

Diffusion Models Generate Images Like Painters: an Analytical Theory of Outline First, Details Later

Binxu Wang
 Kempner Institute
 Harvard University
 Boston, MA
 binxu_wang@hms.harvard.edu

John J. Vastola
 Department of Neurobiology
 Harvard Medical School
 Boston, MA
 John_Vastola@hms.harvard.edu

Abstract

How do diffusion generative models convert pure noise into meaningful images? In a variety of pretrained diffusion models (including conditional latent space models like Stable Diffusion), we observe that the reverse diffusion process that underlies image generation has the following properties: (i) individual trajectories tend to be low-dimensional and resemble 2D ‘rotations’; (ii) high-variance scene features like layout tend to emerge earlier, while low-variance details tend to emerge later; and (iii) early perturbations tend to have a greater impact on image content than later perturbations. To understand these phenomena, we derive and study a closed-form solution to the probability flow ODE for a Gaussian distribution, which shows that the reverse diffusion state rotates towards a gradually-specified target on the image manifold. It also shows that generation involves first committing to an outline, and then to finer and finer details. We find that this solution accurately describes the initial phase of image generation for pretrained models, and can in principle be used to make image generation more efficient by skipping reverse diffusion steps. Finally, we use our solution to characterize the image manifold in Stable Diffusion. Our viewpoint reveals an unexpected similarity between generation by GANs and diffusion and provides a conceptual link between diffusion and image retrieval.

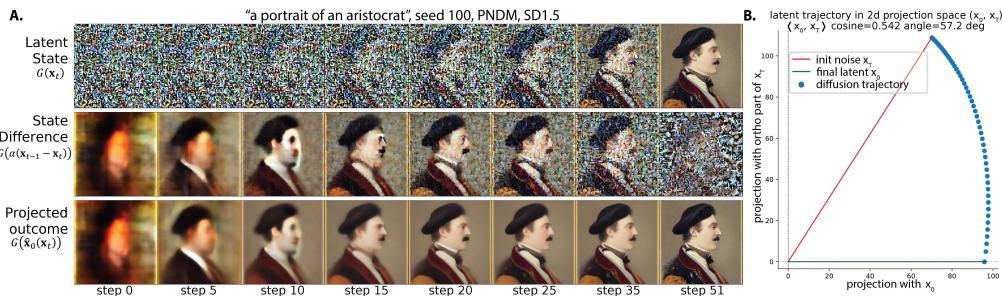


Figure 1: **Characteristics of image generation by diffusion models.** **A.** Tracking latent states $G(\mathbf{x}_t)$ (top row), differences between nearby time steps $G(k(\mathbf{x}_{t-1} - \mathbf{x}_t))$ (middle row), and final image estimates $G(\hat{\mathbf{x}}_0(\mathbf{x}_t))$ (bottom row) suggests different measures of progress. **B.** Individual trajectories are effectively two-dimensional, with the transition from \mathbf{x}_T to \mathbf{x}_0 being rotation-like.

1 Introduction

Imagine an artist painting a picture of a natural landscape. We generally expect higher-level scene elements to appear first, and lower-level details later: the borders of the land and sky might be drawn, then the largest objects (like mountains and trees) might be placed, then minor objects (like rocks and

small animals) might be placed, and finally fine details (like textures and shading) might be filled in. Do diffusion generative models [1–3], which can generate natural landscapes like those of an artist, also construct images like this? If not, how do they work? Specifically, how do they ‘determine’ what to generate from the noise?

By visualizing images throughout a reverse diffusion trajectory—from pure noise to the final image—the naive answer appears to be no. One gets the impression of an image emerging fully-formed from the noise; one is ‘uncovering’ the image, or ‘opening one’s eyes’ to reveal an image that was always there. But visualizing an endpoint estimate of the reverse diffusion indicates that large-scale image features emerge before details (see e.g. Fig. 6 of [4] and Fig. 1 of [5]), which suggests that the naive view is misleading. Hertz et al. reach a similar conclusion from studying conditional diffusion models’ cross-attention maps, finding that different parts of an image may be ‘attended to’ at different times [6], and that coarse features (e.g. of a bear, see their Figure 4) are attended to before details.

Our aim in this work is to explore the apparent outline-first, details-later behavior of reverse diffusion in quantitative detail, using a mix of simple theory and numerical experiments on pre-trained diffusion models. Our theory and experiments together support the following claims about diffusion model image generation: (i) individual reverse diffusion trajectories tend to be very low-dimensional; (ii) scene elements that vary more within training data tend to emerge earlier; and (iii) early perturbations substantially change image content more often than late perturbations.

Our major contribution is to provide a closed-form solution to the sampling trajectory of probability flow ODE. This solution can qualitatively explain the generation behavior of pre-trained diffusion models and quantitatively predict their sampling trajectory in the early phase, just given knowledge of the mean and covariance of the training data. Practically, our result can be leveraged to accelerate sampling by skipping the early phase entirely, and it can also be used to characterize the image manifold embedded in the diffusion models. Finally, by deriving the sampling trajectory for the exact score of training data, we draw connections between diffusion models and image retrieval process.

Conceptually, the viewpoint we develop sheds light on the geometry of diffusion models, and in particular on the difficult-to-identify low-dimensional manifold that smoothly parameterizes generated images [7]. We identify some interesting parallels with the geometry of the analogous manifold for generative adversarial networks (GANs) [8].

2 Diffusion generative modeling basics

There are several complementary theoretical frameworks of diffusion generative modeling [1, 4, 2, 9]. Guided by the unifying view of [5], in this work, we focus on the continuous-time framework of Song et al. [3]. Diffusion generative models involve mapping a data distribution $p(\mathbf{x})$ to a simpler one $p(\mathbf{x}_T)$ via a stochastic process—typically pure diffusion or an Ornstein-Uhlenbeck (OU) process. This so-called ‘forward’ process can be inverted via a ‘reverse’ process, which is mathematically guaranteed to exist for reasonable choices of initial distribution and forward process [10]. Thus, to generate new samples from $p(\mathbf{x})$, we can sample from the simpler distribution (e.g. a Gaussian) and run the reverse process.

Forward/reverse diffusion. We consider forward processes defined by the stochastic differential equation (SDE)

$$\dot{\mathbf{x}} = -\beta(t)\mathbf{x} + g(t)\boldsymbol{\eta}(t) \quad (1)$$

where $\beta(t)$ controls the decay of signal, $g(t)$ is a time-dependent noise amplitude, $\boldsymbol{\eta}(t)$ is a vector of independent Gaussian white noise terms, and time runs from $t = 0$ to T . Its reverse process is

$$\dot{\mathbf{x}} = -\beta(t)\mathbf{x} - g(t)^2\mathbf{s}(\mathbf{x}, t) + g(t)\boldsymbol{\eta}(t) \quad (2)$$

where $\mathbf{s}(\mathbf{x}, t) := \nabla_{\mathbf{x}} \log p(\mathbf{x}, t)$ is the score function, and where we use the standard convention that time runs *backward*, i.e. from $t = T$ to 0. In this paper, we focus on one popular forward process: the variance-preserving SDE, which enforces the constraint $\beta(t) = \frac{1}{2}g^2(t)$. The marginal probabilities of this process are

$$p(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t | \alpha_t \mathbf{x}_0, \sigma_t^2 \mathbf{I}) \quad \alpha_t := e^{-\int_0^t \beta(t') dt'} \quad \sigma_t^2 := 1 - e^{-2 \int_0^t \beta(t') dt'} \quad (3)$$

where α_t and σ_t represent the signal and noise scale, satisfying $\alpha_t^2 + \sigma_t^2 = 1$. Normally, as t goes from $0 \rightarrow T$, signal scale α_t monotonically decreases from $1 \rightarrow 0$ and σ increases from $0 \rightarrow 1$.

(Appendix D relates our notation to others’ notation.) Note that there exists a deterministic *probability flow ODE* with the same marginal probabilities [3] as the reverse SDE:

$$\dot{\mathbf{x}} = -\beta(t)\mathbf{x} - \frac{1}{2}g(t)^2\mathbf{s}(\mathbf{x}, t) \quad (4)$$

where time again runs backward from $t = 1$ to $t = 0$. In practice, instead of the SDE (Eq.1), this deterministic process is often used to sample from the distribution [5]. The behavior of the probability flow ODE will be our main focus.

Learning the score function. The score function, which is required to reverse the forward process, can be learned via gradient descent on the denoising score-matching objective

$$\mathbb{E}_{\mathbf{x}_0 \sim p(\mathbf{x}_0), \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})} \int_0^1 \gamma_t \|\epsilon_\theta(\alpha_t \mathbf{x}_0 + \sigma_t \epsilon, t) - \epsilon\|_2^2 dt \quad \epsilon_\theta(\mathbf{x}_t, t) \approx -\sigma_t \nabla_{\mathbf{x}} \log p(\mathbf{x}_t) \quad (5)$$

where $\epsilon_\theta(\mathbf{x}_t, t)$ can be parameterized by a network, and γ_t is a positive weighting function [3].

DDIM/PNDM samplers. Sampling the reverse process is somewhat independent of score function learning [5], enabling researchers to separately study its efficiency. Most samplers are equivalent to integrating the reverse SDE or ODE with some discretization. The original DDPM[4] is effectively the same as discretizing a reverse SDE (Eq. 2). The deterministic DDIM sampler [11] is equivalent to solving the probability flow ODE (Eq. 4) with an Euler method, which dramatically reduced the required number of steps. More advanced numerical methods have been used to integrate Eq. 4; PNDM [12], the default sampler for Stable Diffusion, utilizes an RK4 method. In this work, we focus our theory and analysis on the probability flow ODE and the corresponding DDIM/PNDM samplers. We comment on how other samplers affect our results in Sec. B.8.

3 Salient observations about image generation

How should we measure generation progress? A common way to monitor image generation progress is to observe how \mathbf{x}_t (or the decoded image $G(\mathbf{x}_t)$ in the case of latent diffusion [13]) changes over time. As previously mentioned, this approach tends to show a fully-formed image unveiled from noise (Fig. 1A, top row). But is this what is ‘actually’ happening? A simple but useful alternative is to observe (appropriately scaled) *differences* $k(\mathbf{x}_{t-k} - \mathbf{x}_t)$ between close time points—the next ‘layer of paint’ that has been added to the canvas. These often appear to be like a sketch of the final outcome early in generation, and are increasingly contaminated by noise towards the end (Fig. 1A, middle row).

A more principled alternative, proposed by [4], is to consider the sequence of *endpoint estimates* $\hat{\mathbf{x}}_0$ of the reverse diffusion trajectory. In particular, the weighted combination of the state and score

$$\hat{\mathbf{x}}_0(\mathbf{x}(t)) := \frac{\mathbf{x}(t) + \sigma_t^2 \mathbf{s}(\mathbf{x}(t), t)}{\alpha_t} \approx \frac{\mathbf{x}(t) - \sigma_t \epsilon_\theta(\mathbf{x}(t), t)}{\alpha_t} \quad (6)$$

provides an endpoint estimate that improves over time. Technically, this is the minimum mean squared error (MMSE) estimator of \mathbf{x}_0 given \mathbf{x}_t and Gaussian noise [14], which has also been called ideal denoiser [5]. We found tracking this statistic throughout reverse diffusion provides substantial insight into generation: as early as the first time step, a rough outline is visible. As time goes on, one tends to see progressively finer details filled in (Fig. 1A, bottom). We observed similar results for unconditional diffusion models (trained on MNIST, CIFAR-10, and CelebA-HQ; see SI Fig. 6-8).

When do different image features tend to emerge? According to the endpoint estimate $\hat{\mathbf{x}}_0(\mathbf{x}_t)$, high-level features tend to emerge before low-level ones [4, 5]. For example, when generating “a portrait of an aristocrat” (Fig. 1A), a generic face-like shape appears, and then is refined to include blobs that correspond to hat/hair and body. Coarse facial features and various image colors emerge, facial hair appears, and the blob above the face is ‘reinterpreted’ into a hat. Finally, high-frequency details of the face and clothes are added. Unconditional models exhibit similar behavior (SI Fig. 6,7).

What is the shape of individual trajectories? We also studied the geometry of reverse diffusion trajectories. Although the dimensionality of image/latent space is quite large, individual trajectories are effectively two-dimensional: the average variance explained by the top two principal components (PCs) is 99.98% for our CelebA model, and 99.54% for Stable Diffusion (see Table B.2 and Fig. 10). To good approximation, \mathbf{x}_t always remains in the plane defined by the initial noise \mathbf{x}_T and the final state \mathbf{x}_0 : the variance explained by a projection onto this 2D plane is higher than 99.2% for all models. Furthermore, the reverse diffusion trajectory is well-approximated by a rotation within this plane (Fig. 1B), i.e.

$$\mathbf{x}_t \approx K_t \mathbf{x}_0 + \sqrt{1 - K_t^2} \mathbf{x}_T \quad (7)$$

where $0 \leq K_t \leq 1$, $K_T = 0$, and $K_0 = 1$. (This may not be a ‘true’ rotation if \mathbf{x}_0 and \mathbf{x}_T have unequal norms, which depends on training data normalization.) When $K_t = \alpha_t$, this equation explains almost all trajectory variance: 97.51% for Stable Diffusion and 98.93% for CelebA diffusion.

4 Theoretical analysis of sampling trajectories

In this section, we will develop a simple but quantitative theory to explain our previous observations. For reasons of analytical tractability, we will start by studying reverse diffusion assuming that the training set is well-described by a multivariate Gaussian distribution, and that the score function is exact. Though simple, the Gaussian model allows us to explore the implications of low-dimensional manifolds and features with different variances. Further, the Gaussian mode assumption is appropriate early in reverse diffusion since the VP-SDE maps all distributions to Gaussians. It may also be reasonable for describing the short time scale dynamics of arbitrary distributions, since arbitrary distributions can be well-approximated by Gaussian mixtures, and the score functions of high-dimensional Gaussian mixtures could be locally dominated by the contribution of a single mode as long as the modes are well-separated.

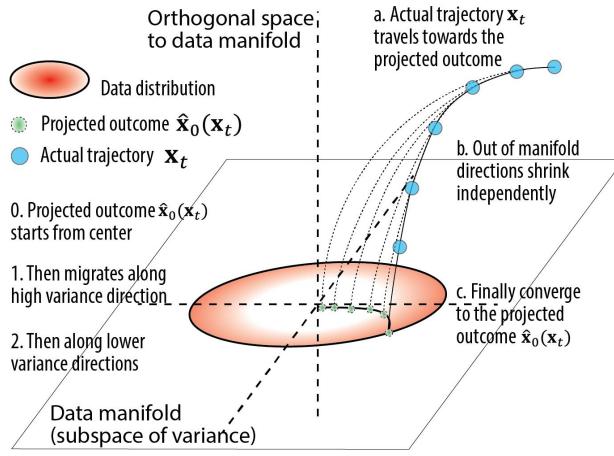


Figure 2: **Geometry of single mode reverse diffusion.**

4.1 Exact solution to a Gaussian score model

Let $\mathbf{x}_t \in \mathbb{R}^D$, and the mean and covariance of the mode be $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. Assuming $\boldsymbol{\Sigma}$ has rank $r \leq D$, it has a compact singular value decomposition (SVD) $\boldsymbol{\Sigma} = \mathbf{U} \boldsymbol{\Lambda} \mathbf{U}^T$, where $\mathbf{U} = [\mathbf{u}_1, \dots, \mathbf{u}_r]$ is a $D \times r$ semi-orthogonal matrix. The columns of \mathbf{U} are the principal axes along which the mode varies, and their span comprises the ‘image manifold’. The score function $s(\mathbf{x}, t) = \nabla_{\mathbf{x}} p(\mathbf{x}, t)$ at time $t > 0$ is the score of a Gaussian $\mathcal{N}(\alpha_t \boldsymbol{\mu}, \sigma_t^2 \mathbf{I} + \alpha_t^2 \boldsymbol{\Sigma})$, so the probability flow ODE is

$$\dot{\mathbf{x}} = -\beta(t)\mathbf{x} - \beta(t)s(\mathbf{x}, t) = -\beta(t)\mathbf{x} - \beta(t)(\sigma_t^2 \mathbf{I} + \alpha_t^2 \boldsymbol{\Sigma})^{-1}(\alpha_t \boldsymbol{\mu} - \mathbf{x}), \quad (8)$$

which is exactly solvable (Appendix G). The solution is a sum of on- and off-manifold components:

$$\begin{aligned} \mathbf{x}_t &= \alpha_t \boldsymbol{\mu} + \frac{\sigma_t}{\sigma_T} \mathbf{y}_T^\perp + \sum_{k=1}^r \psi(t, \lambda_k) c_k(T) \mathbf{u}_k & \psi(t, \lambda_k) &= \sqrt{\frac{\sigma_t^2 + \lambda_k \alpha_t^2}{\sigma_T^2 + \lambda_k \alpha_T^2}} \\ \mathbf{y}_T^\perp &= (\mathbf{I} - \mathbf{U}^T \mathbf{U})(\mathbf{x}_T - \alpha_T \boldsymbol{\mu}) & c_k(T) &= \mathbf{u}_k^T (\mathbf{x}_T - \alpha_T \boldsymbol{\mu}). \end{aligned} \quad (9)$$

There are three terms: 1) the scaling up of the distribution mean; 2) the scaling down of the off-manifold component \mathbf{y}_T^\perp , proportional to the noise scale $\psi(t, 0) = \frac{\sigma_t}{\sigma_T}$; and 3) the on-manifold movement along each eigenvector governed by $\psi(t, \lambda_k)$ (visualized in Fig. 3A). The initial condition \mathbf{x}_T can be decomposed into contributions along each principal direction, an off-manifold contribution, and a $\alpha_T \boldsymbol{\mu}$ contribution. Below, we will explore how this exact solution recapitulates our observations.

But first, we note that it explicitly connects the initial noise pattern \mathbf{x}_T to the final sample \mathbf{x}_0 :

$$\mathbf{x}_0 = \boldsymbol{\mu} + \sum_{k=1}^r \psi(0, \lambda_k) \mathbf{u}_k \mathbf{u}_k^T (\mathbf{x}_T - \alpha_T \boldsymbol{\mu}). \quad (10)$$

This is reminiscent of *linear filtering* adapted to the data distribution. The final location of \mathbf{x}_0 along each feature axis \mathbf{u}_k is determined by the projection of the initial noise pattern onto that feature, amplified by the standard deviation $\psi(0, \lambda_k) \approx \sqrt{\lambda_k}$. Thus, it is the subtle alignment between the noise pattern and image manifold features that determine what is generated.

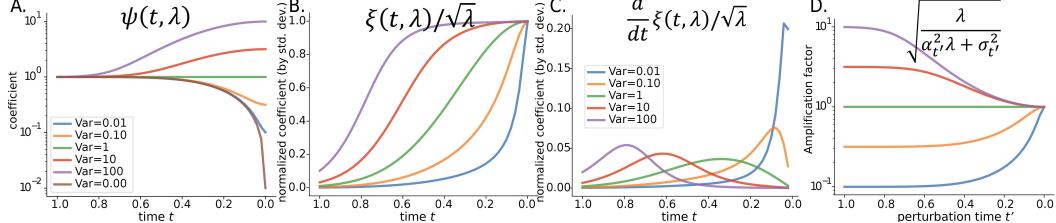


Figure 3: **Analytical solution to diffusion dynamics in Gaussian case** **A.** $\psi(t, \lambda)$ governs the dynamic of state \mathbf{x}_t along each principal axis \mathbf{u}_k . **B.** $\xi(t, \lambda)$ governs the dynamics of endpoint estimate $\hat{\mathbf{x}}_0(\mathbf{x}_t)$ along each PC, normalized by the standard deviation $\sqrt{\lambda_k}$. **C.** Time derivative of $\xi(t, \lambda)/\sqrt{\lambda}$, highlighting the ‘critical period’ when the feature develops. **D.** $\sqrt{\lambda}/(\sigma_{t'}^2 + \lambda \alpha_{t'}^2)$, which quantify the amplification effect of a perturbation along PC \mathbf{u}_k at time t' (Eq.14). We used the α_t schedule from ddpm-CIFAR-10.

4.2 Theoretical support for primary claims

State trajectory. Throughout the generation process $t = T \rightarrow 0$, all $\psi(t, \lambda)$ moves from 1 to $\sqrt{\lambda}$. So, the off-manifold component \mathbf{y}^\perp decays to 0 towards the end; while the on-manifold component along \mathbf{u}_k is scaled up by $\psi(0, \lambda_k) \approx \sqrt{\lambda_k}$ which is the standard deviation along \mathbf{u}_k . From Fig. 3 A, we can see the state moves along high variance dimensions first, while the low variance and off-manifold dimensions decay late until the end. This explains when visualizing the state itself, we see the well-formed image unveiled from noise till the end.

2D rotations. Our solution for \mathbf{x}_t implies (see Appendix I for the derivation and more discussion)

$$\mathbf{x}_t \approx \alpha_t \mathbf{x}_0 + \sqrt{1 - \alpha_t^2} \mathbf{x}_T + \sum_{k=1}^r \left\{ \sqrt{\sigma_t^2 + \lambda_k \alpha_t^2} - \alpha_t \sqrt{\lambda_k} - \sigma_t \right\} c_k(T) \mathbf{u}_k,$$

i.e. \mathbf{x}_t dynamics tend to look like a rotation within the 2D plane formed by \mathbf{x}_0 and \mathbf{x}_T (Fig. 1B) up to on-manifold correction terms. The correction terms tend to be small; assuming $r \ll D$, and that the typical overlap between the initial noise and any given eigendirection is roughly $1/\sqrt{D}$,

$$\left\| \mathbf{x}_t - \alpha_t \mathbf{x}_0 - \sqrt{1 - \alpha_t^2} \mathbf{x}_T \right\|_2^2 \leq \left(1 - \frac{\sqrt{2}}{2} \right)^2 \frac{r}{D} \ll 1. \quad (11)$$

Interestingly, the rotation only depends on α_t and σ_t , and not on any properties of the mode. We empirically find, however, that it *does* depend on the sampler (Section B.8.1). Generally, for any VP-SDE, the dynamics of the scaled state \mathbf{x}_t/α_t can be written as (see Appendix I)

$$\frac{d}{dt} \left(\frac{\mathbf{x}_t}{\alpha_t} \right) = -\frac{\beta_t}{\sigma_t^2} \left[\hat{\mathbf{x}}_0(\mathbf{x}_t) - \frac{\mathbf{x}_t}{\alpha_t} \right]. \quad (12)$$

This ODE is generally difficult to solve; however, assuming $\hat{\mathbf{x}}_0(\mathbf{x}_t)$ changes more slowly than \mathbf{x}_t , we find that the trajectory of \mathbf{x}_t can be understood as constantly ‘rotating’ towards the endpoint estimate (Fig.2, dashed curves).

Feature emergence order. The endpoint estimate $\hat{\mathbf{x}}_0$ can be written as

$$\hat{\mathbf{x}}_0(\mathbf{x}_t) = \boldsymbol{\mu} + \sum_{k=1}^r \xi(t, \lambda_k) c_k(T) \mathbf{u}_k \quad \xi(t, \lambda) := \frac{\alpha_t \lambda}{\sqrt{(\alpha_t^2 \lambda + \sigma_t^2)(\alpha_T^2 \lambda + \sigma_T^2)}}. \quad (13)$$

This equation implies that $\hat{\mathbf{x}}_0$ *always remains on the image manifold*, which explains why the endpoint estimates of well-trained models look like images, rather than images contaminated with noise. Since $\xi(T, \lambda) \approx 0$, $\hat{\mathbf{x}}_0$ is initially similar to the distribution average (e.g. the generic face for CelebA [15]). Note that we can exploit $\hat{\mathbf{x}}_0$ being on-manifold to *infer the structure of the image manifold* (Sec. 6).

The sigmoidal behavior of the $\xi(t, \lambda)$ function indicates that a given eigendirection is reflected in the endpoint estimate around when $\sigma_t = \alpha_t \sqrt{\lambda}$, i.e. when the noise variance matches the scaled signal variance (Fig. 3C). Moreover, since this happens when $\alpha_t \approx 1/\sqrt{1+\lambda}$, image manifold features appear in order of descending variance: first the highest variance features, then the next highest, and so on. Natural images have more power and variance in low frequencies than high frequencies [16]. For face images, features such as gender, head orientation, and skin color, have higher variance than subtle features such as glasses, facial, and hair texture [17]. Thus, the combination of natural image statistics and the diffusion process explain why features such as the layout of a scene and the ‘semantic’ features of faces are specified first in the endpoint estimate, or why generation is outline-first, details later.

Effect of perturbations. Finally we examined the effect of perturbation and feature commitment. Suppose at time $t' \in (0, T)$ the off-manifold directions are perturbed by $\delta \mathbf{y}^\perp$, and the on-manifold direction coefficients are perturbed by amounts δc_k ; then their effect on the generated image \mathbf{x}_0 is

$$\Delta \mathbf{x}_0 = \sum_{k=1}^r \frac{\psi(0, \lambda_k)}{\psi(t', \lambda_k)} \delta c_k \mathbf{u}_k = \sum_{k=1}^r \sqrt{\frac{\lambda_k}{\sigma_{t'}^2 + \lambda_k \alpha_{t'}^2}} \delta c_k \mathbf{u}_k \quad (14)$$

Thanks to denoising, the off-manifold perturbation has no effect on the sample, while on-manifold perturbations have *maximal effect at different periods* (Fig. 3 D). Perturbations of high variance features ($\lambda > 1$) are amplified at the start and then decayed; perturbation along low variance features ($\lambda < 1$) has a reduced effect until the end. This time-dependent ‘filtering’ explains the classic finding [4] that during the reverse diffusion process when noise is injected at different steps, early perturbation creates variations of layout and semantic features and late perturbation varies details.

Summary. The examination of the solution to Eq.1 suggests a conceptual understanding of the generation process, as depicted in Fig. 2: The endpoint estimate $\hat{\mathbf{x}}_0$ travels on the image manifold, starting from the center of the distribution, moving first along the high variance axes, and then the lower variance axes; concurrently, the state \mathbf{x}_t in the ambient space keeps rotating towards the evolving endpoint estimate.

4.3 Beyond the Gaussian score function approximation

Diffusion as retrieval: dynamics of the state with general point cloud. In practice, diffusion models are trained using a finite set of points $\{\mathbf{y}_i\}, i = 1, \dots, N$. Thus, without augmentation, the training distribution is effectively a collection of delta functions or a mixture of Gaussian with negligible width. $p(\mathbf{x}_0) = \frac{1}{N} \sum_i \delta(\mathbf{x}_0 - \mathbf{y}_i)$. We proved that (see Appendix J), with the same forward process (Eq.1), the endpoint estimate is a weighted average of the ‘nearest’ training data, while the score is locally equivalent to that of an isotropic Gaussian centered at $\hat{\mathbf{x}}_0(\mathbf{x}_t)$ diffused to time t

$$\hat{\mathbf{x}}_0(\mathbf{x}_t) = \sum_i w_i(\mathbf{x}_t, t) \mathbf{y}_i, \quad w_i(\mathbf{x}_t, t) := \text{softmax}\left(\left\{-\frac{\alpha_t^2}{2\sigma_t^2} \|\mathbf{y}_i - \frac{\mathbf{x}_t}{\alpha_t}\|^2\right\}\right)_i \quad (15)$$

$$\mathbf{s}(\mathbf{x}, t) = \frac{-\mathbf{x}_t + \alpha_t \hat{\mathbf{x}}_0(\mathbf{x}_t)}{\sigma_t^2}. \quad (16)$$

The weights are defined by the softmax of the negative squared distance between \mathbf{x}_t/α_t and all data points, with the temperature set at $2\sigma_t^2/\alpha_t^2$. Consistent with the Gaussian case, at the start of generation, the temperature $2\sigma_T^2/\alpha_T^2$ is much higher than the distances, so the estimated outcome corresponds to the mean of all data points. As the generation progresses, $w_i(\mathbf{x}_t, t)$ focuses on the set of training samples that are closest to \mathbf{x}_t/α_t . Towards the end, the temperature approaches 0, and the softmax focuses on one training sample—the one that generation converges to.

In summary, we can see when the *score is exact*, the reverse diffusion process is *equivalent to an iterative image retrieval process* for a discrete dataset: the scaled state \mathbf{x}_t/α_t migrates towards the

weighted average of a subset of data points, and gradually focuses the weights on the nearest data points until it finally converges to one data point. But a priori, it is unclear if this matches what neural network score approximators learn.

5 Validating the normative theory on actual diffusion models

In this section, we aim to test the extent to which the Gaussian theory can accurately predict actual reverse diffusion trajectories, and to compare the quality of its predictions to two other score function approximations: the delta function mixture described in the previous section, and the Gaussian mixture. Although real image distributions are certainly not Gaussian, a Gaussian approximation may be a reasonable description of the beginning of reverse diffusion, when the data distribution is sufficiently ‘blurred’. On the other extreme, a mixture of delta functions centered on the training data is an important point of comparison because it represents the optimal solution to Eq. 5 if no data augmentation is used (Sec.4.3); deviations of the score network from this model suggest that the network does not converge to the ‘optimal’ score, and may hint at how they learn to generalize.

Gaussian solution predicts early diffusion trajectory. To test the Gaussian approximation, we numerically computed the mean and covariance of training samples in pixel space for models trained on MNIST, CIFAR-10, and CelebA-HQ. Then Eq. 9 was used with these means and covariances to predict the evolution of \mathbf{x}_t . We found that the early phase of reverse diffusion is well-predicted by the Gaussian solution (Fig. 4A-B). Visually, as the low-frequency information is determined early on, the ‘layout’ of the final image is also well-predicted, while high-frequency details such as edges are less well-predicted. The deviation between the predicted and actual trajectory grows large at around 20 reverse diffusion steps ($t = 0.6$, Fig. 4C); we interpret this as the moment when the *single mode* assumption breaks down, and the trajectory starts to be guided by a more complicated distribution. (For MNIST and CelebA see Fig. 12,13.) We also found that the dynamics along off-manifold directions are well-predicted by the Gaussian solution (Fig. 15).

This result bears interesting implications for score function approximation. It suggests that even for natural images, at high noise scales, $p(\mathbf{x}_t)$ is indistinguishable from a multivariate Gaussian. Therefore, the early phase score function can be effectively approximated by the Gaussian score, which is an *affine function* of \mathbf{x} , specifically $\Sigma^{-1}(\mu - \mathbf{x})$. This raises doubts about the necessity of a nonlinear neural network for this phase (Sec.6.1).

Gaussian solution predicts late trajectory better than the exact score model. Taking a step further, we tested two other models on MNIST and CIFAR-10: the 10-mode Gaussian mixture model (*GMM*), where each class is fit by one Gaussian mode; and the *exact* score model (Sec.4.3), where a delta mode is defined on each training image. For these models, the score function can be evaluated precisely, but the trajectory has no closed-form solution, so we used an off-the-shelf RK4 ODE solver to integrate it. (For the score of the Gaussian mixture model, see App.J.) While all models predict the early trajectory well, surprisingly, we found that the trajectory predicted by the exact score deviates from the actual DDIM trajectory, and does so even earlier than the Gaussian solution (Fig. 4D). Visually, both the Gaussian model and GMM predict the generated image better than the exact score model (Fig. 4B), with a significantly lower MSE ($p < 10^{-30}$). Though the Gaussian model and GMM have comparable predictions, GMM has slightly lower error ($p < 10^{-10}$). Thus, we can infer something about the structure of the learned score function through these trajectories. This result implies that the actual score function learned by optimizing Eq. 5 is different from the exact score—especially late in reverse diffusion—and that it is more similar to the ‘blurrier’ score of a Gaussian or Gaussian mixture, possibly due to the regularizing effect of our neural network function approximator. Similar results were observed for the MNIST model (Fig. 16).

6 Applications: accelerating sampling, characterizing image manifold

6.1 Accelerating unconditional diffusion by teleportation

We can exploit the fact that the Gaussian analytical solution provides a surprisingly good approximation to the early part of the sampling trajectory by using the solution to ‘teleport’ to time t . Namely, instead of evaluating the score function approximated by neural network ϵ_θ and integrating the

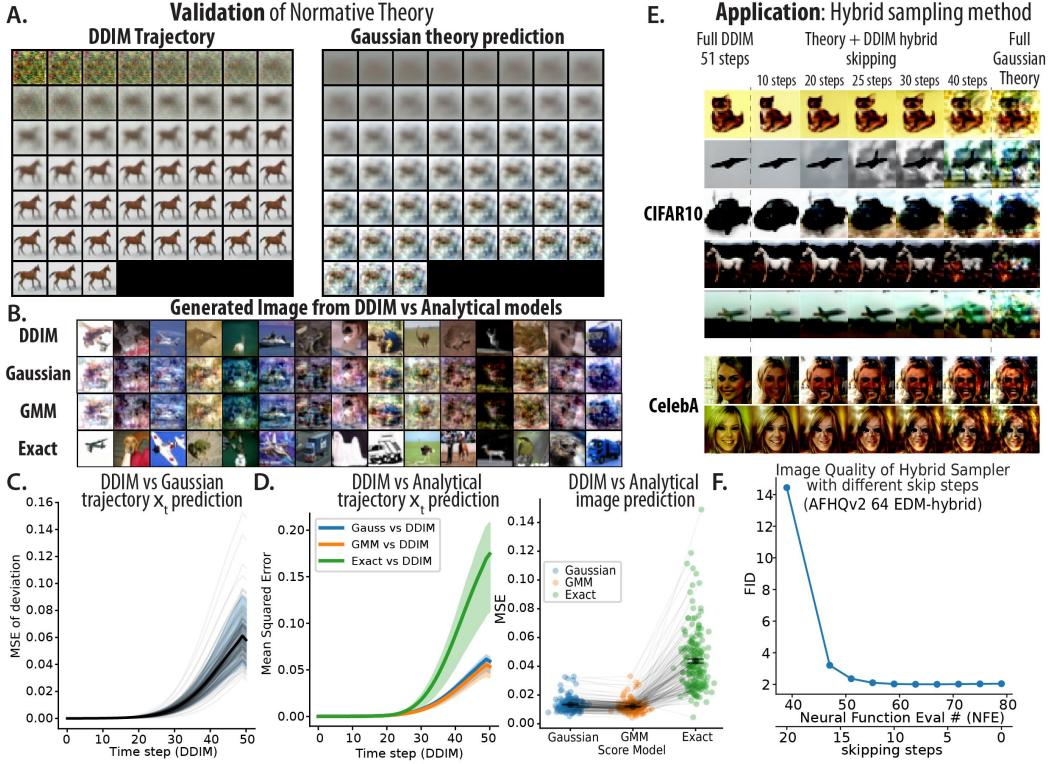


Figure 4: Comparing analytical solution to DDIM sampling for CIFAR-10 diffusion model.

A. $\hat{x}_0(x_t)$ of a DDIM trajectory and the Gaussian solution with the same initial condition x_T . **B.** Samples generated by DDIM and the analytical theories from the same initial condition. **C.** Mean squared error between the x_t trajectory of DDIM and Gaussian solution. **D.** Comparing the state trajectory and final sample of three normative models (Gaussian, GMM, exact) with DDIM. **E.** Hybrid sampling method combines Gaussian theory prediction with DDIM. **F.** Image quality of the hybrid method (FID score) as a function of different numbers of skipped steps for EDM model and sampler [5] (see Appendix F).

probability flow ODE, we can use the Gaussian prediction for x_t instead, where the μ and Σ used are those of the training set. In principle, this speedup can be combined with any deterministic or stochastic sampler. Here, we showcase its effectiveness with DDIM and Heun’s sampler [5].

We tested this hybrid sampler on unconditional diffusion models of MNIST, CIFAR-10, and CelebA-HQ. For the MNIST and CIFAR-10 models, we can easily skip 40% of the initial steps with the Gaussian solution without much of a perceptible change in the final sample (Fig.4E). Quantitatively, we found skipping up to 40% of the initial steps can even slightly decrease the Frechet Inception Distance score, and improve the quality of generated samples (Fig.11). For models of higher resolution data sets like CelebA-HQ, we need to be more careful; skipping more than 20% of the initial steps will induce some perceptible distortions in the generated images (Fig.4E bottom), which suggests that the Gaussian approximation is less effective for larger images. The reason may have to do with a low-quality covariance matrix estimate, which could arise from the small number of training images compared to the effective dimensionality of the image manifold.

With the more optimized pre-trained diffusion models in EDM and Heun’s sampler [5], we can still reliably skip 15–30% neural function evaluation time, while maintaining FID scores competitive with the state-of-the-art level. Specifically, we achieved FID score of 1.934 on CIFAR10 with 25 NFEs, and FID score of 2.026 on AFHQv2 64 with 59 NFEs (Fig.4F , full results in Fig.17 in Sec.B.5). This shows that our hybrid acceleration trick is generally effective even when combined with state-of-the-art diffusion models and samplers.

6.2 Characterizing image manifold by analyzing sampling trajectory

For a large text-to-image conditional model like Stable Diffusion [13], which given a text prompt τ samples from the distribution $p(\mathbf{x}|\tau)$, we cannot easily apply our Gaussian theory since we do not have easy access to the conditional distribution. However, we can still leverage the qualitative

insight that the endpoint estimate \hat{x}_0 remains on the image manifold; in particular, \hat{x}_0 trajectories in principle reflect interesting manifold directions. When we visualized the PC directions through the decoder, they appeared to be a clean variation of the target image, i.e. a tangent vector to the image manifold (Fig. 5A, 18). Consistent with our theory (Eq.14), we found perturbations in these PC directions more effectively produce nontrivial image variants than perturbations in random directions (Fig. 20,21,22).

This gave us an effective way to find local on-manifold directions using a single sampling trajectory. We found two ways to apply these directions: 1) linearly perturb the final state x_0 along these directions; and 2) perturb the state x_t during reverse diffusion (Eq.14). The first method can visualize the local linear image manifold around the generated image (Fig.5C). But since the image manifold is not linear, traveling too far along PC directions will induce distortion. In contrast, the second method can visualize the local nonlinear manifold, showing that object identity and layout can undergo dramatic changes along these nonlinear axes (Fig.5D). This proved the principle that we can use sampling trajectories to characterize the ‘image manifold’ embedded in the diffusion model.

7 Discussion

To what extent do our main findings—low-dimensional trajectories, outline-first and details-later image generation, and increasing commitment to image elements—hold true for other diffusion model variants and generative models? Simulating reverse SDEs (Eq. 2) instead of ODEs should not yield many qualitative differences; instead of linear ODEs, one has similarly-behaved OU processes. Antognini and Sohl-Dickstein [18] show OU trajectories are also very low-dimensional. Arguments similar to ones we have made suggest other kinds of models may possess a simple analytic description early in generation. For example, Xu and Liu et al.’s Poisson flow generative models [19, 20] may feature smeared-out charge distributions.

Our observations provide a phenomenological bridge between diffusion models and GANs. In the GAN literature, the idea of generating images by progressively modeling low-to-high resolution is well-established, e.g. by successful architectures like Progressive Growing GAN and StyleGAN [21, 22]. In this paradigm, early layers synthesize the rough layout of the image from noise, while the last few layers add realistic details. We showed that when looking at the projected outcome \hat{x}_0 , diffusion models have an intriguingly similar generative process. Moreover, the effect of injecting noise at different times is similar to injecting noise into different layers of e.g. StyleGAN. In this analogy, the sampling steps of diffusion are equivalent to GAN layers. This connection may facilitate shared techniques to understand both of them.

The latent manifold geometry of GANs and diffusion models may also be similar. Wang and Ponce [17] found that perturbing large variance directions of a latent space metric has large and interpretable effects on image generation in GANs. If we interpret the covariance matrix of a Gaussian mode as inducing a metric on the latent space of diffusion models, our perturbation-related observations can be cast in a similar light.

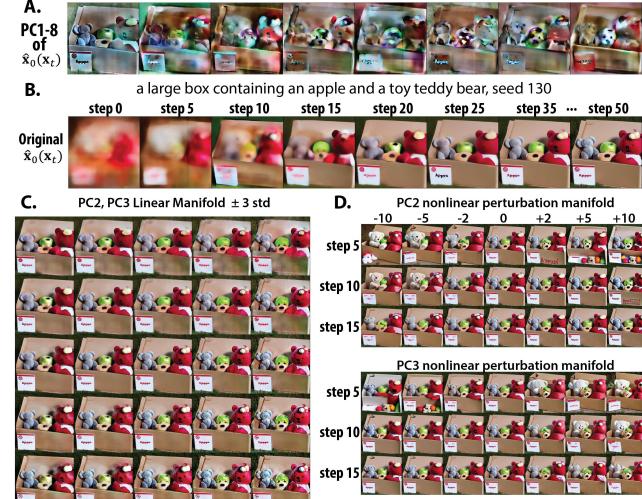


Figure 5: **Stable Diffusion: Local manifold map.** A. PCs of the projected outcome trajectory $\hat{x}_0(x_t)$ are on-manifold. B. Trajectory of endpoint estimate $G(\hat{x}_0(x_t))$. C. Perturbation by PC2 and PC3; notice an apple morphing into a teddy bear. D. Perturbing trajectory along PC2 or PC3 during reverse diffusion. Rows: different perturbation times. Columns: different magnitudes.

Our finding that the early diffusion trajectory is well-predicted by the Gaussian model is somewhat surprising. It calls for more attention to normative analyses of the score function (i.e. given some data, what should score be?). We showed that in the early phase, it has a simple linear structure that does not require advanced function approximation. With a deeper understanding of the score, we can build a better neural architecture that can approximate it more efficiently.

References

- [1] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265, Lille, France, 07–09 Jul 2015. PMLR. URL <https://proceedings.mlr.press/v37/sohl-dickstein15.html>.
- [2] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL <https://proceedings.neurips.cc/paper/2019/file/3001ef257407d5a371a96dc947c7d93-Paper.pdf>.
- [3] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=PxTIG12RRHS>.
- [4] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.
- [5] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *arXiv preprint arXiv:2206.00364*, 2022.
- [6] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022.
- [7] Li Kevin Wenliang and Ben Moran. Score-based generative model learn manifold-like structures with constrained mixing. In *NeurIPS 2022 Workshop on Score-Based Methods*, 2022. URL <https://openreview.net/forum?id=eSZqaIrDLZR>.
- [8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL <https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf>.
- [9] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Yingxia Shao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *arXiv preprint arXiv:2209.00796*, 2022.
- [10] Brian D.O. Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982. ISSN 0304-4149. doi: [https://doi.org/10.1016/0304-4149\(82\)90051-5](https://doi.org/10.1016/0304-4149(82)90051-5). URL <https://www.sciencedirect.com/science/article/pii/0304414982900515>.
- [11] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.
- [12] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. *arXiv preprint arXiv:2202.09778*, 2022.
- [13] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

- [14] Zahra Kadkhodaie and Eero Simoncelli. Stochastic solutions for linear inverse problems using the prior implicit in a denoiser. *Advances in Neural Information Processing Systems*, 34: 13242–13254, 2021.
- [15] Judith H. Langlois and Lori A. Roggman. Attractive faces are only average. *Psychological Science*, 1(2):115–121, 1990. doi: 10.1111/j.1467-9280.1990.tb00079.x. URL <https://doi.org/10.1111/j.1467-9280.1990.tb00079.x>.
- [16] Daniel L Ruderman. The statistics of natural images. *Network: computation in neural systems*, 5(4):517, 1994.
- [17] Bin Xu Wang and Carlos R Ponce. A geometric analysis of deep generative image models and its applications. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=GH7QRzUDdXG>.
- [18] Joseph M. Antognini and Jascha Sohl-Dickstein. PCA of high dimensional random walks with comparison to neural network training. In *NeurIPS*, pages 10328–10337, 2018.
- [19] Yilun Xu, Ziming Liu, Max Tegmark, and Tommi S. Jaakkola. Poisson flow generative models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=voV_TRqcWh.
- [20] Yilun Xu, Ziming Liu, Yonglong Tian, Shangyuan Tong, Max Tegmark, and Tommi Jaakkola. PFGM++: Unlocking the Potential of Physics-Inspired Generative Models. *arXiv e-prints*, art. arXiv:2302.04265, February 2023. doi: 10.48550/arXiv.2302.04265.
- [21] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.
- [22] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of Stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020.
- [23] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.
- [24] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/8a1d694707eb0fef65871369074926d-Paper.pdf.
- [25] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015. doi: 10.1109/CVPR.2015.7298594.

A Diffusion models used in numerical experiments

Table 1: **Diffusion models used for this paper’s numerical experiments.**

†: The MNIST diffusion model uses the upsampled $3 \times 32 \times 32$ RGB pixel space as sample space, while the original MNIST data set consists of 28×28 single channel black and white images. Thus the effective dimensionality of these images is around 784.

DATA SET	HUGGING FACE MODEL_ID	DIMENSIONALITY	LATENTS?	CONDITIONAL?
MNIST	DIMPO/DDPM-MNIST	$3 \times 32 \times 32 = 3072$ †	✗	✗
CIFAR-10	GOOGLE/DDPM-CIFAR10-32	$3 \times 32 \times 32 = 3072$	✗	✗
LSUN-CHURCH	GOOGLE/DDPM-CHURCH-256	$3 \times 256 \times 256 = 196,608$	✗	✗
CELEBA-HQ	GOOGLE/DDPM-CELEBAHQ-256	$3 \times 256 \times 256 = 196,608$	✗	✗
LAION-2B	RUNWAYML/STABLE-DIFFUSION-V1-5	$4 \times 64 \times 64 = 16,384$	✓	✓

B Supplementary Results

B.1 Visualizing image generation progress for other diffusion models

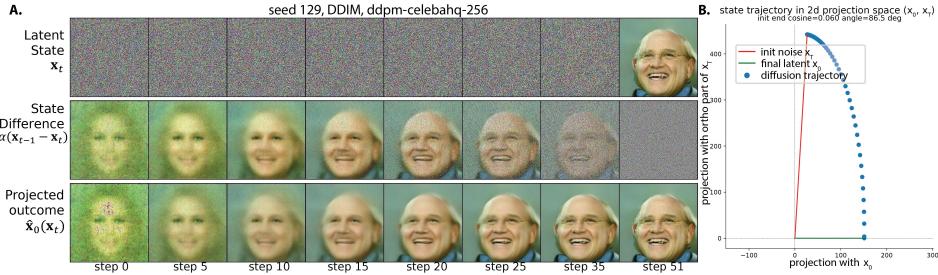


Figure 6: **Visualizing image generation process for DDPM-CelebA model.** Same layout as Fig.1.

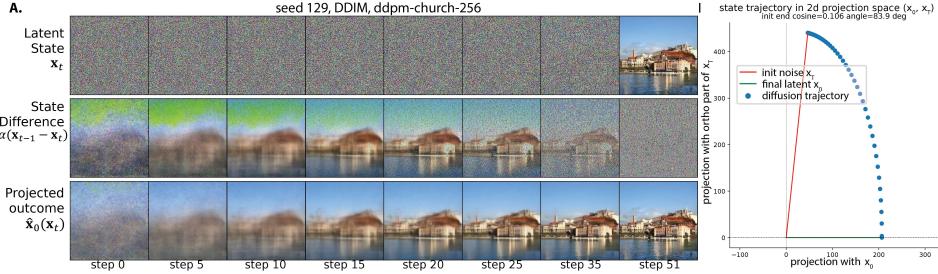


Figure 7: **Visualizing image generation process for DDPM-Church model.** Same layout as Fig.1.

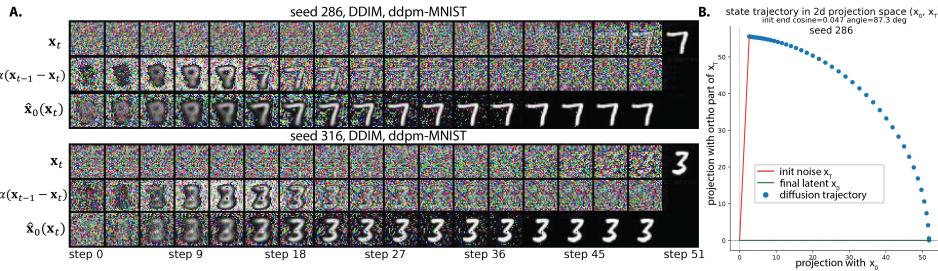


Figure 8: **Visualizing image generation process for DDPM-MNIST model.** Same layout as Fig.1.

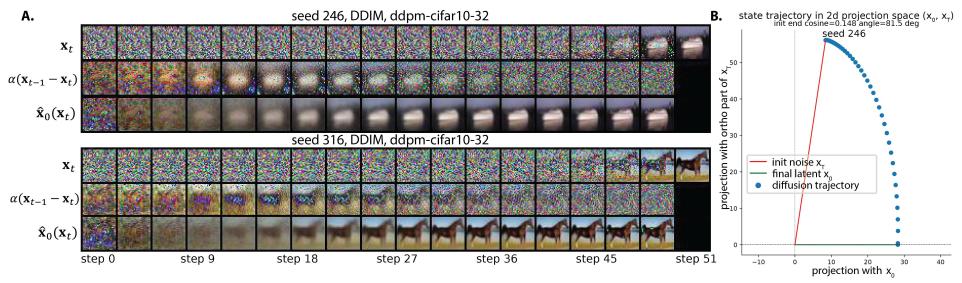


Figure 9: **Visualizing image generation process for DDPM-CIFAR-10 model.** Same layout as Fig. 1.

B.2 Latent space trajectory geometry statistics

Table 2: **Trajectory geometry statistics for different diffusion models.** Residual variance is here defined to be the squared norm of the error vector divided by the squared norm $\|\mathbf{x}_t\|^2$. Residual variance is computed for three approximations: 1) projecting a trajectory onto the top 2 PCs, 2) projecting a trajectory onto the plane spanned by \mathbf{x}_0 and \mathbf{x}_T , 3) approximating the trajectory by a rotation $\alpha_t \mathbf{x}_0 + \sqrt{1 - \alpha_t^2} \mathbf{x}_T$. Effective dimensionality is defined as the number of PCs needed to account for 99.9% of the variance. Shown are the effective dimensionalities of the trajectory \mathbf{x}_t , difference $\mathbf{x}_{t-1} - \mathbf{x}_t$, and the U-Net output $\epsilon_\theta(\mathbf{x}_t)$. All sampling used 51 time steps.

SAMPLER		RESIDUAL VARIANCE			DIM. FOR 99.9 VAR.		
		TOP 2 PC PROJ.	$\mathbf{x}_0, \mathbf{x}_T$ PROJ.	$\mathbf{x}_0, \mathbf{x}_T$ ROTATION	\mathbf{x}_t	$\Delta \mathbf{x}_t$	$\epsilon_\theta(\mathbf{x}_t)$
DDPM-MNIST	DDIM	0.08%	0.26%	1.70%	2	5	8
DDPM-CIFAR10-32	DDIM	0.05%	0.30%	1.01%	2	4	7
DDPM-CHURCH-256	DDIM	0.05%	0.31%	1.09%	2	5	8
DDPM-CELEBAHQ-256	DDIM	0.02%	0.10%	1.07%	2	4	7
STABLE-DIFFUSION-V1-5	PNDM	0.46%	0.81%	2.49%	5	34	28

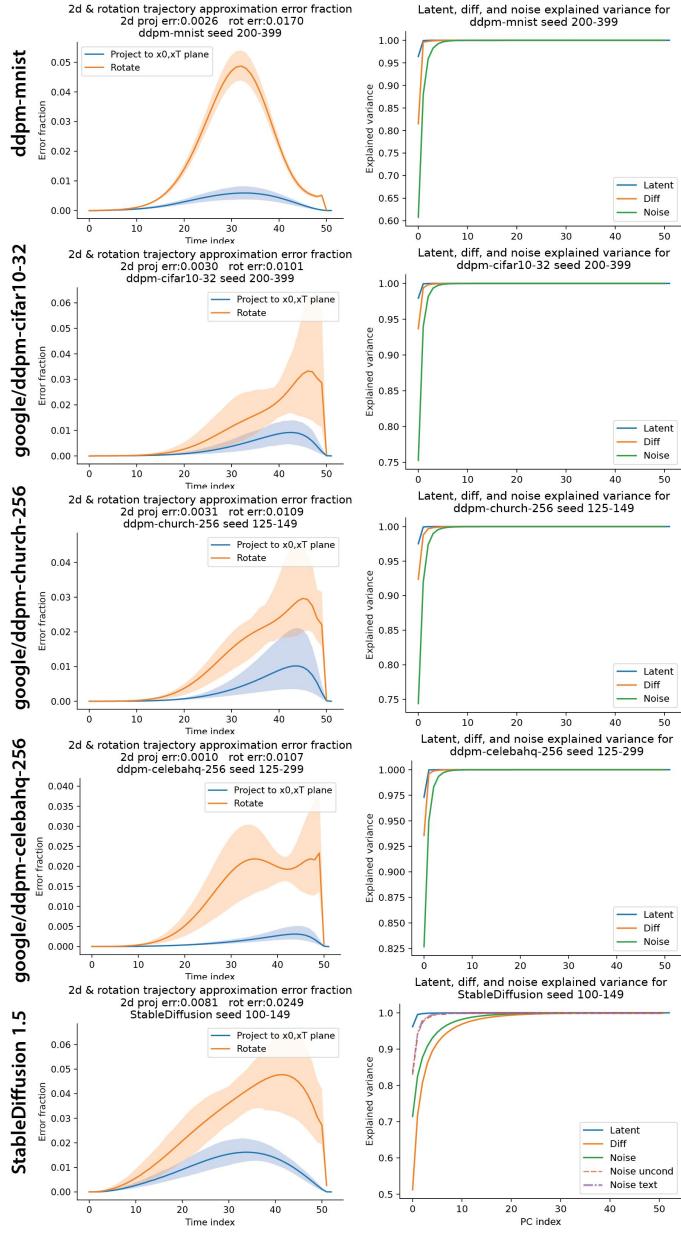


Figure 10: Geometry of diffusion sampling trajectories for 5 different diffusion models. **Left panel:** the error fraction of the projection onto the 2D plane defined by x_0 and x_T , and the error fraction under the rotation approximation. **Right panel:** the cumulative explained variance of PCs for the trajectory x_t , state differences $x_{t-1} - x_t$, and the output from the U-Net $\epsilon_\theta(x_t)$. 2 PCs explained almost all variance for the x_t trajectory, while the state difference $x_{t-1} - x_t$ and U-Net outputs are higher dimensional. For quantification see Tab. B.2.

B.3 Validation of the single mode theory on CIFAR, MNIST, and CelebA

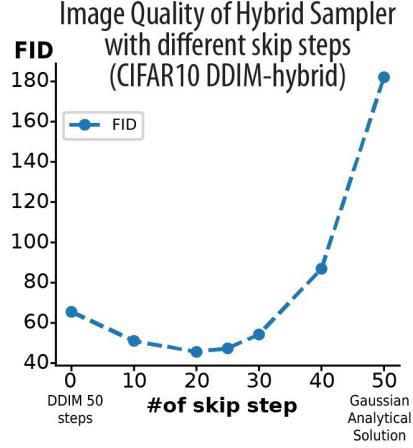


Figure 11: **Comparing analytical solution and actual diffusion process for the DDPM-CIFAR10 model** Image quality of the hybrid method (FID score) as a function of different numbers of skipped steps for the DDIM sampler. Note that the FID score of the original diffusion model without skipping is also not optimal.

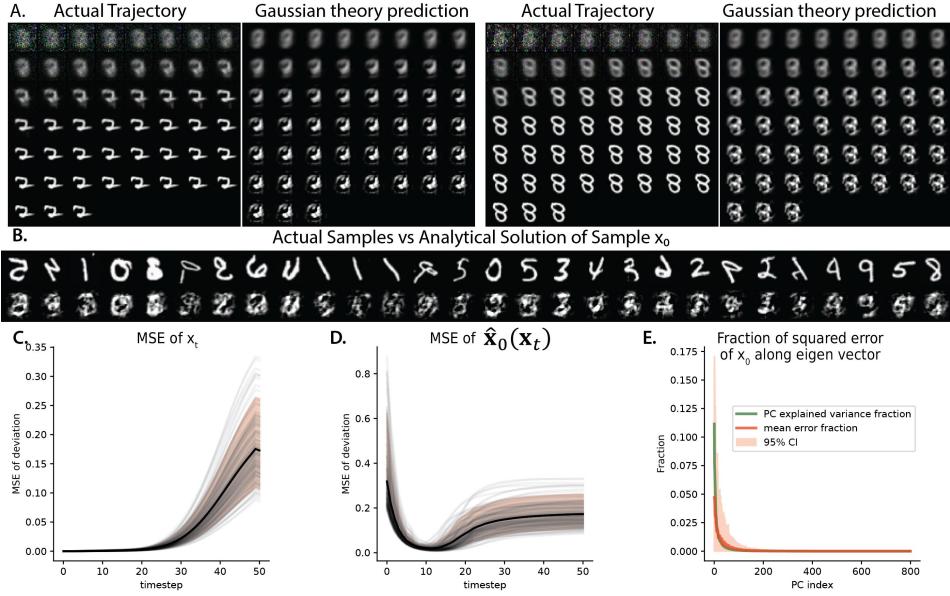


Figure 12: **Comparing analytical solution and actual diffusion process for the DDPM-MNIST model.** **A.** True and predicted endpoint estimate $\hat{x}_0(x_t)$ throughout reverse diffusion. **B.** Collection of samples of diffusion-generated images and the corresponding images predicted by our analytical theory. **C.** Mean squared error of the trajectory x_t . **D.** MSE of the endpoint estimate during diffusion. **E.** x_0 prediction error along each eigendimension.

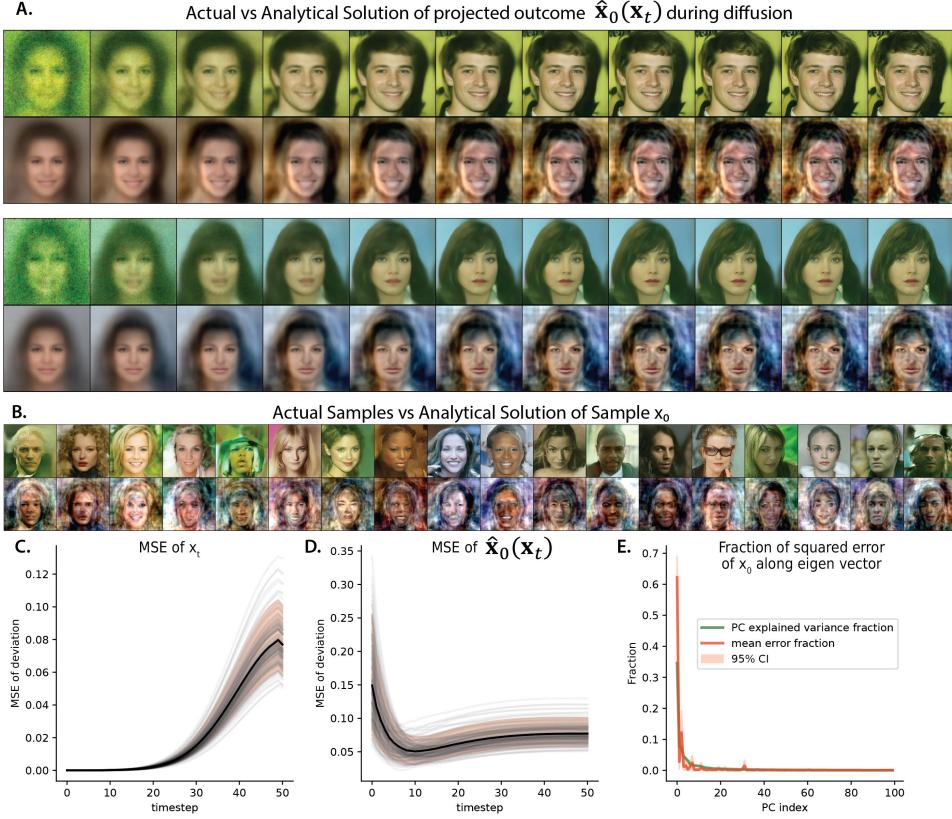


Figure 13: **Comparing analytical solution and actual diffusion process for the DDPM-CelebA model.** Same layout as Fig.12. Note that in **A.** the general layout and shading around the face is consistent between the theory and actual diffusion trajectory. Note in **B.**, the zoomed-out version of the predicted and actual sample look highly similar.

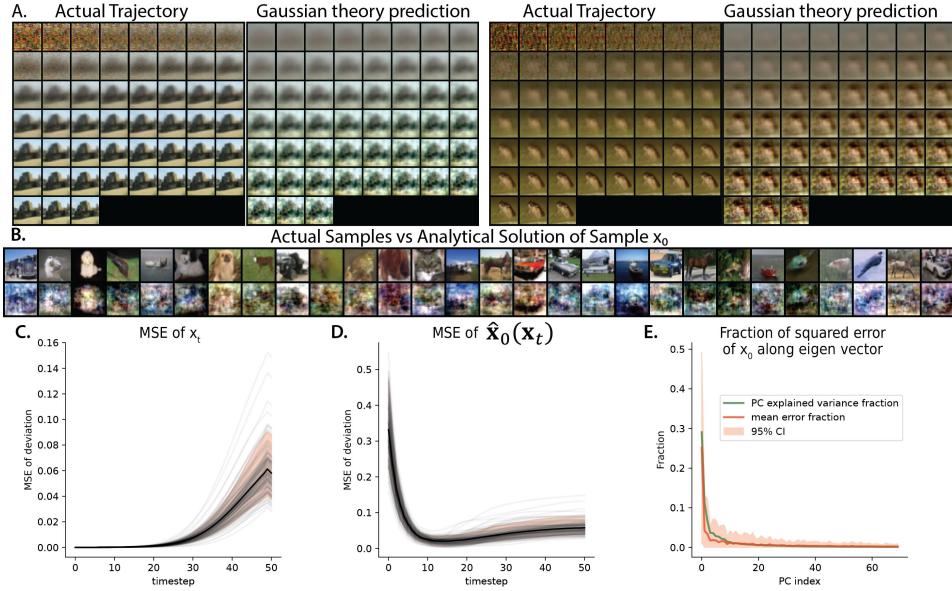


Figure 14: **Comparing analytical solution and actual diffusion process for the DDPM-CIFAR-10 model.** Same layout as Fig.12.

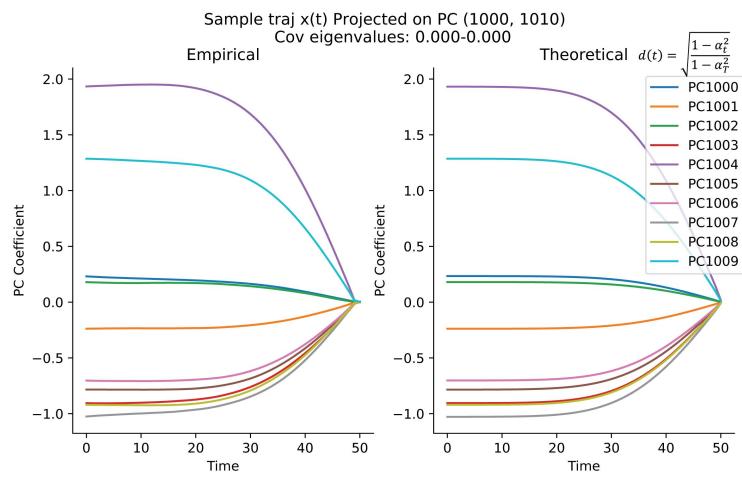


Figure 15: **Trajectory along off-manifold directions well-predicted by theory.** We computed the PC projection of actual trajectories and compared them to the theoretical prediction given the same initial value; they aligned well. DDPM-MNIST model.

B.4 Validation of the single mode and Gaussian mixture theory on an MNIST model

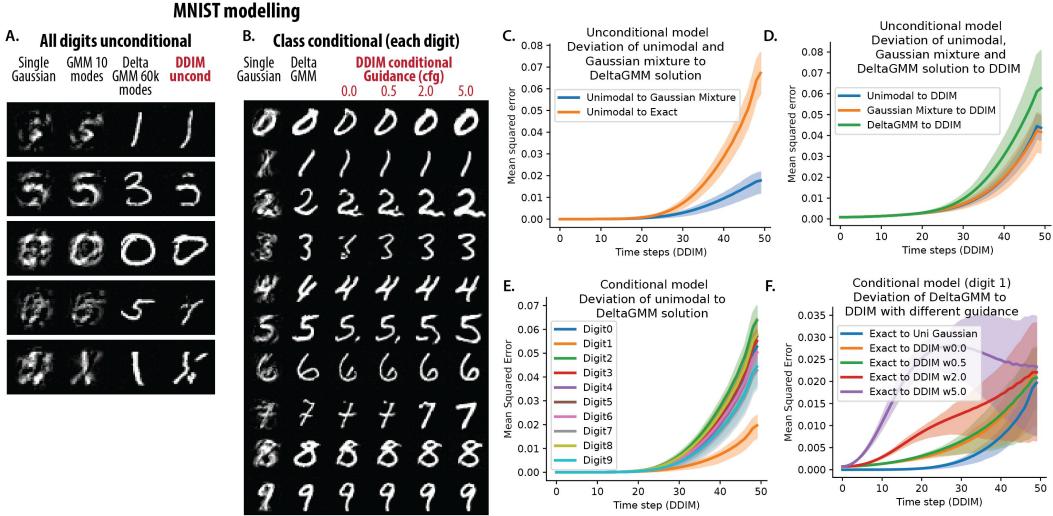


Figure 16: **Comparing the predictions of the Gaussian model, a 10-mode Gaussian mixture model, and the exact (delta function mixture) score function to real MNIST reverse diffusion trajectories.** **A.** True $\hat{x}_0(x_t)$ and $\hat{x}_0(x_t)$ predicted by various score models, all using the same initial conditions. **B.** Collections of actual and theory-predicted x_0 . **C.** Mean squared error between the actual trajectory x_t and trajectory predicted by each model. **D.** Fraction of squared error as a function of PC, between the actual and predicted x_0 .

B.5 Acceleration results with EDM model and Heun’s sampler

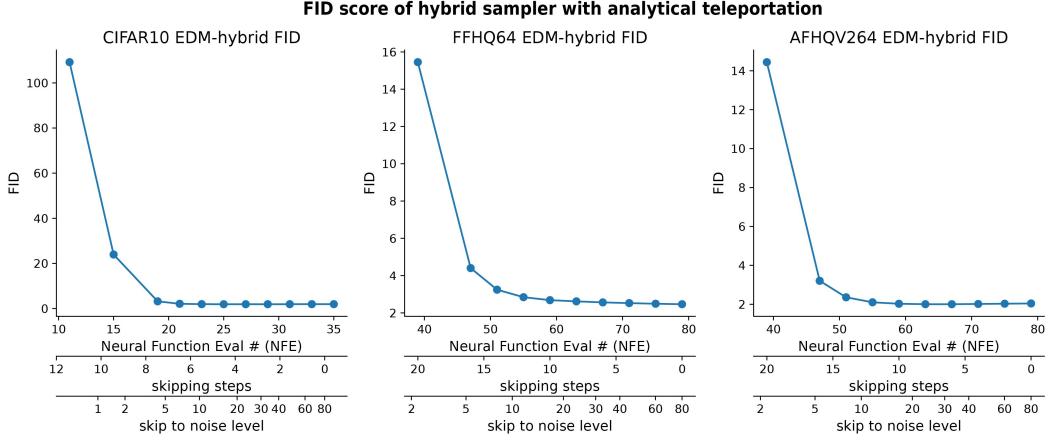


Figure 17: **Image quality as a function of skipping steps for hybrid sampling approach.** Note the main x-axes are the number of Neural Function Evaluation (NFE); the secondary x-axes are the number of skipping steps from the Heun sampler; the tertiary-axes are the time or noise level σ_{skip} at which we evaluate the Gaussian solution. See Tab.5,3,4 for numbers.

Table 3: FFHQ64 FID with analytical teleportation

Nskip	NFE	time/noise scale	FID
0	79	80.0	2.464
2	75	60.1	2.489
4	71	44.6	2.523
6	67	32.7	2.561
8	63	23.6	2.617
10	59	16.8	2.681
12	55	11.7	2.841
14	51	8.0	3.243
16	47	5.4	4.402
20	39	2.2	15.451

Table 4: AFHQV264 FID with analytical teleportation

Nskip	NFE	time/noise scale	FID
0	79	80.0	2.043
2	75	60.1	2.029
4	71	44.6	2.016
6	67	32.7	2.003
8	63	23.6	2.005
10	59	16.8	2.026
12	55	11.7	2.102
14	51	8.0	2.359
16	47	5.4	3.206
20	39	2.2	14.442

Table 5: CIFAR10 FID with analytical teleportation

Nskip	NFE	time/noise scale	FID
0	35	80.0	1.958
1	33	57.6	1.955
2	31	40.8	1.949
3	29	28.4	1.940
4	27	19.4	1.932
5	25	12.9	1.934
6	23	8.4	1.963
7	21	5.3	2.123
8	19	3.3	3.213
10	15	1.1	23.947
12	11	0.3	109.178

B.6 Principal dimensions of Stable Diffusion sampling trajectories

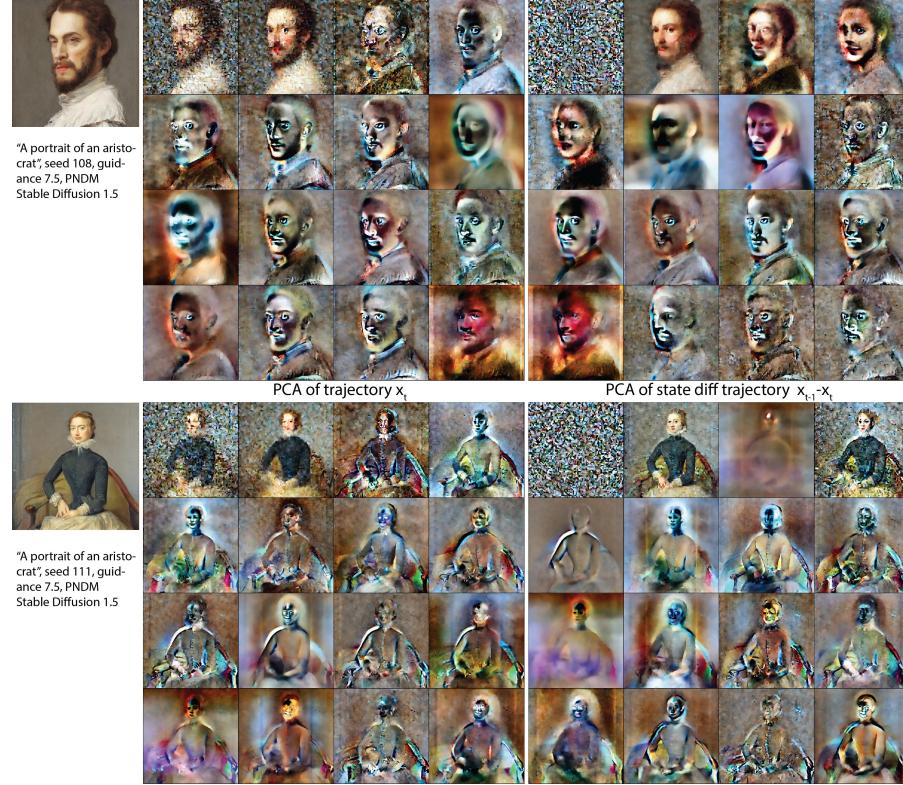


Figure 18: **Visualizing PCs of Stable Diffusion trajectories.** We computed the principal components of the trajectory \mathbf{x}_t and the trajectory difference $\mathbf{x}_{t-1} - \mathbf{x}_t$, and visualized the scaled version of PC vectors through the decoder. We can see that they represent an interpretable vector space around the actual sample \mathbf{x}_0 .

B.7 Additional perturbation experiments



Figure 19: **Example perturbation experiment result.** Different rows represent different perturbation times, from top to bottom 5, 10, ...50. Different columns represent different perturbation scales, left to right: negative to positive, $-20, -15, \dots, 15, 20$. Perturbations are along PC6 of U-Net output. Stable Diffusion, PNDM sampler, seed 100. Prompt: “a portrait of an aristocrat”.



Figure 20: **Example perturbation experiment result.** Different rows represent different perturbation times, from top to bottom 5, 10, ...50. Different columns represent different perturbation scales, left to right: negative to positive, $-20, -15, \dots, 15, 20$. Perturbations are along PC5 of U-Net output. Stable Diffusion, PNDM sampler, seed 101. Prompt: “a portrait of an aristocrat”.

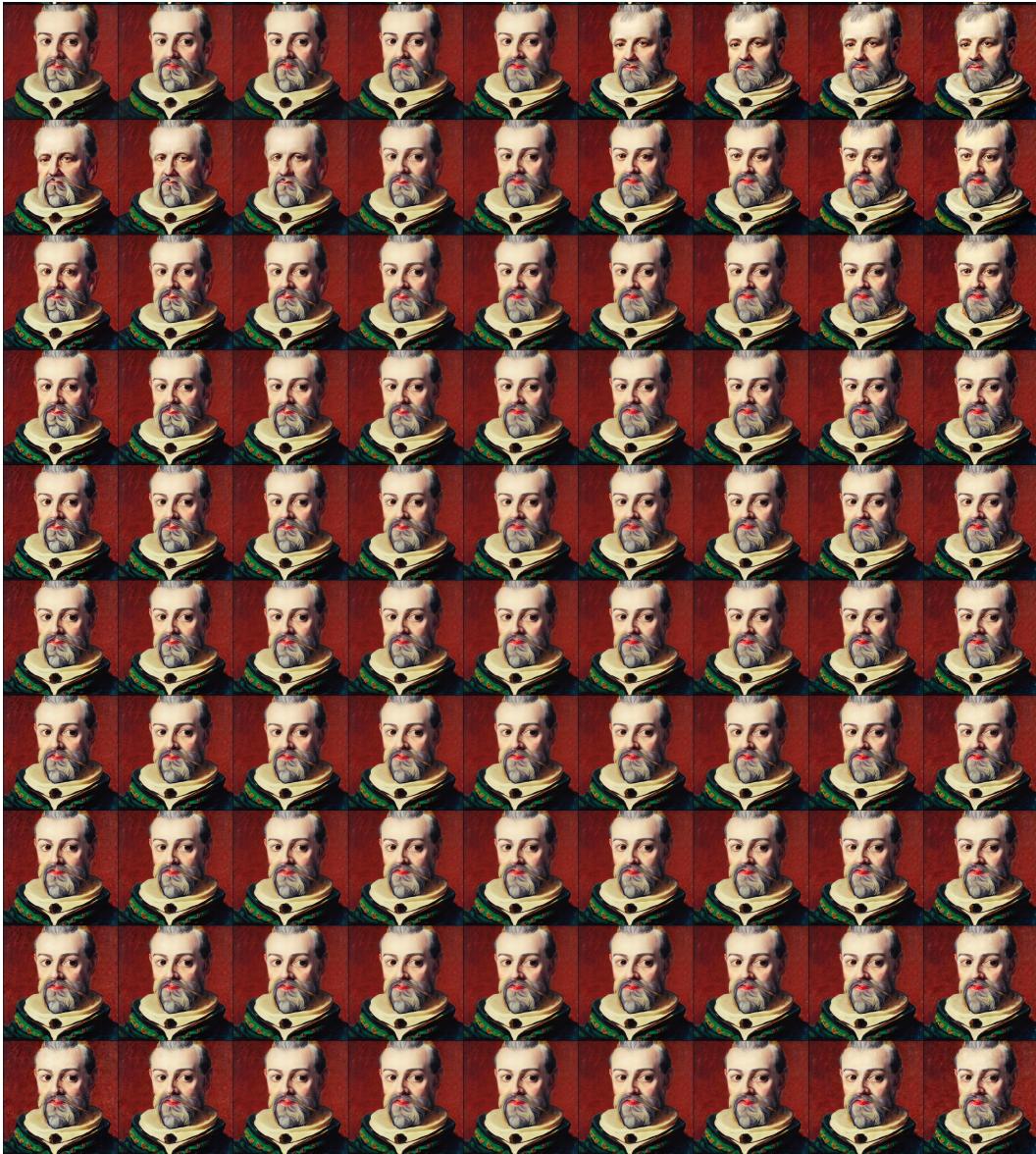


Figure 21: **Example perturbation experiment result.** Different rows represent different perturbation times, from top to bottom 5, 10, ...50. Different columns represent different perturbation scales, left to right: negative to positive, $-20, -15, \dots 15, 20$. Perturbations are along a random noise direction, which is less effective than the previous PC perturbation. Stable Diffusion, PNDM sampler, seed 101. Prompt: “a portrait of an aristocrat”.

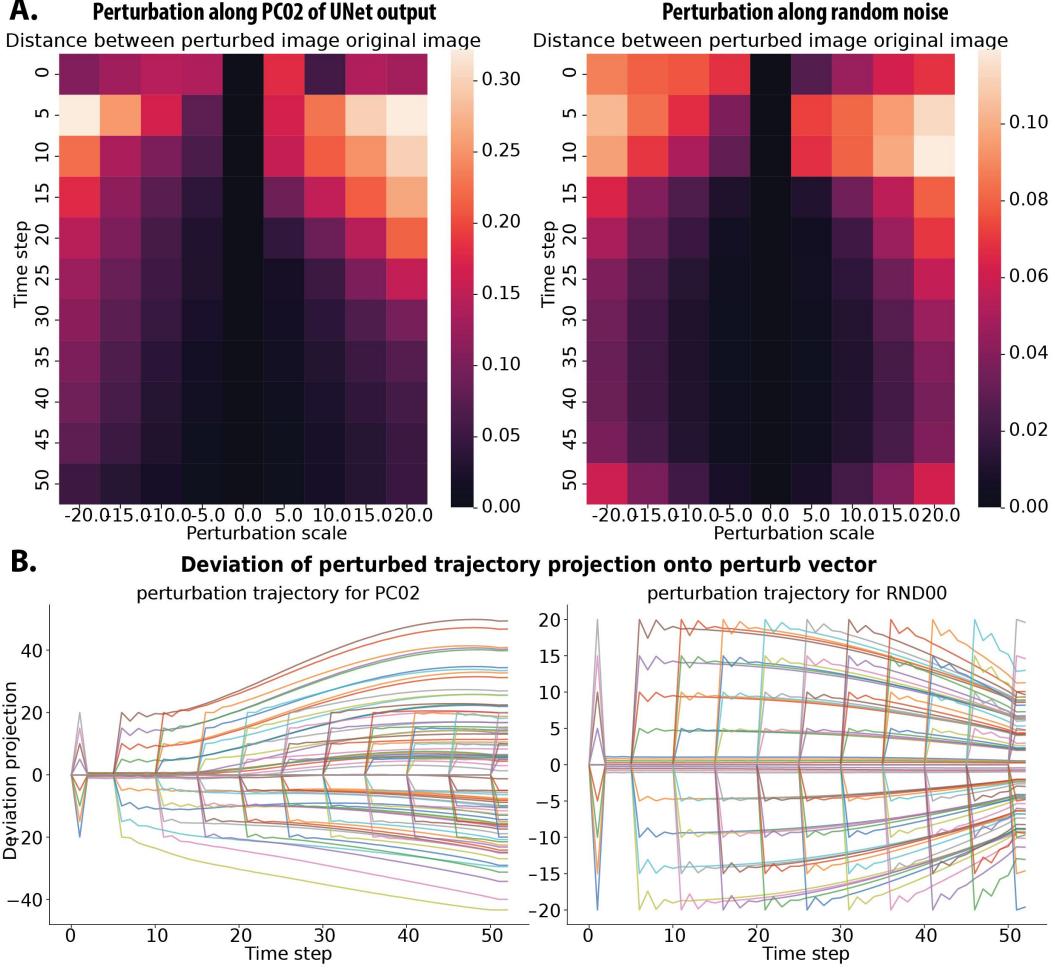


Figure 22: **Geometry of trajectory perturbation.** **A.** Quantification of sample deviation due to a perturbation at time t' and strength K , heatmap encodes the perceptual image similarity per LPIPS. Left panel: perturbation along PC02, Right panel: perturbation along a random vector. The perturbation along PC02 is much more effective than the random pattern in affecting the sample. **B.** The underlying geometry, projecting the difference between the perturbed trajectory and the original trajectory onto the unit perturbation vector. As we predicted, the perturbation along the signal manifold gets amplified, while the effect of a random perturbation decays over time. (14)

B.8 Impact of design choices on the geometry of diffusion

The field of diffusion generative modeling has greatly advanced since the DDIM paper [11]. Here, we consider how making slightly different design choices (e.g. using different samplers) affects the geometry and dynamics of sampling trajectories

B.8.1 Differential equation solver

Stable Diffusion by default uses the deterministic PNDM sampler, but one can use other solvers. Because the choice of solver slightly modifies the size and direction of individual time steps, changing the solver is akin to doing an early perturbation on the trajectory. Consistent with this expectation, we observe a variety of effects on image generation with a fixed random seed and different solvers: sometimes there is effectively no difference (e.g. DDIM and PNDM), sometimes there is a slight difference, and sometimes there is a major difference (e.g. PNDM vs LMSDiscrete Solver) (Fig.B.8.2). As for trajectory geometry, usually, DDIM, PNDM, and DPMSolverMultistep create rotation-like 2D trajectories well-predicted by our theory, while LMSDiscrete and EulerDiscrete create more linear 1D trajectories.

B.8.2 Classifier-free guidance strength

Classifier-free guidance [23] has been widely used in conditional diffusion models as a method to generate samples highly aligned with the conditional signal (e.g. prompt). We examined the effect of the strength of classifier-free guidance on the geometry of trajectories. We found that, generally, a higher guidance scale generates trajectories x_t and trajectory differences Δx_t with higher dimensionality (Fig.24 top). Furthermore, when visualizing the top PC vectors, a larger number of interpretable PC dimensions can be found for a higher guidance scale (Fig. 24 bottom). We also observed that with a smaller guidance value, the trajectory is usually smooth; with a higher guidance value, it induced strong oscillatory movement in the trajectory at the early phase, when combined with certain higher-order schedulers e.g. the default solver PNDM [12] (Fig. 24 middle). This effect could be a feature for the sampler to explore the landscape more. It could also be an artifact, which could be fixed by modifying the sampler.



Figure 23: **Effect of classifier-free guidance (cfg) strength and diffusion sampler on the sample.**

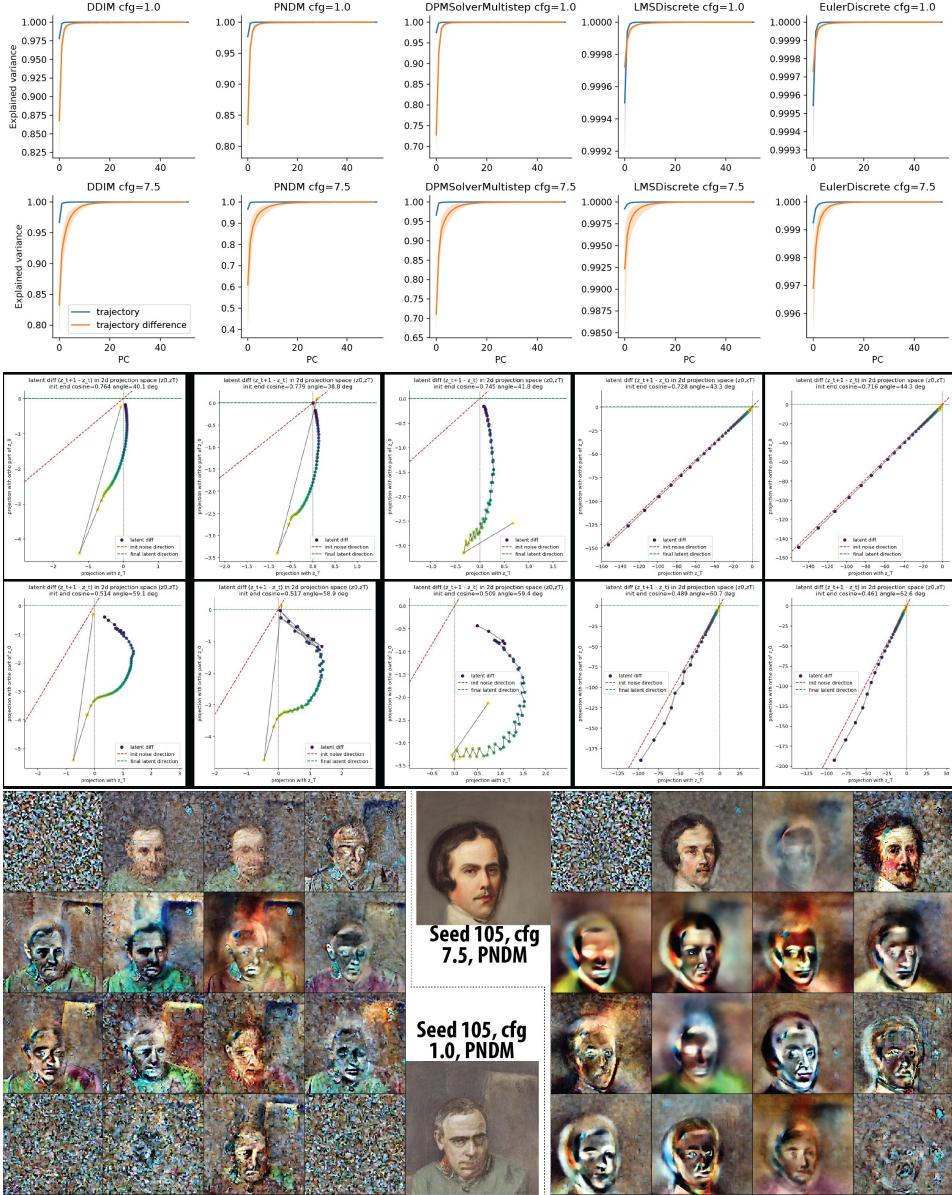


Figure 24: Effect of classifier-free guidance strength and sampler on trajectory geometry. **Top panel:** Dimensionality of trajectory \mathbf{x}_t and state difference $\Delta \mathbf{x}_t$, measured by explained variance of PCs. Higher guidance induces higher dimensionality in \mathbf{x}_t and $\Delta \mathbf{x}_t$. **Middle panel:** Trajectory difference $\Delta \mathbf{x}_t$ projected onto the $\mathbf{x}_0, \mathbf{x}_T$ plane. Higher guidance induced oscillation in the search trajectory, esp. for PNDM sampler. **Bottom panel:** Comparing the top 16 PCs of state difference $\Delta \mathbf{x}_t$, for high guidance (7.5) versus low guidance (1.0) trajectory from the same noise seed. The samples are shown in the middle. Trajectories sampled with higher guidance have more ‘on-manifold’ PC dimensions.

C Endpoint estimate trajectory examples



Figure 25: **Example endpoint estimate trajectory.** CelebA-HQ, DDIM sampler, seed 129.

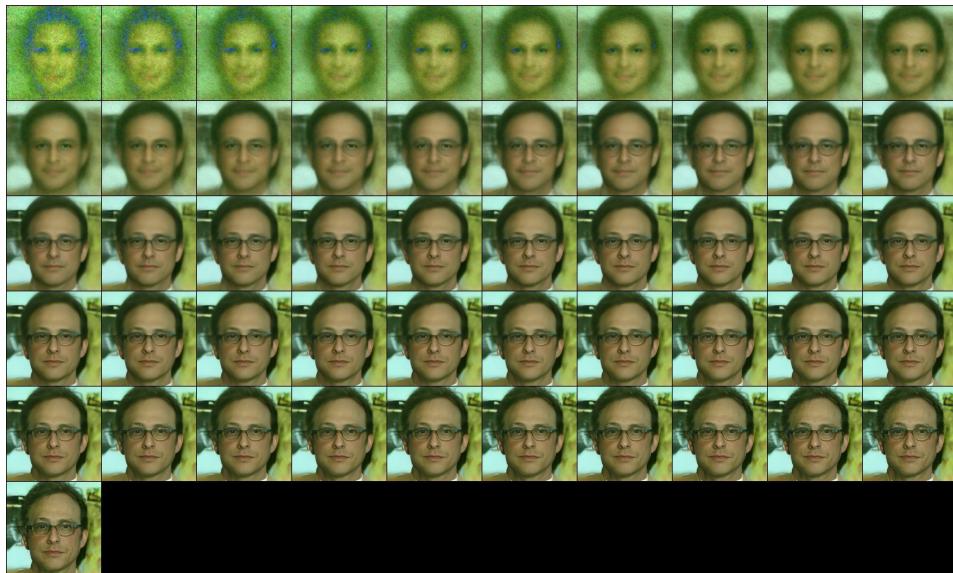


Figure 26: **Example endpoint estimate trajectory.** CelebA-HQ, DDIM sampler, seed 152.



Figure 27: **Example endpoint estimate trajectory.** Stable Diffusion, PNDM sampler, seed 101.
Prompt: “a portrait of an aristocrat”.

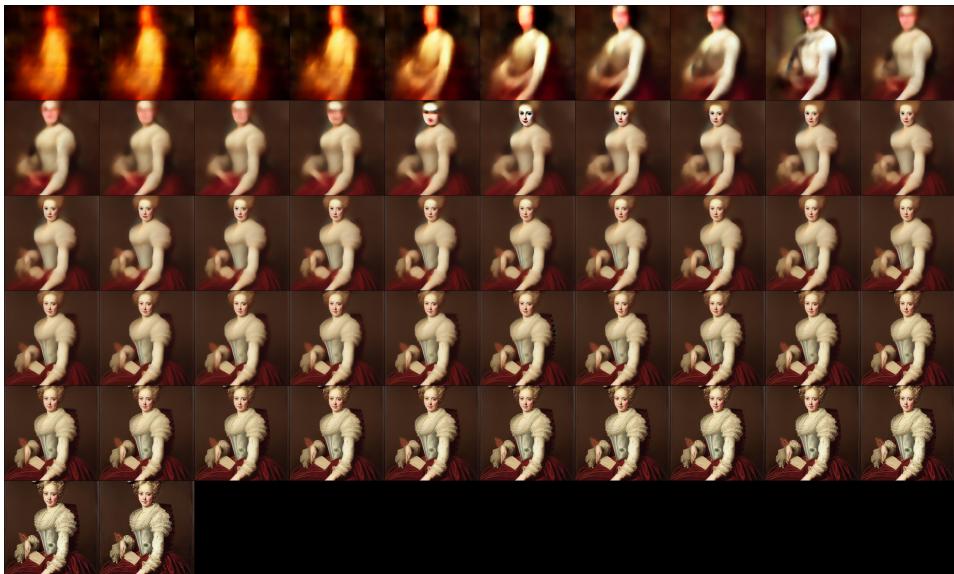


Figure 28: **Example endpoint estimate trajectory.** Stable Diffusion, PNDM sampler, seed 107.
Prompt: “a portrait of an aristocrat”.

D Notation correspondence

Diffusion models usually have forward processes whose conditional probabilities are

$$p(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(A_t \mathbf{x}_0, B_t \mathbf{I}) \quad (17)$$

for all $t \in [0, T]$. In the limit of small time steps, the transition probability distribution can be captured by the SDE

$$\dot{\mathbf{x}}_t = -C_t \mathbf{x}_t + D_t \eta(t) \quad (18)$$

where $\eta(t)$ is a vector of independent Gaussian white noise terms.

Papers discussing these models may use slightly different notation. In the table below, we briefly indicate how various choices of notation correspond to one another. To make comparing discrete and continuous models easier, we assume the time step size is $\Delta t = 1$.

Table 6: Comparison of notation for diffusion model parameters.

PAPER	CITATION	A_t	B_t	C_t	D_t
DDPM	[4]	$\sqrt{\alpha_t}$	$1 - \bar{\alpha}_t$	$1 - \sqrt{1 - \beta_t}$	$\sqrt{\beta_t}$
DDIM	[11]	$\sqrt{\alpha_t}$	$1 - \alpha_t$	$1 - \sqrt{\alpha_t / \alpha_{t-1}}$	$\sqrt{1 - \alpha_t / \alpha_{t-1}}$
STABLE DIFF.	[13]	α_t	σ_t^2	$1 - \alpha_t / \alpha_{t-1}$	$\sqrt{\sigma_t^2 - (\alpha_t / \alpha_{t-1})^2 \sigma_{t-1}^2}$
VP SDE	[3]	$\exp\left[-\frac{1}{2} \int_0^t \beta(s) ds\right]$	$1 - \exp\left[-\int_0^t \beta(s) ds\right]$	$\beta(t)/2$	$\sqrt{\beta(t)}$
OURS		α_t	σ_t^2	$\beta(t)$	$g(t)$

In the popular huggingface `diffusers` library implementation of diffusion models, the function `alphas_cumprod` corresponds to our α_t^2 .

E Details of the numerical simulation of solution

To sharpen our intuition about our analytical results (especially Eq. 9 and 13), we used a common α_t schedule (see Appendix A) and plotted scaled projection coefficients $\bar{c}_k(t) := c_k(t)/c_k(T)$ and $d(t)$ for different λ_k , i.e. the variance of the distribution in the direction \mathbf{u}_k . The projection coefficients of \mathbf{x}_t along large variance directions increase at first, while the coefficients along the low variance directions remain the same or shrink (Fig. 3A).

The derivative $\dot{\mathbf{x}}_t$, which indicates the direction of particle movement, is initially dominated by contributions from the large variance dimensions. Close to the end, it features comparable but opposite-sign contributions from the high variance dimensions and effectively off-manifold (i.e. noise) directions (Fig. 3B). This explains the observation that state differences $\mathbf{x}_{t-k} - \mathbf{x}_t$ look interpretable at first, and later look like noise-contaminated images (Fig. 1A, middle row).

The projection coefficients of the mean-adjusted endpoint estimate $\hat{\mathbf{x}}_0(\mathbf{x}_t) - \mu$, when normalized by their standard deviation $\sqrt{\lambda_k}$, look sigmoidal (Fig. 3C). The coefficients converge to their final value earlier along large variance dimensions, and later along small variance dimensions. The time derivatives of these coefficients are bump-like, and peak earlier along high variance dimensions. In plain language, high variance features are added to the endpoint estimate first, and low variance features are added later. This partly explains our earlier observations about the order of feature emergence (Fig. 1A, bottom row).

F Fréchet inception distance score for assessing generated image quality

The Fréchet inception distance (FID) score [24] provides one way to assess the quality of images produced by generative models, including GANs and diffusion models. In this paper, we have used it to assess the effect of using our Gaussian solution to skip some number of initial reverse diffusion steps on generated images (e.g. on a model of CIFAR-10; see Fig. 4); we found that image quality begins to seriously suffer when the number of skipped steps becomes somewhat larger than 30, although the exact point varies for models trained on different data sets.

The idea behind the FID score is the following. We would like the distribution of generated images $p(\cdot)$ to be similar to some distribution of images $p_w(\cdot)$ in the world. Hence, one way to assess image quality is via a measure that quantifies the difference between these distributions—or between the distributions of suitably transformed images. The Fréchet distance is one such measure, but it is difficult to compute in general.

We will need two facts. First, the Fréchet distance can be computed analytically when the distributions being compared are Gaussian. Second, it is possible to apply transformations to images that make their distribution approximately Gaussian, and in particular, deep networks that perform tasks like object recognition well are known to do this.

The algorithm for implementing the FID score, then, is the following. (1) Transform a set of real and generated images using some nonlinear function. (2) Fit Gaussians to both distributions. (This can be done simply by computing the mean and covariance of each distribution.) (3) Compute the Fréchet distance between the two Gaussians. The specific formula used to compare Gaussians $\mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ and $\mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$ is

$$d = \|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|_2^2 + \text{Tr} \left[\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2 - 2 \left(\boldsymbol{\Sigma}_1^{1/2} \boldsymbol{\Sigma}_2 \boldsymbol{\Sigma}_1^{1/2} \right)^{1/2} \right]. \quad (19)$$

Lower FID scores are interpreted as indicating that generated images are of higher quality.

We computed it with the `fidelity` function from the `torch-fidelity` library. We also confirmed it with the `fid.py` script from the official repository of [5] (<https://github.com/NVlabs/edm>). In these implementations, images are transformed using the penultimate layer of the Inception V3 model [25] trained on ImageNet for object classification. For each sampling method, we sampled 50,000 images from the same initial states \mathbf{x}_T generated by the random seeds, 1-50,000. The FID score is computed by comparing these 50,000 samples and the 50,000 training set images of CIFAR-10.

G Derivation of exact solution to Gaussian score model

In this section, we derive the analytic solution \mathbf{x}_t for the reverse diffusion trajectory of the Gaussian score model. As in Sec. 4 of the main text, we will assume throughout that the score function corresponds to a Gaussian image distribution whose mean is $\boldsymbol{\mu}$ and covariance matrix is Σ . We will also assume, given that images are usually thought of as residing on a low-dimensional manifold within pixel space, that the rank r of the covariance matrix may be less than the dimensionality D of state space.

Let $\Sigma = \mathbf{U}\Lambda\mathbf{U}^T$ be the eigendecomposition or compact SVD of the covariance matrix, where \mathbf{U} is a $D \times r$ semi-orthogonal matrix whose columns are normalized (i.e. $\mathbf{U}^T\mathbf{U} = \mathbf{I}_r$), and Λ is the $r \times r$ diagonal eigenvalue matrix. Denote the k th column of \mathbf{U} by \mathbf{u}_k and the k th diagonal element of Λ by λ_k .

For a Gaussian score model, the probability flow ODE that reverses a VP-SDE forward process is

$$\dot{\mathbf{x}} = -\beta(t)\mathbf{x} - \frac{1}{2}g^2(t)(\sigma_t^2\mathbf{I} + \alpha_t^2\Sigma)^{-1}(\alpha_t\boldsymbol{\mu} - \mathbf{x}). \quad (20)$$

Using the decomposition of Σ described above,

$$\dot{\mathbf{x}} = -\beta(t)\mathbf{x} - \frac{1}{2}g^2(t)\frac{1}{\sigma_t^2}(\mathbf{I} - \mathbf{U}\tilde{\Lambda}_t\mathbf{U}^T)(\alpha_t\boldsymbol{\mu} - \mathbf{x}) \quad (21)$$

where $\tilde{\Lambda}_t$ is defined to be the time-dependent diagonal matrix

$$\tilde{\Lambda}_t = \text{diag}\left[\frac{\alpha_t^2\lambda_k}{\alpha_t^2\lambda_k + \sigma_t^2}\right]. \quad (22)$$

Consider the dynamics of the quantity $\mathbf{x}_t - \alpha_t\boldsymbol{\mu}$. Using the relationship between β_t and α_t , we have

$$\frac{d}{dt}(\mathbf{x}_t - \alpha_t\boldsymbol{\mu}) = \dot{\mathbf{x}}_t - \boldsymbol{\mu}\dot{\alpha}_t \quad (23)$$

$$= \dot{\mathbf{x}}_t + \beta_t\alpha_t\boldsymbol{\mu} \quad (24)$$

$$= \beta_t(\alpha_t\boldsymbol{\mu} - \mathbf{x}) - \frac{1}{2}g^2(t)\frac{1}{\sigma_t^2}(\mathbf{I} - \mathbf{U}\tilde{\Lambda}_t\mathbf{U}^T)(\alpha_t\boldsymbol{\mu} - \mathbf{x}) \quad (25)$$

$$= \left[\frac{1}{2}g^2(t)\frac{1}{\sigma_t^2}(\mathbf{I} - \mathbf{U}\tilde{\Lambda}_t\mathbf{U}^T) - \beta_t\mathbf{I}\right](\mathbf{x} - \alpha_t\boldsymbol{\mu}). \quad (26)$$

If we assume that the forward process is a variance-preserving SDE, then $\beta_t = \frac{1}{2}g^2(t)$, which implies $\alpha_t^2 = 1 - \sigma_t^2$. Using this, we obtain

$$\frac{d}{dt}(\mathbf{x}_t - \alpha_t\boldsymbol{\mu}) = \beta_t\left[\frac{1}{\sigma_t^2}(\mathbf{I} - \mathbf{U}\tilde{\Lambda}_t\mathbf{U}^T) - \mathbf{I}\right](\mathbf{x} - \alpha_t\boldsymbol{\mu}) \quad (27)$$

$$= \beta_t\left[\left(\frac{1}{\sigma_t^2} - 1\right)\mathbf{I} - \frac{1}{\sigma_t^2}\mathbf{U}\tilde{\Lambda}_t\mathbf{U}^T\right](\mathbf{x} - \alpha_t\boldsymbol{\mu}). \quad (28)$$

Define the variable $\mathbf{y}_t := \mathbf{x}_t - \alpha_t\boldsymbol{\mu}$. We have just shown that its dynamics are fairly ‘nice’, in the sense that the above equation is well-behaved separable linear ODE. As we are about to show, it is exactly solvable.

Write \mathbf{y}_t in terms of the orthonormal columns of \mathbf{U} and a component that lies entirely in the orthogonal space \mathbf{U}^\perp :

$$\mathbf{y}_t = \mathbf{y}^\perp(t) + \sum_{k=1}^r c_k(t)\mathbf{u}_k, \quad \mathbf{y}^\perp(t) \in \mathbf{U}^\perp. \quad (29)$$

The dynamics of the coefficient $c_k(t)$ attached to the eigenvector \mathbf{u}_k are

$$\dot{c}_k(t) = \frac{d}{dt}(\mathbf{u}_k^T\mathbf{y}_t) = \beta_t\left[\left(\frac{1}{\sigma_t^2} - 1\right) - \frac{1}{\sigma_t^2}\frac{\alpha_t^2\lambda_k}{\alpha_t^2\lambda_k + \sigma_t^2}\right](\mathbf{u}_k^T\mathbf{y}_t) \quad (30)$$

$$= \frac{\beta_t}{\sigma_t^2}\left(1 - \sigma_t^2 - \frac{\alpha_t^2\lambda_k}{\alpha_t^2\lambda_k + \sigma_t^2}\right)c_k(t) \quad (31)$$

$$= \frac{\beta_t\alpha_t^2}{\sigma_t^2}\left(1 - \frac{\lambda_k}{\alpha_t^2\lambda_k + \sigma_t^2}\right)c_k(t). \quad (32)$$

Using the constraint that $\alpha_t^2 + \sigma_t^2 = 1$, this becomes

$$\dot{c}_k(t) = \frac{\beta_t \alpha_t^2 (1 - \lambda_k)}{\alpha_t^2 \lambda_k + \sigma_t^2} c_k(t). \quad (33)$$

For the orthogonal space component $\mathbf{y}^\perp(t)$, it will stay in the orthogonal space \mathbf{U}^\perp , and more specifically the 1D space spanned by the initial $\mathbf{y}^\perp(t)$ —so, when going backward in time, its dynamics is simply a downscaling of $\mathbf{y}^\perp(T)$.

$$\dot{\mathbf{y}}^\perp(t) = \beta_t \left(\frac{1}{\sigma_t^2} - 1 \right) \mathbf{y}^\perp(t) = \beta_t \frac{1 - \sigma_t^2}{\sigma_t^2} \mathbf{y}^\perp(t) = \frac{\beta_t \alpha_t^2}{\sigma_t^2} \mathbf{y}^\perp(t). \quad (34)$$

Combining these two results and solving the ODEs in the usual way, we have the trajectory solution

$$\mathbf{y}_t = d(t) \mathbf{y}^\perp(T) + \sum_{k=1}^r c_k(t) \mathbf{u}_k \quad (35)$$

$$d(t) = \exp \left(\int_T^t d\tau \frac{\beta_\tau \alpha_\tau^2}{\sigma_\tau^2} \right) \quad (36)$$

$$c_k(t) = c_k(T) \exp \left(\int_T^t d\tau \frac{\beta_\tau \alpha_\tau^2 (1 - \lambda_k)}{\alpha_\tau^2 \lambda_k + \sigma_\tau^2} \right). \quad (37)$$

The initial conditions are

$$c_k(T) = \mathbf{u}_k^T \mathbf{y}_T \quad (38)$$

$$\mathbf{y}^\perp(T) = \mathbf{y}_T - \sum_{k=1}^r c_k(T) \mathbf{u}_k, \quad \mathbf{y}^\perp(T) \in \mathbf{U}^\perp. \quad (39)$$

To solve the ODEs, it is helpful to use a particular reparameterization of time. In particular, consider a reparameterization in terms of α_t using the relationship $-\beta_t \alpha_t dt = d\alpha_t$. The integral we must do is

$$\int_T^t d\tau \frac{\beta_\tau \alpha_\tau^2 (1 - \lambda_k)}{\alpha_\tau^2 \lambda_k + \sigma_\tau^2} = \int_T^t d\tau \frac{\beta_\tau \alpha_\tau^2 (1 - \lambda_k)}{1 + \alpha_\tau^2 (\lambda_k - 1)} \quad (40)$$

$$= \int_{\alpha_T}^{\alpha_t} d\alpha_\tau \frac{\alpha_\tau (\lambda_k - 1)}{1 + \alpha_\tau^2 (\lambda_k - 1)} \quad (41)$$

$$= \frac{1}{2} \log(1 + \alpha_\tau^2 (\lambda_k - 1)) \Big|_{\alpha_T}^{\alpha_t} \quad (42)$$

$$= \frac{1}{2} \log \left(\frac{1 + (\lambda_k - 1)\alpha_t^2}{1 + (\lambda_k - 1)\alpha_T^2} \right). \quad (43)$$

Note that taking $\lambda_k = 0$ gives us the solution to dynamics in the directions orthogonal to the manifold. We have

$$c_k(t) = c_k(T) \sqrt{\frac{1 + (\lambda_k - 1)\alpha_t^2}{1 + (\lambda_k - 1)\alpha_T^2}} \quad (44)$$

$$d(t) = \sqrt{\frac{1 - \alpha_t^2}{1 - \alpha_T^2}}. \quad (45)$$

The time derivatives of these coefficients are

$$\dot{c}_k(t) = c_k(T) \frac{-(\lambda_k - 1)\alpha_t^2 \beta_t}{\sqrt{(1 + (\lambda_k - 1)\alpha_T^2)(1 + (\lambda_k - 1)\alpha_t^2)}} \quad (46)$$

$$\dot{d}(t) = \frac{\alpha_t^2 \beta_t}{\sqrt{(1 - \alpha_T^2)(1 - \alpha_t^2)}}. \quad (47)$$

Finally, we can write out the explicit solution for the trajectory \mathbf{x}_t :

$$\mathbf{x}_t = \alpha_t \boldsymbol{\mu} + d(t) \mathbf{y}^\perp(T) + \sum_{k=1}^r c_k(t) \mathbf{u}_k. \quad (48)$$

We can see that there are three terms: 1) $\alpha_t \boldsymbol{\mu}$, an increasing term that scales up to the mean $\boldsymbol{\mu}$ of the distribution; 2) $d(t)\mathbf{y}^\perp(T)$, a decaying term downscaling the residual part of the initial noise vector, which is orthogonal to the data manifold; and 3) the $c_k(t)\mathbf{u}_k$ sum, each term of which has independent dynamics.

We also now have the analytical solution for the projected outcome:

$$\begin{aligned}\hat{\mathbf{x}}_0(\mathbf{x}_t) - \boldsymbol{\mu} &= \frac{1}{\alpha_t} \mathbf{U} \tilde{\boldsymbol{\Lambda}}_t \mathbf{U}^T (\mathbf{x}_t - \alpha_t \boldsymbol{\mu}) \\ &= \sum_{k=1}^r c_k(t) \frac{\alpha_t \lambda_k}{\alpha_t^2 \lambda_k + \sigma_t^2} \mathbf{u}_k \\ &= \sum_{k=1}^r c_k(T) \frac{\alpha_t \lambda_k}{\sqrt{(\alpha_t^2 \lambda_k + \sigma_t^2)(\alpha_T^2 \lambda_k + \sigma_T^2)}} \mathbf{u}_k.\end{aligned}\tag{49}$$

Similarly, we can write out the exact formula for the trajectory's tangent vector $\dot{\mathbf{x}}_t$:

$$\begin{aligned}\dot{\mathbf{x}}_t &= \dot{\alpha}_t \boldsymbol{\mu} + d'(t) \mathbf{y}^\perp(T) + \sum_{k=1}^r \dot{c}_k(t) \mathbf{u}_k \\ &= -\alpha_t \beta_t \boldsymbol{\mu} + \frac{\alpha_t^2 \beta_t}{\sqrt{(1 - \alpha_T^2)(1 - \alpha_t^2)}} \mathbf{y}^\perp(T) - \sum_{k=1}^r c_k(T) \frac{(\lambda_k - 1) \alpha_t^2 \beta_t}{\sqrt{(1 + (\lambda_k - 1) \alpha_T^2)(1 + (\lambda_k - 1) \alpha_t^2)}} \mathbf{u}_k.\end{aligned}\tag{50}$$

H Derivation of endpoint estimate properties

In this section, we study the endpoint estimate (or projected outcome) $\hat{\mathbf{x}}_0$ in the case of a Gaussian score model.

H.1 Projected outcome for a Gaussian image distribution

By definition, the endpoint estimate $\hat{\mathbf{x}}_0(\mathbf{x}_t)$ is

$$\hat{\mathbf{x}}_0(\mathbf{x}_t) = \frac{\mathbf{x}_t + \sigma_t^2 \mathbf{s}(\mathbf{x}_t, t)}{\alpha_t} = \frac{\mathbf{x}_t + \sigma_t^2 (\sigma_t^2 \mathbf{I} + \alpha_t^2 \boldsymbol{\Sigma})^{-1} (\alpha_t \boldsymbol{\mu} - \mathbf{x}_t)}{\alpha_t}. \quad (51)$$

Let's examine the difference between $\hat{\mathbf{x}}_0(\mathbf{x}_t)$ and the mean $\boldsymbol{\mu}$. We find that

$$\begin{aligned} \hat{\mathbf{x}}_0(\mathbf{x}_t) - \boldsymbol{\mu} &= \frac{1}{\alpha_t} [\mathbf{x}_t - \alpha_t \boldsymbol{\mu} + \sigma_t^2 (\sigma_t^2 \mathbf{I} + \alpha_t^2 \boldsymbol{\Sigma})^{-1} (\alpha_t \boldsymbol{\mu} - \mathbf{x}_t)] \\ &= \frac{\sigma_t^2}{\alpha_t} \left[\frac{1}{\sigma_t^2} \mathbf{I} - (\sigma_t^2 \mathbf{I} + \alpha_t^2 \boldsymbol{\Sigma})^{-1} \right] (\mathbf{x}_t - \alpha_t \boldsymbol{\mu}). \end{aligned} \quad (52)$$

What can we learn from this formula? We explore several consequences of it below.

The projected outcome is always exact for a delta function. If the initial distribution $p(\mathbf{x}_0)$ is a delta function, $\boldsymbol{\Sigma} \rightarrow 0$. Then at any time t , $p(\mathbf{x}_t) = \mathcal{N}(\alpha_t \boldsymbol{\mu}, \sigma_t^2 \mathbf{I})$, and the projected outcome is $\hat{\mathbf{x}}_0(\mathbf{x}_t) - \boldsymbol{\mu} \equiv 0, \forall \mathbf{x}_t, \forall t$. Hence, the projected outcome is always exact, regardless of the position of \mathbf{x}_t and the time t . This can be regarded as one justification for this statistic $\hat{\mathbf{x}}_0(\mathbf{x}_t)$: it is exact and invariant in the isotropic score field created by a point distribution.

This point may be relevant for understanding the very end of reverse diffusion dynamics, when \mathbf{x}_t is likely to live within a score field created by a single point. At such a time, $\hat{\mathbf{x}}_0(\mathbf{x}_t)$ scarcely changes.

Projected outcome for an isotropic Gaussian. When $\boldsymbol{\Sigma} = \bar{\sigma}^2 \mathbf{I}$, Eq. 52 takes a particularly simple form:

$$\hat{\mathbf{x}}_0(\mathbf{x}_t) - \boldsymbol{\mu} = \frac{1}{\alpha_t} \frac{\alpha_t^2 \bar{\sigma}^2}{\sigma_t^2 + \alpha_t^2 \bar{\sigma}^2} (\mathbf{x}_t - \alpha_t \boldsymbol{\mu}). \quad (53)$$

H.2 Low rank $\boldsymbol{\Sigma}$

Assume that the covariance matrix $\boldsymbol{\Sigma}$ has rank r somewhat less than D , the dimensionality of state space. This case is of particular interest, since images (as previously mentioned) images are often viewed as residing on low-dimensional manifolds. We can use the Woodbury matrix inversion identity to write

$$(\sigma_t^2 \mathbf{I} + \alpha_t^2 \boldsymbol{\Sigma})^{-1} = (\sigma_t^2 \mathbf{I} + \alpha_t^2 \mathbf{U} \boldsymbol{\Lambda} \mathbf{U}^T)^{-1} \quad (54)$$

$$= \frac{1}{\sigma_t^2} \mathbf{I} - \frac{1}{\sigma_t^4} \mathbf{U} \left(\frac{1}{\alpha_t^2} \boldsymbol{\Lambda}^{-1} + \frac{1}{\sigma_t^2} \mathbf{U}^T \mathbf{U} \right)^{-1} \mathbf{U}^T \quad (55)$$

$$= \frac{1}{\sigma_t^2} \mathbf{I} - \frac{1}{\sigma_t^4} \mathbf{U} \left(\frac{1}{\alpha_t^2} \boldsymbol{\Lambda}^{-1} + \frac{1}{\sigma_t^2} \mathbf{I}_r \right)^{-1} \mathbf{U}^T. \quad (56)$$

We can reuse the previously defined diagonal matrix

$$\tilde{\boldsymbol{\Lambda}}_t = \text{diag} \left[\frac{\alpha_t^2 \lambda_k}{\alpha_t^2 \lambda_k + \sigma_t^2} \right] \quad (57)$$

to write $(\sigma_t^2 \mathbf{I} + \alpha_t^2 \boldsymbol{\Sigma})^{-1}$ as

$$(\sigma_t^2 \mathbf{I} + \alpha_t^2 \boldsymbol{\Sigma})^{-1} = \frac{1}{\sigma_t^2} (\mathbf{I} - \mathbf{U} \tilde{\boldsymbol{\Lambda}}_t \mathbf{U}^T). \quad (58)$$

Using this result, we can write the endpoint estimate as

$$\hat{\mathbf{x}}_0(\mathbf{x}_t) - \boldsymbol{\mu} = \frac{\sigma_t^2}{\alpha_t} \left(\frac{1}{\sigma_t^2} \mathbf{I} - (\sigma_t^2 \mathbf{I} + \alpha_t^2 \boldsymbol{\Sigma})^{-1} \right) (\mathbf{x}_t - \alpha_t \boldsymbol{\mu}) = \frac{1}{\alpha_t} \mathbf{U} \tilde{\boldsymbol{\Lambda}}_t \mathbf{U}^T (\mathbf{x}_t - \alpha_t \boldsymbol{\mu}). \quad (59)$$

This formula has a series of interesting implications.

Projected outcome stays on image manifold. Note that the deviation of the endpoint estimate from the distribution mean $\hat{\mathbf{x}}_0(\mathbf{x}_t) - \boldsymbol{\mu}$ always remains in the subspace spanned by the columns of \mathbf{U} , i.e. $\hat{\mathbf{x}}_0(\mathbf{x}_t) - \boldsymbol{\mu} \in \text{span}(\mathbf{U})$. This is an interesting result: if the data distribution is a low-dimensional manifold (e.g. image manifold), the estimate $\hat{\mathbf{x}}_0$ will not deviate from this manifold. Even if the projected outcome does not exactly reflect the true outcome, it will not make errors out of the image manifold, i.e. orthogonal to high variance directions. Visually, such directions would correspond to random noise, and hence perturbations in those directions would be ‘nonsense’ perturbations to images.

Projected outcome starts around the center of the distribution. At the start of the reverse diffusion, $t \approx T$, $\alpha_t \approx 0$, and $\sigma_t^2 \gg \alpha_t^2$. This means $\tilde{\Lambda}_t \approx \mathbf{0}$ and $\hat{\mathbf{x}}_0(\mathbf{x}_t) - \boldsymbol{\mu} \approx \mathbf{0}$. Hence, the projected outcome $\hat{\mathbf{x}}_0$ initially corresponds to the center of the distribution. For multi-class unconditional generation, this center will be relatively class-ambiguous. For the unconditional generation of faces, we observed that $\hat{\mathbf{x}}_0(\mathbf{x}_T)$ points to a generic face, which is close to the ‘average face’.

Projected outcome ends in the real outcome. At the end of reverse diffusion, $t = 0$, $\sigma_t = 0$, and $\alpha_t = 1$. This means $\hat{\mathbf{x}}_0(\mathbf{x}_0) = \mathbf{x}_0$, i.e. the projected outcome corresponds to the real result of reverse diffusion.

Features emerge in descending order of their variance. Consider the projection of this vector $\hat{\mathbf{x}}_0(\mathbf{x}_t) - \boldsymbol{\mu}$ on an eigenvector \mathbf{u}_k of Σ :

$$\mathbf{u}_k^T(\hat{\mathbf{x}}_0(\mathbf{x}_t) - \boldsymbol{\mu}) = \frac{1}{\alpha_t} \frac{\alpha_t^2 \lambda_k}{\alpha_t^2 \lambda_k + \sigma_t^2} \mathbf{u}_k^T(\mathbf{x}_t - \alpha_t \boldsymbol{\mu}). \quad (60)$$

When $\sigma_t^2 \gg \alpha_t^2 \lambda_k$, i.e. when the noise scale is much larger than the signal scale, we have

$$\mathbf{u}_k^T(\hat{\mathbf{x}}_0(\mathbf{x}_t) - \boldsymbol{\mu}) \ll \frac{\mathbf{u}_k^T(\mathbf{x}_t - \alpha_t \boldsymbol{\mu})}{\alpha_t}, \quad (61)$$

so the projected outcome is approximately $\mathbf{0}$. At the other extreme, when $\sigma_t^2 \ll \alpha_t^2 \lambda_k$, the signal variance is much bigger than the noise variance, and we have

$$\mathbf{u}_k^T(\hat{\mathbf{x}}_0(\mathbf{x}_t) - \boldsymbol{\mu}) \approx \frac{\mathbf{u}_k^T(\mathbf{x}_t - \alpha_t \boldsymbol{\mu})}{\alpha_t}. \quad (62)$$

To interpret this, note that if we regard \mathbf{u}_k as a feature direction, then λ_k is the variance along this feature direction. The above equation tells us that this feature stays around the mean value of the distribution when the noise variance is much larger than the scaled variance of this feature. In plain language, when the signal scale along a certain dimension is less than the noise scale, the projected outcome along that dimension will remain undetermined.

Empirically, people have found that natural image space has spectra close to $1/f$ [16], which means more image variance exists in low-frequency features than in high-frequency features. Thus, we expect that in the generating process, the projected outcome $\hat{\mathbf{x}}_0(\mathbf{x}_t)$ will specify the low-frequency layout first, and then gradually add the high-frequency details.

Invariance of projected outcome to off-manifold perturbations. Consider a perturbation of \mathbf{x}_t at time t by $\delta \mathbf{x}$. The projected outcome will change by an amount

$$\hat{\mathbf{x}}_0(\mathbf{x}_t + \delta \mathbf{x}) - \hat{\mathbf{x}}_0(\mathbf{x}_t) = \frac{1}{\alpha_t} \mathbf{U} \tilde{\Lambda}_t \mathbf{U}^T \delta \mathbf{x}. \quad (63)$$

Notice that if \mathbf{x}_t is perturbed in a direction orthogonal to the manifold spanned by the columns of \mathbf{U} , the projected outcome $\hat{\mathbf{x}}_0$ will not change. Thus, perturbation in a random non-signal direction will not change the projected outcome of reverse diffusion (and it will also, as we will see, not affect the real result either). In contrast, if the perturbation is aligned with the image manifold, the perturbation *will* affect the projected outcome.

Effect of a perturbation decreases over time. As we can see from the formula above, the effect of the same perturbation changes over time. After $\alpha_t^2 \lambda_k \gg \sigma_t^2$, the term $\frac{\alpha_t^2 \lambda_k}{\alpha_t^2 \lambda_k + \sigma_t^2}$ is approximately 1, so $\frac{1}{\alpha_t} \frac{\alpha_t^2 \lambda_k}{\alpha_t^2 \lambda_k + \sigma_t^2}$ will decrease over time. We predict on this basis that after certain features emerge, perturbations along those dimensions will have a limited effect.

I Derivation of rotational dynamics

In this section, we derive various results quantifying how reverse diffusion trajectories are rotation-like. In particular, under certain assumptions, we will show that the dynamics of the state \mathbf{x}_t looks like a rotation within a 2D plane spanned by \mathbf{x}_0 (the reverse diffusion endpoint) and \mathbf{x}_T (the initial noise). We will derive the formula by assuming that the training set consists of a single Gaussian mode, but will explain why this assumption may not be strictly necessary.

I.1 Derivation of rotation formula and correction terms

Assume that reverse diffusion begins at time T with $\alpha_T \approx 0$, and ends at time $t = 0$ with $\alpha_0 = 1$. Using our exact solution for \mathbf{x}_t (Eq. 9), at some intermediate time t we have that

$$\mathbf{x}_t = \alpha_t \boldsymbol{\mu} + \sqrt{\frac{1 - \alpha_t^2}{1 - \alpha_T^2}} \mathbf{y}^\perp(T) + \sum_{k=1}^r \sqrt{\frac{1 + (\lambda_k - 1)\alpha_t^2}{1 + (\lambda_k - 1)\alpha_T^2}} c_k(T) \mathbf{u}_k. \quad (64)$$

It is also true, by substituting $t = 0$ and $t = T$, that

$$\begin{aligned} \boldsymbol{\mu} &= \mathbf{x}_0 - \sum_{k=1}^r \sqrt{\frac{\lambda_k}{1 + (\lambda_k - 1)\alpha_T^2}} c_k(T) \mathbf{u}_k \\ \mathbf{y}^\perp(T) &= \mathbf{x}_T - \alpha_T \boldsymbol{\mu} - \sum_{k=1}^r c_k(T) \mathbf{u}_k. \end{aligned} \quad (65)$$

Using these two equations, we can rewrite Eq. 64 as

$$\begin{aligned} \mathbf{x}_t &= \alpha_t \boldsymbol{\mu} + \sqrt{\frac{1 - \alpha_t^2}{1 - \alpha_T^2}} \left[\mathbf{x}_T - \alpha_T \boldsymbol{\mu} - \sum_{k=1}^r c_k(T) \mathbf{u}_k \right] + \sum_{k=1}^r \sqrt{\frac{1 + (\lambda_k - 1)\alpha_t^2}{1 + (\lambda_k - 1)\alpha_T^2}} c_k(T) \mathbf{u}_k \\ &= \left[\alpha_t - \alpha_T \sqrt{\frac{1 - \alpha_t^2}{1 - \alpha_T^2}} \right] \boldsymbol{\mu} + \sqrt{\frac{1 - \alpha_t^2}{1 - \alpha_T^2}} \mathbf{x}_T + \sum_{k=1}^r \left\{ \sqrt{\frac{1 + (\lambda_k - 1)\alpha_t^2}{1 + (\lambda_k - 1)\alpha_T^2}} - \sqrt{\frac{1 - \alpha_t^2}{1 - \alpha_T^2}} \right\} c_k(T) \mathbf{u}_k \\ &= \left[\alpha_t - \alpha_T \sqrt{\frac{1 - \alpha_t^2}{1 - \alpha_T^2}} \right] \mathbf{x}_0 + \sqrt{\frac{1 - \alpha_t^2}{1 - \alpha_T^2}} \mathbf{x}_T + \mathbf{R}_t \end{aligned} \quad (66)$$

where the remainder term \mathbf{R}_t is equal to

$$\mathbf{R}_t = \sum_{k=1}^r \left\{ \sqrt{\frac{1 + (\lambda_k - 1)\alpha_t^2}{1 + (\lambda_k - 1)\alpha_T^2}} - \sqrt{\frac{1 - \alpha_t^2}{1 - \alpha_T^2}} - \left[\alpha_t - \alpha_T \sqrt{\frac{1 - \alpha_t^2}{1 - \alpha_T^2}} \right] \sqrt{\frac{\lambda_k}{1 + (\lambda_k - 1)\alpha_T^2}} \right\} c_k(T) \mathbf{u}_k.$$

The expression simplifies somewhat if we take $\alpha_T \approx 0$. Doing so, we obtain the equation seen in the main text:

$$\mathbf{x}_t \approx \alpha_t \mathbf{x}_0 + \sqrt{1 - \alpha_t^2} \mathbf{x}_T + \sum_{k=1}^r \left\{ \sqrt{\sigma_t^2 + \lambda_k \alpha_t^2} - \alpha_t \sqrt{\lambda_k} - \sigma_t \right\} c_k(T) \mathbf{u}_k.$$

Let's examine the correction terms more closely. Define the function

$$\begin{aligned} J(\alpha_t; \lambda) &:= \sqrt{\sigma_t^2 + \lambda \alpha_t^2} - \alpha_t \sqrt{\lambda} - \sigma_t \\ &= \sqrt{1 + (\lambda - 1)\alpha_t^2} - \alpha_t \sqrt{\lambda} - \sqrt{1 - \alpha_t^2}. \end{aligned} \quad (67)$$

Note that we can rewrite J as

$$\begin{aligned} J(\alpha_t; \lambda) &= \frac{\left(\sqrt{\sigma_t^2 + \lambda \alpha_t^2} - \alpha_t \sqrt{\lambda} - \sigma_t \right) \left(\sqrt{\sigma_t^2 + \lambda \alpha_t^2} + \alpha_t \sqrt{\lambda} + \sigma_t \right)}{\sqrt{\sigma_t^2 + \lambda \alpha_t^2} + \alpha_t \sqrt{\lambda} + \sigma_t} \\ &= -2 \frac{\sigma_t \alpha_t \sqrt{\lambda}}{\sqrt{\sigma_t^2 + \lambda \alpha_t^2} + \alpha_t \sqrt{\lambda} + \sigma_t}. \end{aligned} \quad (68)$$

From this, it is immediately clear that the correction term is not completely arbitrary. First, J is always negative. Second, its time course is bowl-shaped: it begins close to zero (since $\alpha_T \approx 0$), becomes more negative, then ends close to zero (since $\sigma_0 \approx 0$). It achieves its most negative value roughly when σ_t and $\alpha_t\sqrt{\lambda}$ are comparable, i.e. when

$$\alpha_t \approx \sqrt{\frac{1}{\lambda + 1}}, \quad (69)$$

in which case

$$J(\alpha_t; \lambda) \approx -2 \left(1 - \frac{\sqrt{2}}{2}\right) \sqrt{\frac{\lambda}{\lambda + 1}} \geq -2 \left(1 - \frac{\sqrt{2}}{2}\right) \approx -0.586. \quad (70)$$

Notice that $|J| < 1$, regardless of λ .

Although we derived this formula by assuming a single Gaussian mode, its form does not actually depend on any properties of the mode. This suggests that, as long as the score function landscape looks *locally* Gaussian, the formula may still be applicable. For example, suppose the learned image distribution is a Gaussian mixture. Even though the mean and the covariance of the nearest mode—the one which we expect to dominate the score function—may regularly change throughout reverse diffusion, even in a discontinuous way, the rotation equation should stay the same.

I.2 Low-rank image distribution sufficient for small correction terms

Suppose that the rank r of the covariance matrix Σ is much less than D , the dimensionality of state space. The error in the rotation formula is

$$\left\| \mathbf{x}_t - \alpha_t \mathbf{x}_0 - \sqrt{1 - \alpha_t^2} \mathbf{x}_T \right\|_2^2 = \sum_{k=1}^r J(\alpha_t; \lambda_k)^2 c_k(T)^2. \quad (71)$$

Recall that $c_k(T)$ is the coefficient of the original noise seed $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ along the direction \mathbf{u}_k . Assuming D is large, the norm of the noise seed is approximately 1. Since there is a prior no relationship between \mathbf{x}_T and \mathbf{u}_k , we expect that $\mathbf{x}_T \cdot \mathbf{u}_k \approx 1/\sqrt{D}$. (Suppose we express \mathbf{x}_T in terms of a set of D orthonormal basis vectors. Given that its norm is 1, and that it has no special relationship with any basis vector, the overlap between \mathbf{x}_T and each vector must be about $1/\sqrt{D}$.) The error becomes

$$\left\| \mathbf{x}_t - \alpha_t \mathbf{x}_0 - \sqrt{1 - \alpha_t^2} \mathbf{x}_T \right\|_2^2 \approx \sum_{k=1}^r J(\alpha_t; \lambda_k)^2 \frac{1}{D} \leq \sum_{k=1}^r 4 \left(1 - \frac{\sqrt{2}}{2}\right)^2 \frac{1}{D} = 4 \left(1 - \frac{\sqrt{2}}{2}\right)^2 \frac{r}{D}$$

where we have used the bound from the previous subsection. Since $r \ll D$, this error is small.

It is worth noting, however, that the $r \ll D$ assumption is not *necessary* for the rotation formula correction terms to be negligible. Another case in which this is true is when the image distribution is isotropic, i.e. $\lambda_k = \lambda$ for all k . Then the error is

$$\left\| \mathbf{x}_t - \alpha_t \mathbf{x}_0 - \sqrt{1 - \alpha_t^2} \mathbf{x}_T \right\|_2^2 \approx 4 \left(1 - \frac{\sqrt{2}}{2}\right)^2 \frac{\lambda}{\lambda + 1} \frac{r}{D} < \frac{\lambda}{\lambda + 1}.$$

This is somewhat smaller than the typical scale of \mathbf{x}_t , since $\|\mathbf{x}_t\|_2^2$ remains roughly between 1 and λ .

I.3 Can the rotation formula be used to predict the trajectory endpoint?

Naively, since any two vectors are mathematically sufficient to define a plane, the rotation plane should be completely determined from the first two steps—or if not the first two steps, one might naively expect the first *several* steps to be sufficient. In particular, since

$$\mathbf{x}_t \approx \alpha_t \mathbf{x}_0 + \sqrt{1 - \alpha_t^2} \mathbf{x}_T + \mathbf{R}_t, \quad (72)$$

where \mathbf{R}_t is the correction term we derived earlier, we can approximate \mathbf{x}_0 as

$$\mathbf{x}_0 \approx \frac{\mathbf{x}_t - \sqrt{1 - \alpha_t^2} \mathbf{x}_T}{\alpha_t}. \quad (73)$$

The problem is that the correction term along each feature direction is ‘large’ until that feature has been ‘committed to’. Concretely, the correction term along direction \mathbf{u}_k is proportional to

$$\frac{J(\alpha_t; \lambda_k)}{\alpha_t} = -2 \frac{\sigma_t \sqrt{\lambda}}{\sqrt{\sigma_t^2 + \lambda \alpha_t^2 + \alpha_t \sqrt{\lambda} + \sigma_t}}. \quad (74)$$

This function has saturating behavior, and remains high (in the sense of being $\sim \sqrt{\lambda_k}$) until around the time when $\alpha_t \sqrt{\lambda_k} \approx \sqrt{1 - \alpha_t^2}$. But consistent with our other results (see Figure 3 and the associated formulas), this is roughly around the time of ‘feature commitment’, or more specifically when the sigmoid-shaped coefficients of $\hat{\mathbf{x}}_0$ begin to transition to their final value. So the rotation formula only becomes useful for determining the endpoint after enough time has passed that one is sufficiently close to the endpoint.

Using multiple \mathbf{x}_t does not help reduce the error in applying the rotation formula, since the error equation above is monotonic in time. In other words, averaging over multiple recent \mathbf{x}_t is *strictly worse* than just using the rotation formula and the most recent (i.e. greatest number of reverse diffusion time steps) \mathbf{x}_t .

As a final note: since we have the form of the correction terms, why not use that as additional information? We *could* do this, but this only works for the Gaussian case, where we already have access to the full solution! And knowing these terms at all times is also roughly equivalent to knowing the full trajectory. So in summary, viewing reverse diffusion trajectories as 2D ‘rotations’ is a useful geometric picture, but it is less quantitatively useful than the full analytical solution to the Gaussian model, e.g. for accelerating sampling.

I.4 Rotation-like dynamics beyond the Gaussian model

There is an alternative source of evidence that reverse diffusion dynamics are rotation-like, even in a more general non-Gaussian setting. Using the form of the projected outcome (Eq. 52), we can write the probability flow ODE (Eq. 4) as

$$\dot{\mathbf{x}} = -\beta(t)\mathbf{x} - \frac{1}{2}g^2(t)\nabla_{\mathbf{x}} \log p(\mathbf{x}, t) = -\beta(t)\mathbf{x} - \frac{1}{2}\frac{g^2(t)}{\sigma_t^2} [\alpha_t \hat{\mathbf{x}}_0(\mathbf{x}) - \mathbf{x}]. \quad (75)$$

Notice that

$$\alpha(t) = \exp\left(-\int_0^t \beta(\tau)d\tau\right) \quad \frac{d}{dt}\alpha(t) = -\beta(t)\alpha(t), \quad (76)$$

which allows us to write

$$\frac{d}{dt}\left(\frac{\mathbf{x}_t}{\alpha_t}\right) = \frac{\dot{\mathbf{x}}_t}{\alpha_t} - \frac{\dot{\alpha}_t \mathbf{x}_t}{\alpha_t^2} = \frac{\dot{\mathbf{x}}_t}{\alpha_t} + \frac{\beta_t \mathbf{x}_t}{\alpha_t}. \quad (77)$$

Equivalently,

$$\frac{d}{dt}\left(\frac{\mathbf{x}_t}{\alpha_t}\right) = -\frac{1}{2}\frac{g^2(t)}{\sigma_t^2} \left(\hat{\mathbf{x}}_0(\mathbf{x}_t) - \frac{\mathbf{x}_t}{\alpha_t}\right) = -\frac{\beta_t}{\sigma_t^2} \left(\hat{\mathbf{x}}_0(\mathbf{x}_t) - \frac{\mathbf{x}_t}{\alpha_t}\right). \quad (78)$$

From this, we can see that the quantity \mathbf{x}_t/α_t , i.e. the state scaled by the signal scale, is isotropically attracted towards the moving target $\hat{\mathbf{x}}_0(\mathbf{x}_t)$ at a rate determined by $\frac{1}{2}\frac{g^2(t)}{\sigma_t^2}$.

Suppose that the endpoint estimates $\hat{\mathbf{x}}_0$ change slowly compared to the state \mathbf{x}_t ; we will show that this gives us rotation-like dynamics.

First, note that we can evaluate the integral

$$\int_T^t \frac{\beta_t}{\sigma_t^2} dt = \int_T^t \frac{\beta_t}{1 - \alpha_t^2} dt \quad (79)$$

using a change of variables $\alpha := \alpha_t$ with $d\alpha = -\beta\alpha dt$. The integral becomes

$$-\int_{\alpha_T}^{\alpha_t} \frac{d\alpha}{\alpha(1 - \alpha^2)} = \log \frac{\sqrt{1 - \alpha^2}}{\alpha} \Big|_{\alpha_T}^{\alpha_t}. \quad (80)$$

Using this integral, we can find that the solution to Eq. 78 under the assumption that $\hat{\mathbf{x}}_0$ remains constant is

$$\begin{aligned} \log \left(\frac{\frac{\mathbf{x}_t}{\alpha_t} - \hat{\mathbf{x}}_0}{\frac{\mathbf{x}_T}{\alpha_T} - \hat{\mathbf{x}}_0} \right) &= \log \left(\frac{\sqrt{1 - \alpha_t^2}}{\alpha_t} \frac{\alpha_T}{\sqrt{1 - \alpha_T^2}} \right) \\ \implies \frac{\mathbf{x}_t - \alpha_t \hat{\mathbf{x}}_0}{\sqrt{1 - \alpha_t^2}} &= \frac{\mathbf{x}_T - \alpha_T \hat{\mathbf{x}}_0}{\sqrt{1 - \alpha_T^2}}. \end{aligned} \quad (81)$$

Interestingly, this indicates that there is a conserved quantity

$$\frac{\mathbf{x}_t - \alpha_t \hat{\mathbf{x}}_0}{\sqrt{1 - \alpha_t^2}} = \text{const.} \quad (82)$$

along the reverse diffusion trajectory, under this approximation. Since $\alpha_T \approx 0$, $\mathbf{x}_T \approx \text{const.}$, i.e. the value of the constant roughly matches the initial noise seed \mathbf{x}_T . Given any $\hat{\mathbf{x}}_0$ the solution to the ODE at time t can be written as

$$\mathbf{x}_t = \alpha_t \hat{\mathbf{x}}_0 + \sqrt{1 - \alpha_t^2} \text{ const.} \approx \alpha_t \hat{\mathbf{x}}_0 + \sqrt{1 - \alpha_t^2} \mathbf{x}_T. \quad (83)$$

In words: through a rotation, \mathbf{x}_t interpolates between $\hat{\mathbf{x}}_0$ and const. This solution paints the picture that the state is constantly rotating towards the estimated outcome $\hat{\mathbf{x}}_0$, with Eq. 83 describing that hypothetical trajectory's shape. But as the target $\hat{\mathbf{x}}_0$ is moving, the actual trajectory will be similar to Eq. 83 only on short time scales, and not on longer time scales. This idea is visualized by the circular dashed curves in Fig. 2.

Beyond the constant $\hat{\mathbf{x}}_0$ approximation, there will be some correction terms to the above rotation formula, whose precise form depends on the (generally not Gaussian) score function.

J Derivation of properties of non-Gaussian score models

In this appendix, we will derive some properties of non-Gaussian score models. In particular, we will derive the score function and endpoint estimate for a Gaussian mixture model, and show that both have a simple form and intuitive interpretation.

J.1 Score function of general Gaussian mixture

Let

$$q(\mathbf{x}) = \sum_i \pi_i \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \quad (84)$$

be a Gaussian mixture distribution, where the π_i are mixture weights, $\boldsymbol{\mu}_i$ is the i -th mean, and $\boldsymbol{\Sigma}_i$ is the i -th covariance matrix. The score function for this distribution is

$$\begin{aligned} \nabla_{\mathbf{x}} \log q(\mathbf{x}) &= \frac{\sum_i \pi_i \nabla_{\mathbf{x}} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{q(\mathbf{x})} \\ &= \sum_i -\boldsymbol{\Sigma}_i^{-1}(\mathbf{x} - \boldsymbol{\mu}_i) \frac{\pi_i \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{q(\mathbf{x})} \\ &= \sum_i \frac{\pi_i \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{q(\mathbf{x})} \nabla_{\mathbf{x}} \log \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \\ &= \sum_i w_i(\mathbf{x}) \nabla_{\mathbf{x}} \log \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \end{aligned} \quad (85)$$

where we have defined the mixing weights

$$w_i(\mathbf{x}) := \frac{\pi_i \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{q(\mathbf{x})}. \quad (86)$$

Thus, the score of the Gaussian mixture is a weighted mixture of the score fields of each of the individual Gaussians.

In the context of diffusion, we are interested in the *time-dependent* score function. Given a Gaussian mixture initial condition, the end result of the VP-SDE forward process will also be a Gaussian mixture:

$$p_t(\mathbf{x}) = \sum_i \pi_i \mathcal{N}(\mathbf{x}; \alpha_t \boldsymbol{\mu}_i, \sigma_t^2 \mathbf{I} + \alpha_t^2 \boldsymbol{\Sigma}_i). \quad (87)$$

The corresponding time-dependent score is

$$\begin{aligned} \mathbf{s}(\mathbf{x}, t) &= \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) \\ &= \sum_i -(\sigma_t^2 \mathbf{I} + \alpha_t^2 \boldsymbol{\Sigma}_i)^{-1}(\mathbf{x} - \alpha_t \boldsymbol{\mu}_i) \frac{\pi_i \mathcal{N}(\mathbf{x}; \alpha_t \boldsymbol{\mu}_i, \sigma_t^2 \mathbf{I} + \alpha_t^2 \boldsymbol{\Sigma}_i)}{p_t(\mathbf{x})} \\ &= \sum_i -(\sigma_t^2 \mathbf{I} + \alpha_t^2 \boldsymbol{\Sigma}_i)^{-1}(\mathbf{x} - \alpha_t \boldsymbol{\mu}_i) w_i(\mathbf{x}, t). \end{aligned} \quad (88)$$

Note that we have a formula for $(\sigma_t^2 \mathbf{I} + \alpha_t^2 \boldsymbol{\Sigma}_i)^{-1}$ (derived using the Woodbury matrix inversion identity; see Eq.58) in terms of the (compact) SVD of $\boldsymbol{\Sigma}_i$. We can use it to write

$$\mathbf{s}(\mathbf{x}, t) = \frac{1}{\sigma_t^2} \sum_i -(\mathbf{I} - \mathbf{U}_i \tilde{\boldsymbol{\Lambda}}_{it} \mathbf{U}_i^T)(\mathbf{x} - \alpha_t \boldsymbol{\mu}_i) w_i(\mathbf{x}, t) \quad (89)$$

where

$$\boldsymbol{\Sigma}_i = \mathbf{U}_i \boldsymbol{\Lambda}_i \mathbf{U}_i^T \quad \tilde{\boldsymbol{\Lambda}}_t := \text{diag} \left[\frac{\alpha_t^2 \lambda_k}{\alpha_t^2 \lambda_k + \sigma_t^2} \right]. \quad (90)$$

This representation of the score function is numerically convenient, since (once the SVDs of each covariance matrix have been obtained), it can be evaluated using a relatively small number of matrix multiplications, which are cheaper than the covariance matrix inversions that a naive implementation of the Gaussian mixture score function would require.

We used this formula (Eq.89) and an off-the-shelf ODE solver to simulate the reverse diffusion trajectory of a 10-mode Gaussian mixture score model (Fig. 4).

J.2 Score function of Gaussian mixture with identical and isotropic covariance

Assume each Gaussian mode has covariance $\Sigma_i = \sigma^2 \mathbf{I}$ and that every mixture weight is the same (i.e. $\pi_i = \pi_j, \forall i, j$). Then the score for this kind of specific Gaussian mixture is

$$\begin{aligned}\nabla_{\mathbf{x}} \log q(\mathbf{x}) &= \frac{\sum_i \pi_i \nabla_{\mathbf{x}} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i, \sigma^2 \mathbf{I})}{\sum_i \pi_i \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i, \sigma^2 \mathbf{I})} \\ &= \frac{\sum_i -\frac{1}{\sigma^2}(\mathbf{x} - \boldsymbol{\mu}_i) \exp\left(-\frac{1}{2\sigma^2} \|\mathbf{x} - \boldsymbol{\mu}_i\|_2^2\right)}{\sum_i \exp\left(-\frac{1}{2\sigma^2} \|\mathbf{x} - \boldsymbol{\mu}_i\|_2^2\right)} \\ &= \frac{1}{\sigma^2} \sum_i w_i(\mathbf{x})(\boldsymbol{\mu}_i - \mathbf{x}),\end{aligned}\quad (91)$$

where the weight $w_i(\mathbf{x})$ is a softmax of the negative squared distance to all the means, with σ^2 functioning as a temperature parameter:

$$w_i(\mathbf{x}) = \text{Softmax}\left(\left\{-\frac{1}{2\sigma^2} \|\mathbf{x} - \boldsymbol{\mu}_i\|_2^2\right\}\right) = \frac{\exp\left(-\frac{1}{2\sigma^2} \|\mathbf{x} - \boldsymbol{\mu}_i\|_2^2\right)}{\sum_i \exp\left(-\frac{1}{2\sigma^2} \|\mathbf{x} - \boldsymbol{\mu}_i\|_2^2\right)}. \quad (92)$$

Since the weights $w_i(\mathbf{x})$ sum to 1, we can also write the score function in the suggestive form

$$\nabla_{\mathbf{x}} \log q(\mathbf{x}) = \frac{(\sum_i w_i(\mathbf{x}) \boldsymbol{\mu}_i) - \mathbf{x}}{\sigma^2}. \quad (93)$$

This has a form analogous to the score of a single Gaussian mode—but instead of \mathbf{x} being ‘attracted’ towards a single mean $\boldsymbol{\mu}$, it is attracted towards a weighted combination of all of the means, with modes closer to the state \mathbf{x} being more highly weighted.

The time-dependent score function of this model is

$$\nabla_{\mathbf{x}} \log q(\mathbf{x}) = \frac{(\sum_i w_i(\mathbf{x}, t) \alpha_t \boldsymbol{\mu}_i) - \mathbf{x}}{\sigma_t^2 + \alpha_t^2 \sigma^2} \quad (94)$$

where

$$w_i(\mathbf{x}, t) = \text{Softmax}\left(\left\{-\frac{1}{2(\sigma_t^2 + \alpha_t^2 \sigma^2)} \|\mathbf{x} - \alpha_t \boldsymbol{\mu}_i\|_2^2\right\}\right). \quad (95)$$

J.3 Endpoint estimate of Gaussian mixture with identical and isotropic covariance

The endpoint estimate of the Gaussian mixture whose modes have identical isotropic covariances is

$$\hat{\mathbf{x}}_0(\mathbf{x}_t) = \frac{\mathbf{x}_t + \sigma_t^2 \nabla \log p(\mathbf{x}_t)}{\alpha_t} = \frac{1}{\alpha_t} \left[\frac{\alpha_t^2 \sigma^2}{\sigma_t^2 + \alpha_t^2 \sigma^2} \mathbf{x}_t + \frac{1}{\sigma_t^2 + \alpha_t^2 \sigma^2} \sum_i w_i(\mathbf{x}_t, t) \alpha_t \boldsymbol{\mu}_i \right]. \quad (96)$$

J.4 Score and endpoint estimate of exact (delta mixture) score model

A particularly interesting special case of the Gaussian mixture model is the delta mixture model used in the main text, whose components are vanishing width Gaussians centered on the training images. In particular, consider a data set $\{\mathbf{y}_i\}$ with $i = 1, \dots, N$, so that the starting distribution is

$$p(\mathbf{x}) = \frac{1}{N} \sum_i \delta(\mathbf{x} - \mathbf{y}_i). \quad (97)$$

At time t , the marginal distribution will be a Gaussian mixture

$$p_t(\mathbf{x}_t) = \frac{1}{N} \sum_i \mathcal{N}(\mathbf{x}_t; \alpha_t \mathbf{y}_i, \sigma_t^2 \mathbf{I}). \quad (98)$$

Then using the Eq. 91 above we have

$$\begin{aligned}
s(\mathbf{x}_t, t) &= \nabla \log p_t(\mathbf{x}_t) = \frac{1}{\sigma_t^2} \sum_i w_i(\mathbf{x}_t)(\alpha_t \mathbf{y}_i - \mathbf{x}_t) \\
&= \frac{1}{\sigma_t^2} \left[-\mathbf{x}_t + \alpha_t \sum_i w_i(\mathbf{x}_t) \mathbf{y}_i \right] \\
&= \frac{1}{\sigma_t^2} \left[-\mathbf{x}_t + \alpha_t \sum_i \text{Softmax}\left(\left\{-\frac{1}{2\sigma_t^2} \|\alpha_t \mathbf{y}_i - \mathbf{x}_t\|^2\right\}\right) \mathbf{y}_i \right].
\end{aligned} \tag{99}$$

The endpoint estimate of the distribution is

$$\begin{aligned}
\hat{\mathbf{x}}_0(\mathbf{x}_t) &= \frac{\mathbf{x}_t + \sigma_t^2 \nabla \log p(\mathbf{x}_t)}{\alpha_t} \\
&= \sum_i \text{Softmax}\left(\left\{-\frac{1}{2\sigma_t^2} \|\alpha_t \mathbf{y}_i - \mathbf{x}_t\|^2\right\}\right) \mathbf{y}_i \\
&= \sum_i w_i(\mathbf{x}_t) \mathbf{y}_i.
\end{aligned} \tag{100}$$

Thus, the endpoint estimate is a weighted average of training data, with the softmax of negative squared distance as weights and σ_t^2 as a temperature parameter.