat{\mathbf{x}}_{t_n}; \mathbf{y}_i, h^2\mathbf{I}\right) r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}). \tag{52}$$

Since $\|r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) - \hat{\mathbf{x}}_{t_n}\|_2 \geq \|\mathbf{v}\|_2$, or equivalently, $\|r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) - \hat{\mathbf{x}}_{t_n}\|_2^2 \geq \|\mathbf{v}\|_2^2$, we conclude that the sample likelihood monotonically increases from $\hat{\mathbf{x}}_{t_n}$ to $r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) + \mathbf{v}$ unless $\hat{\mathbf{x}}_{t_n} = r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) + \mathbf{v}$, in terms of the kernel density estimate $p_h(\mathbf{x}) = \frac{1}{|\mathcal{I}|}\sum_i \mathcal{N}(\mathbf{x}; \mathbf{y}_i, h^2\mathbf{I})$ with any positive bandwidth h.

We next provide two settings for $\mathbf{v}$, which trivially satisfy the condition $\|r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) - \hat{\mathbf{x}}_{t_n}\|_2 \geq \|\mathbf{v}\|_2$, and have the following corollaries:

- $p_h(r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})) \geq p_h(\hat{\mathbf{x}}_{t_n})$, when $\mathbf{v} = r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) - r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})$.
- $p_h(\hat{\mathbf{x}}_{t_{n-1}}) \geq p_h(\hat{\mathbf{x}}_{t_n})$, when $\mathbf{v} = r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) - r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) + \frac{t_{n-1}}{t_n}\left(\hat{\mathbf{x}}_{t_n} - r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})\right)$.

□

Therefore, each sampling trajectory monotonically converges ($p_h(\hat{\mathbf{x}}_{t_{n-1}}) \geq p_h(\hat{\mathbf{x}}_{t_n})$), and its coupled denoising trajectory converges even faster ($p_h(r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n})) \geq p_h(\hat{\mathbf{x}}_{t_n})$) in terms of the sample likelihood. Given an empirical data distribution, Proposition A.14 applies to any marginal distributions of our forward SDE $\{p_t(\mathbf{x})\}_{t=0}^T$, which should include the optimal bandwidth for the considered dataset.

We can also obtain the standard monotone convergence property of mean shift (Comaniciu & Meer, 2002) from Proposition A.14 when diffusion models are trained to achieve the optimal parameters.

Corollary A.15. *We have $p_h(\mathbf{m}(\hat{\mathbf{x}}_{t_n})) \geq p_h(\hat{\mathbf{x}}_{t_n})$, when $r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) = r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}) = \mathbf{m}(\hat{\mathbf{x}}_{t_n})$.*

This connection also implies that once a diffusion model has converged to the optimum, all ODE trajectories will be uniquely determined and governed by a bandwidth-varying mean shift. In this case, the forward (encoding) process and backward (decoding) process only depend on the data distribution and the given noise distribution, regardless of model architectures or optimization algorithms. Such a property was previously referred to as *uniquely identifiable encoding* and empirically verified in (Song et al., 2021c), while we theoretically characterize the optimum with annealed mean shift, and thus reveal the asymptotic behavior of diffusion models.

Besides, the optimal diffusion models simply memorize the dataset and replay a certain discrete data point in sampling. We argue that in practice, a slight score deviation from the optimum ensures the generative ability of diffusion models while greatly alleviating the mode collapse issue. Experimental results are provided in the next section.

A.3.5. Diagnosis of Score Deviation

We simulate four new trajectories based on the optimal denoising output $r^\star_{\boldsymbol{\theta}}(\cdot)$ to monitor the score deviation from the optimum. The first one is *optimal sampling trajectory* $\{\hat{\mathbf{x}}^\star_t\}$, where we generate samples as the sampling trajectory $\{\hat{\mathbf{x}}_t\}$ by simulating (5) but adopt $r^\star_{\boldsymbol{\theta}}(\cdot)$ rather than $r_{\boldsymbol{\theta}}(\cdot)$ for score estimation. The other three trajectories are simulated by tracking the (optimal) denoising output of each sample in $\{\hat{\mathbf{x}}^\star_t\}$ or $\{\hat{\mathbf{x}}_t\}$, and designated as $\{r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^\star_t)\}$, $\{r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^\star_t)\}$, $\{r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_t)\}$. According to (6) and $t_0 = 0$, we have $\hat{\mathbf{x}}^\star_{t_0} = r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^\star_{t_1})$, and similarly, $\hat{\mathbf{x}}_{t_0} = r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_1})$. As $t \to 0$, $r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^\star_t)$ and $r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_t)$ serve as the approximate nearest neighbors of $\hat{\mathbf{x}}^\star_t$ and $\hat{\mathbf{x}}_t$ to the real data, respectively.

We calculate the deviation of denoising output to quantify the score deviation across all time steps using the L^2 distance, though they should differ by a factor t^2, and have the following observation: *The learned score is well-matched to the optimal score in the large-noise region, otherwise they may diverge or almost coincide depending on different regions.* In fact, our learned score has to moderately diverge from the optimum to guarantee the generative ability. Otherwise, the ODE-based sampling reduces to an approximate (single-step) annealed mean shift for global mode-seeking (see Section A.3.2), and simply replays the dataset. As shown in Figure 11, the nearest sample of $\hat{\mathbf{x}}^\star_{t_0}$ to the real data is almost the same as itself, which indicates the optimal sampling trajectory has a very limited ability to synthesize novel samples. Empirically, score deviation in a small region is sufficient to bring forth a decent generative ability.

From the comparison of $\{r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^\star_t)\}$, $\{r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^\star_t)\}$ sequences in Figures 11 and 12, we can clearly see that *along the optimal sampling trajectory*, the deviation between the learned denoising output $r_{\boldsymbol{\theta}}(\cdot)$ and its optimal counterpart $r^\star_{\boldsymbol{\theta}}(\cdot)$ behaves

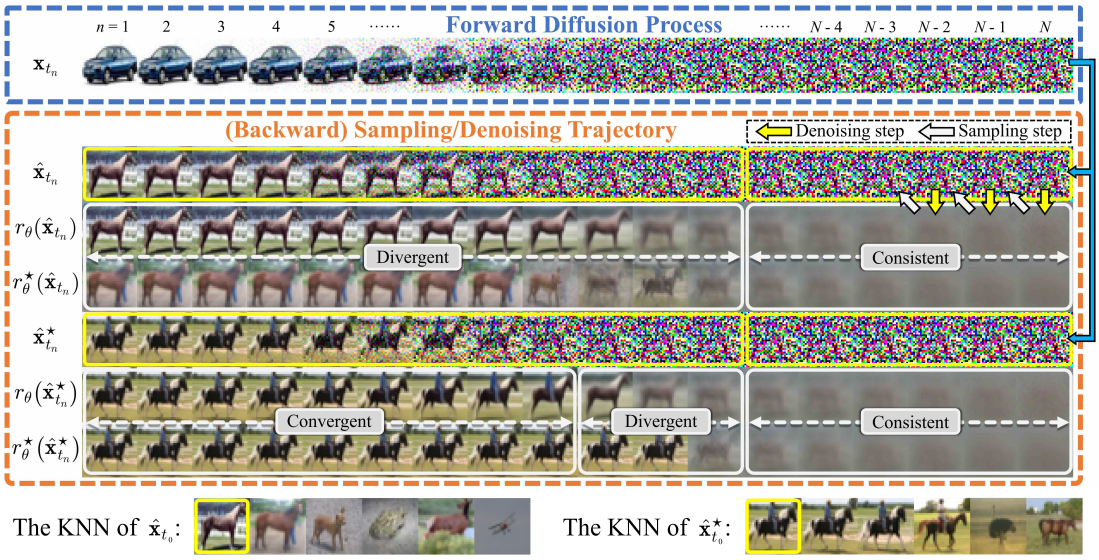


Figure 11. Top: We visualize a forward diffusion process of a randomly-selected image to obtain its encoding $\hat{\mathbf{x}}_{t_N}$ (first row) and simulate multiple trajectories starting from this encoding (other rows). *Bottom*: The k-nearest neighbors (k=5) of $\hat{\mathbf{x}}_{t_0}$ and $\hat{\mathbf{x}}^\star_{t_0}$ to real samples in the dataset.

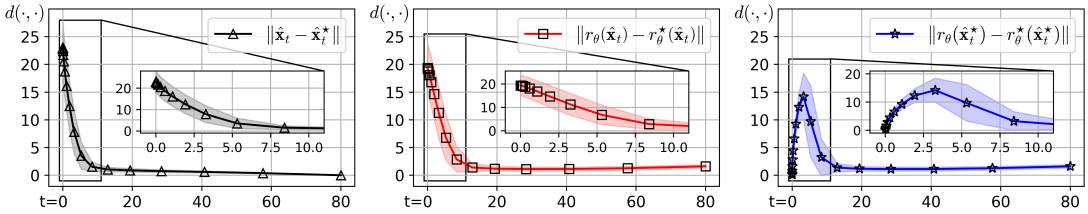


Figure 12. The deviation (measured by L^2 distance) of outputs from their corresponding optima.

differently in three successive regions: the deviation starts off as almost negligible (about $10 < t \leq 80$), gradually increases (about $3 < t \leq 10$), and then drops down to a low level once again (about $0 \leq t \leq 3$). This phenomenon was also validated by a recent work (Xu et al., 2023) with a different perspective. We further observe that *along the sampling trajectory*, this phenomenon disappears and the score deviation keeps increasing (see $\{r_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_t)\}$, $\{r^\star_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_t)\}$ sequences in Figures 11 and 12). Additionally, samples in the latter half of $\{r^\star_\theta(\hat{\mathbf{x}}_t)\}$ appear almost the same as the nearest sample of $\hat{\mathbf{x}}_{t_0}$ to the real data, as shown in Figure 11. This indicates that our score-based model strives to explore novel regions, and synthetic samples in the sampling trajectory are quickly attracted to a real-data mode but do not fall into it.

A.4. Additionally Experimental Results

A.4.1. Dynamic Programming

In this paper, as a simple illustration, we formulated the seeking of an optimal time schedule as an integer programming problem and solved it with the standard dynamic programming (Cormen et al., 2022), as shown in Algorithm 1. We also tried the Branch and Bound Algorithm and obtained similar results. Still, there also exist many different ways to determine the time schedule (e.g., using a trainable neural network) by leveraging our discovered trajectory regularity.

Specifically, we estimate the local truncation errors of PF-ODE based sampling trajectories, and compute the cost matrix for dynamic programming. In the sampling process of diffusion models, the global truncation error does not equal to the accumulation of local truncation error in each step. We thus introduce a coefficient γ to compensate this effect.

Discussion. Watson et al. (2021) was the first one leveraging the idea of dynamic programming to re-allocate the time schedule in diffusion models. However, our motivation behind using DP significantly differs from this previous work.

Algorithm 1 Geometry-Inspired Time Scheduling (standard dynamic programming, similar to (Watson et al., 2021))

Input: the number of teacher NFE N_t for fine-grained sampling, the number of student NFE N_s (NFE budget), the coefficient γ, a cost matrix with $c_{jk} \coloneqq \mathcal{D}(\hat{\mathbf{x}}_{t_{N-j} \to t_{N-k}}, \mathbf{x}_{t_{N-j} \to t_{N-k}})$, $0 \leq j < k \leq N_t$.
Initialize $V_{ij} = +\infty$ $(0 \leq i \leq N_s, 0 \leq j \leq N_t)$, time schedule $\Gamma = [0]$, $m = 0$
$V_{i1} = c_{iN}$ $(0 \leq i \leq N_s)$
for $k = 2$ **to** N_s **do**
  for $j = 0$ **to** $N_t - 1$ **do**
    for $i = j + 1$ **to** $N_t - 1$ **do**
      $V_{jk} = \min\{V_{jk},\ \gamma c_{ji} + V_{ik-1}\}$
    end for
  end for
end for
# Fetch shortest path of N_s steps
for $k = N_s$ **to** 1 **do**
  for $j = m + 1$ **to** N_t **do**
    if $V_{mk} == \gamma c_{mj} + V_{jk-1}$ **then**
      Γ.append(j)
      $m = j$
      break
    end if
  end for
end for
Γ.append(N_t)
Return: Γ

Table 6. Ablation study on the "warmup" sample size with iPNDM+GITS on CIFAR-10. † denotes that we search for a unique time schedule for each sample of 50k generated samples. This special case is more time-consuming while achieving similar results (due to the strong trajectory regularity).

NFE	SAMPLE SIZE							
	1†	16	64	128	**256**	512	1024	2048
5	9.25	9.55±0.75	9.57±0.97	9.21 ±0.44	8.84±0.30	8.81±0.04	8.89±0.11	8.88±0.12
6	5.12	5.36±0.61	5.16±0.28	4.99±0.18	5.03±0.25	5.20±0.27	5.01±0.19	4.92±0.08
8	3.13	3.25±0.13	3.22±0.08	3.28±0.10	3.27±0.11	3.30±0.11	3.29±0.08	3.33±0.10
10	2.41	2.46±0.11	2.46±0.05	2.45±0.05	2.46±0.04	2.45±0.04	2.44±0.05	2.44±0.05

Table 7. Used time (**seconds**) under different "warmup" sample sizes (iPNDM+GITS on CIFAR-10). "warmup" samples are generated by 60 NFE and the NFE budge for dynamic programming is 10.

SAMPLE SIZE	sample generation	cost matrix	dynamic programming	total time (s)
16	5.68	0.73	0.015	6.42
64	8.59	1.08	0.015	9.68
128	13.22	2.61	0.015	15.84
256 (default)	27.47	5.29	0.015	32.78
512	40.20	14.16	0.015	54.46
1024	75.90	29.58	0.015	105.58
2048	149.12	40.83	0.015	189.96

Watson et al. (2021) exploit the fact that ELBO can be decomposed into separate KL terms and utilize DP to find the optimal discrete-time schedule that maximizes the training ELBO, but this strategy even worsens the sample quality, as admitted by authors. In contrast, we first found a strong trajectory regularity shared by all sampling trajectories (Section 3), and then used some "warmup" samples to estimate the trajectory curvature to determine a better time schedule for the sampling of diffusion models.

More results. We provide ablation studies on analytic first step (AFS) and sensitivity analysis of the coefficient γ in Table 8.

Table 8. Sample quality in terms of Fréchet Inception Distance (FID (Heusel et al., 2017), lower is better) on four datasets (resolutions ranging from 32×32 to 256×256). †: After obtaining the DP schedule, we could further optimize the first time step with AFS, using the same 256 "warmup" samples as generating the fine-grained (teacher) sampling trajectory. *The default setting in our main submission dose not use AFS and keeps the coefficient in dynamic programming as 1.1 for LSUN Bedroom and 1.15 otherwise. Although as shown in the Table, the performance can be further improved by carefully tuning the coefficient and using AFS.*

METHOD	Coeff	AFS†	NFE							
			3	4	5	6	7	8	9	10
CIFAR-10 32×32 (Krizhevsky & Hinton, 2009)										
DDIM (Song et al., 2021a)	-	✗	93.36	66.76	49.66	35.62	27.93	22.32	18.43	15.69
DDIM + GITS	1.10	✗	88.68	46.88	32.50	22.04	16.76	13.93	11.57	10.09
DDIM + GITS (default)	1.15	✗	79.67	43.07	28.05	21.04	16.35	13.30	11.62	10.37
DDIM + GITS	1.20	✗	77.22	43.16	29.06	22.69	18.91	14.22	12.03	11.38
iPNDM (Zhang & Chen, 2023)	-	✗	47.98	24.82	13.59	7.05	5.08	3.69	3.17	2.77
iPNDM + GITS	1.10	✗	51.31	17.19	12.90	5.98	6.62	4.36	3.59	3.14
iPNDM + GITS (default)	1.15	✗	43.89	15.10	8.38	4.88	5.11	3.24	2.70	2.49
iPNDM + GITS	1.20	✗	42.06	15.85	9.33	7.13	5.95	3.28	2.81	2.71
iPNDM + GITS	1.10	✓	34.22	11.99	12.44	6.08	6.20	3.53	3.48	2.91
iPNDM + GITS	1.15	✓	29.63	11.23	8.08	4.86	4.46	2.92	2.46	**2.27**
iPNDM + GITS	1.20	✓	**25.98**	**10.11**	**6.77**	**4.29**	**3.43**	**2.70**	**2.42**	2.28
FFHQ 64×64 (Karras et al., 2019)										
DDIM (Song et al., 2021a)	-	✗	78.21	57.48	43.93	35.22	28.86	24.39	21.01	18.37
DDIM + GITS	1.10	✗	62.70	43.12	31.01	24.62	20.35	17.19	14.71	13.01
DDIM + GITS (default)	1.15	✗	60.84	40.81	29.80	23.67	19.41	16.60	14.46	13.06
DDIM + GITS	1.20	✗	59.64	40.56	30.29	23.88	20.07	17.36	15.40	14.05
iPNDM (Zhang & Chen, 2023)	-	✗	45.98	28.29	17.17	10.03	7.79	5.52	4.58	3.98
iPNDM + GITS	1.10	✗	34.82	18.75	13.07	7.79	8.30	4.76	5.36	3.47
iPNDM + GITS (default)	1.15	✗	33.09	17.04	11.22	7.00	6.72	4.52	4.33	3.62
iPNDM + GITS	1.20	✗	31.70	16.87	10.83	7.10	6.37	5.78	4.81	4.39
iPNDM + GITS	1.10	✓	33.19	19.88	12.90	8.29	7.50	4.26	4.95	**3.13**
iPNDM + GITS	1.15	✓	30.39	15.78	10.15	6.86	5.97	**4.09**	**3.76**	3.24
iPNDM + GITS	1.20	✓	**26.41**	**13.59**	**8.85**	**6.39**	**5.36**	4.91	3.89	3.51
ImageNet 64×64 (Russakovsky et al., 2015)										
DDIM (Song et al., 2021a)	-	✗	82.96	58.43	43.81	34.03	27.46	22.59	19.27	16.72
DDIM + GITS	1.10	✗	60.11	36.23	27.31	20.82	16.41	14.16	11.95	10.84
DDIM + GITS (default)	1.15	✗	57.06	35.07	24.92	19.54	16.01	13.79	12.17	10.83
DDIM + GITS	1.20	✗	54.24	34.27	24.67	19.46	16.66	14.15	13.41	11.87
iPNDM (Zhang & Chen, 2023)	-	✗	58.53	33.79	18.99	12.92	9.17	7.20	5.91	5.11
iPNDM + GITS	1.10	✗	36.18	19.64	13.18	9.58	7.68	6.44	5.24	4.59
iPNDM + GITS (default)	1.15	✗	34.47	18.95	10.79	8.43	6.83	5.82	4.96	4.48
iPNDM + GITS	1.20	✗	32.70	18.59	11.04	9.23	7.18	6.20	5.50	5.08
iPNDM + GITS	1.10	✓	31.50	21.50	13.73	10.74	7.99	6.88	5.29	4.64
iPNDM + GITS	1.15	✓	28.01	18.28	10.28	8.68	6.76	5.90	4.81	**4.40**
iPNDM + GITS	1.20	✓	**26.41**	**16.41**	**9.85**	**8.39**	**6.44**	**5.64**	**4.79**	4.47
LSUN Bedroom 256×256 (Yu et al., 2015) (pixel-space)										
DDIM (Song et al., 2021a)	-	✗	86.13	54.45	34.34	25.25	19.49	15.71	13.26	11.42
DDIM + GITS	1.05	✗	81.77	36.89	27.46	18.78	13.60	12.23	10.29	8.77
DDIM + GITS (default)	1.10	✗	61.85	35.12	22.04	16.54	13.58	11.20	9.82	9.04
DDIM + GITS	1.15	✗	60.11	31.02	23.65	17.18	13.42	12.61	10.89	10.57
iPNDM (Zhang & Chen, 2023)	-	✗	80.99	43.90	26.65	20.73	13.80	11.78	8.38	5.57
iPNDM + GITS	1.05	✗	59.02	24.71	19.08	12.77	**8.19**	**6.67**	**5.58**	**4.83**
iPNDM + GITS (default)	1.10	✗	45.75	22.98	**15.85**	**10.41**	8.63	7.31	6.01	5.28
iPNDM + GITS	1.15	✗	**44.78**	**21.67**	17.29	11.52	9.59	8.82	7.22	5.97

We provide ablation studies on the grid size for generating the fine-grained sampling trajectory in Table 9, and the number of "warmup" sample size and used time in Table 6 and Table 7.

Time schedule. The uniform schedule is widely used in papers using a DDPM (Ho et al., 2020) backbone. Following the implementation in EDM (Karras et al., 2022), we transfer this schedule from its original range $[\epsilon_s, 1]$ to $[t_0, t_N]$ where $\epsilon_s = 0.001$, $t_0 = 0.002$ and $t_N = 80$. We first uniformly sample τ_n ($n \in [0, N]$) from $[\epsilon_s, 1]$ and then calculate t_n by

$$t_n = \sqrt{e^{\frac{1}{2}\beta_d {\tau_n}^2 + \beta_{\min}\tau_n} - 1} \tag{53}$$

where

$$\beta_d = \frac{2}{\epsilon_s - 1}\frac{\log(1 + {t_0}^2)}{\epsilon_s} - \log(1 + {t_N}^2), \quad \beta_{\min} = \log(1 + {t_N}^2) - \frac{1}{2}\beta_d. \tag{54}$$

Table 9. Ablation study on the grid size used for the fine-grained sampling on CIFAR-10 with iPNDM. The DP coefficient is kept as 1.15.

GRID SIZE	NFE BUDGET						
	4	5	6	7	8	9	10
11	20.88	10.15	5.11	4.63	3.16	2.78	2.77
21	16.22	9.87	4.83	3.76	3.39	3.20	2.81
41	15.34	9.34	4.83	5.54	3.01	2.66	2.53
61 (default)	15.10	8.38	4.88	5.11	3.24	2.70	2.49
81	15.74	8.57	5.09	5.38	3.10	2.93	2.38
101	15.03	8.72	5.02	5.19	3.12	2.81	2.41
iPNDM	24.82	13.59	7.05	5.08	3.69	3.17	2.77

Table 10. Time schedule on CIFAR-10 found by the dynamic programming. See the formula in texts for more details.

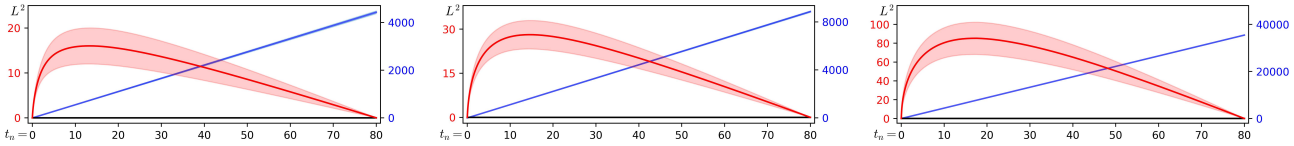
NFE	TIME SCHEDULE	FID
Uniform		
3	[80.0000, 6.9503, 1.2867, 0.0020]	50.44
4	[80.0000, 11.7343, 2.8237, 0.8565, 0.0020]	18.73
5	[80.0000, 16.5063, 4.7464, 1.7541, 0.6502, 0.0020]	17.34
6	[80.0000, 20.9656, 6.9503, 2.8237, 1.2867, 0.5272, 0.0020]	9.75
7	[80.0000, 25.0154, 9.3124, 4.0679, 2.0043, 1.0249, 0.4447, 0.0020]	12.50
8	[80.0000, 28.6496, 11.7343, 5.4561, 2.8237, 1.5621, 0.8565, 0.3852, 0.0020]	7.56
9	[80.0000, 31.8981, 14.1472, 6.9503, 3.7419, 2.1599, 1.2867, 0.7382, 0.3401, 0.0020]	10.60
10	[80.0000, 34.8018, 16.5063, 8.5141, 4.7464, 2.8237, 1.7541, 1.0985, 0.6502, 0.3047, 0.0020]	7.35
LogSNR		
3	[80.0000, 2.3392, 0.0684, 0.0020]	88.38
4	[80.0000, 5.6569, 0.4000, 0.0283, 0.0020]	35.59
5	[80.0000, 9.6090, 1.1542, 0.1386, 0.0167, 0.0020]	19.87
6	[80.0000, 13.6798, 2.3392, 0.4000, 0.0684, 0.0117, 0.0020]	10.68
7	[80.0000, 17.6057, 3.8745, 0.8527, 0.1876, 0.0413, 0.0091, 0.0020]	6.56
8	[80.0000, 21.2732, 5.6569, 1.5042, 0.4000, 0.1064, 0.0283, 0.0075, 0.0020]	4.74
9	[80.0000, 24.6462, 7.5929, 2.3392, 0.7207, 0.2220, 0.0684, 0.0211, 0.0065, 0.0020]	3.53
10	[80.0000, 27.7258, 9.6090, 3.3302, 1.1542, 0.4000, 0.1386, 0.0480, 0.0167, 0.0058, 0.0020]	2.94
Polynomial ($\rho = 7$)		
3	[80.0000, 9.7232, 0.4700, 0.0020]	47.98
4	[80.0000, 17.5278, 2.5152, 0.1698, 0.0020]	24.82
5	[80.0000, 24.4083, 5.8389, 0.9654, 0.0851, 0.0020]	13.59
6	[80.0000, 30.1833, 9.7232, 2.5152, 0.4700, 0.0515, 0.0020]	7.05
7	[80.0000, 34.9922, 13.6986, 4.6371, 1.2866, 0.2675, 0.0352, 0.0020]	5.08
8	[80.0000, 39.0167, 17.5278, 7.1005, 2.5152, 0.7434, 0.1698, 0.0261, 0.0020]	3.69
9	[80.0000, 42.4152, 21.1087, 9.7232, 4.0661, 1.5017, 0.4700, 0.1166, 0.0204, 0.0020]	3.17
10	[80.0000, 45.3137, 24.4083,12.3816, 5.8389, 2.5152, 0.9654, 0.3183, 0.0851, 0.0167, 0.0020]	2.77
GITS (ours)		
3	[80.0000, 3.8811, 0.9654, 0.0020]	43.89
4	[80.0000, 5.8389, 1.8543, 0.4700, 0.0020]	15.10
5	[80.0000, 6.6563, 2.1632, 0.8119, 0.2107, 0.0020]	8.38
6	[80.0000, 10.9836, 3.8811, 1.5840, 0.5666, 0.1698, 0.0020]	4.88
7	[80.0000, 12.3816, 3.8811, 1.5840, 0.5666, 0.1698, 0.0395, 0.0020]	3.76
8	[80.0000, 10.9836, 3.8811, 1.8543, 0.9654, 0.4700, 0.2107, 0.0665, 0.0020]	3.24
9	[80.0000, 12.3816, 4.4590, 2.1632, 1.1431, 0.5666, 0.2597, 0.1079, 0.0300, 0.0020]	2.70
10	[80.0000, 12.3816, 4.4590, 2.1632, 1.1431, 0.5666, 0.3183, 0.1698, 0.0665, 0.0225, 0.0020]	2.49

The losSNR time schedule is proposed for fast sampling in DPM-Solver (Lu et al., 2022a). We first uniformly sample λ_n ($n \in [0, N]$) from $[\lambda_{\min}, \lambda_{\max}]$ where $\lambda_{\min} = -\log t_N$ and $\lambda_{\max} = -\log t_0$. The logSNR schedule is given by $t_n = e^{-\lambda_n}$.

The polynomial time schedule $t_n = (t_0^{1/\rho} + \frac{n}{N}(t_N^{1/\rho} - t_0^{1/\rho}))^\rho$ is proposed in EDM (Karras et al., 2022), where $t_0 = 0.002$, $t_N = 80$, $n \in [0, N]$, and $\rho = 7$.

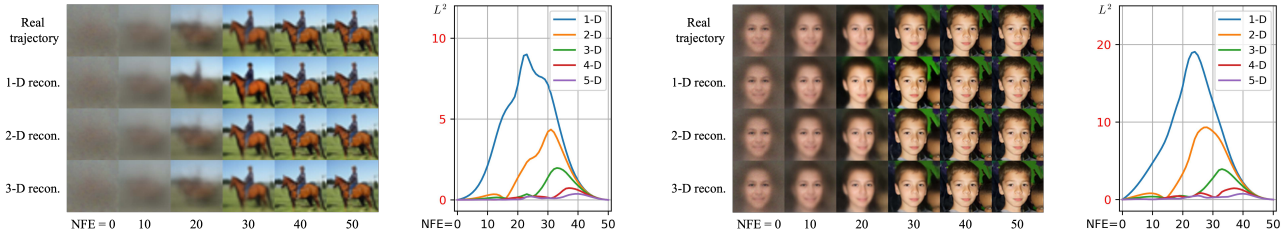
The optimized time schedules for stable diffusion 1.5 in Figure 7 include

- AYS (Sabour et al., 2024): [999, 850, 736, 645, 545, 455, 343, 233, 124, 24, 0].
- GITS: [999, 783, 632, 483, 350, 233, 133, 67, 33, 17, 0].



(a) CIFAR-10 32×32. (b) FFHQ 64×64. (c) LSUN Bedroom 256×256.

Figure 13. Trajectory deviation (red curve) compared to the sample distance (blue curve) in the sampling process starting from $t_N = 80$ to $t_0 = 0.002$. Each trajectory is simulated with Euler method and 100 NFE. The results are averaged by 5k generated samples.



(a) Visual Comparison. (b) L^2 error. (c) Visual Comparison. (d) L^2 error.

Figure 14. (a/c) The visual comparison of trajectory reconstruction on CIFAR-10, and FFHQ. We reconstruct the real sampling trajectory (top row) using $\hat{\mathbf{x}}_{t_N} - \hat{\mathbf{x}}_{t_0}$ (1-d recon.) along with its top 1 or 2 principal components (2-D or 3-D recon.). To amplify the visual difference, we present the denoising outputs of these trajectories. (b/d) We compute the L^2 distance between the real trajectory the reconstructed trajectories up to 5-D reconstruction.

(a) DDIM, NFE = 5, FID = 49.66. (b) DDIM + GITS, NFE = 5, FID = 28.05.

(c) iPNDM, NFE = 5, FID = 13.59. (d) iPNDM + GITS, NFE = 5, FID = 8.38.

Figure 15. Synthesized samples on CIFAR-10 with DDIM (+ GITS) and iPNDM (+ GITS).

A.4.2. Regularity of Sampling Trajectory

Figure 13 provides more experiments about 1-D projections on CIFAR-10, FFHQ, and LSUN Bedroom. Figure 14 provides more experiments about Multi-D projections on CIFAR-10 and FFHQ. Figures 15, 16, 17, and 18 visualize more generated samples on four datasets.

(a) DDIM, NFE = 5, FID = 43.93.

(b) DDIM + GITS, NFE = 5, FID = 29.80.

(c) iPNDM, NFE = 5, FID = 17.17.

(d) iPNDM + GITS, NFE = 5, FID = 11.22.

Figure 16. Synthesized samples on FFHQ 64×64 with DDIM (+ GITS) and iPNDM (+ GITS).

a DDIM, NFE = 5, FID = 43.81.

b DDIM + GITS, NFE = 5, FID = 24.92.

c iPNDM, NFE = 5, FID = 18.99.

d iPNDM + GITS, NFE = 5, FID = 10.79.

Figure 17. Synthesized samples on ImageNet 64×64 with DDIM (+ GITS) and iPNDM (+ GITS).

(a) DDIM, NFE = 5, FID = 34.34.

(b) DDIM + GITS, NFE = 5, FID = 22.04.

(c) iPNDM, NFE = 5, FID = 26.65.

(d) iPNDM + GITS, NFE = 5, FID = 15.85.

Figure 18. Synthesized samples on LSUN Bedroom 256×256 (pixel-space) with DDIM (+ GITS) and iPNDM (+ GITS).