

SiT: Exploring Flow and Diffusion-based Generative Models with Scalable Interpolant Transformers

Nanye Ma, Mark Goldstein, Michael S. Albergo, Nicholas M. Boffi, Eric Vanden-Eijnden[†], and Saining Xie[†]

New York University

Abstract. We present Scalable Interpolant Transformers (SiT), a family of generative models built on the backbone of Diffusion Transformers (DiT). The interpolant framework, which allows for connecting two distributions in a more flexible way than standard diffusion models, makes possible a modular study of various design choices impacting generative models built on dynamical transport: learning in discrete or continuous time, the objective function, the interpolant that connects the distributions, and deterministic or stochastic sampling. By carefully introducing the above ingredients, SiT surpasses DiT uniformly across model sizes on the conditional ImageNet 256×256 and 512×512 benchmark using the exact same model structure, number of parameters, and GFLOPs. By exploring various diffusion coefficients, which can be tuned separately from learning, SiT achieves an FID-50K score of 2.06 and 2.62, respectively. Code is available here: <https://github.com/willisma/SiT>

1 Introduction

Contemporary success in image generation has come from a combination of algorithmic advances, improvements in model architecture, and progress in scaling neural network models and data. State-of-the-art diffusion models [25, 53] proceed by incrementally transforming data into Gaussian noise as prescribed by an iterative stochastic process, which can be specified either in discrete or continuous time. At an abstract level, this corruption process can be viewed as defining a time-dependent distribution that is iteratively smoothed from the original data distribution into a standard normal distribution. Diffusion models learn to reverse this corruption process and push Gaussian noise backwards along this connection to obtain data samples. The objects learned to perform this transformation conventionally predict either the noise in the corruption process [25] or the score of the distribution that connects the data and the Gaussian [64], though alternatives of these choices exist [28, 56]. While diffusion models originally represented these objects with a U-Net architecture [25, 54], recent work has highlighted that architectural advances in vision such as the

[†] Equal advising.

Table 1: Scalable Interpolant Transformers. We systematically vary the following aspects of a generative model: **time discretization**, **model prediction**, **interpolant**, and **sampler**. The resulting Scalable Interpolant Transformer (SiT) model, under identical training compute, consistently outperforms the Diffusion Transformer (DiT) in generating 256×256 ImageNet images. All models employ a patch size of 2. In this work, we ask the question: *What is the source of the performance gain?*

Model	Params(M)	Training Steps	FID ↓
DiT-S	33	400K	68.4
SiT-S	33	400K	57.6
DiT-B	130	400K	43.5
SiT-B	130	400K	33.0
DiT-L	458	400K	23.3
SiT-L	458	400K	18.8
DiT-XL	675	400K	19.5
SiT-XL	675	400K	17.2
DiT-XL	675	7M	9.6
SiT-XL	675	7M	8.3
DiT-XL (cfg=1.5)	675	7M	2.27
SiT-XL (cfg=1.5)	675	7M	2.06

Vision Transformer (ViT) [21] can be incorporated into the standard diffusion model pipeline to improve performance [50].

Orthogonally, significant research effort has gone into exploring the structure of the noising process, which has been shown to lead to performance benefits [33, 36, 37, 60]. Yet, many of these efforts do not move past the notion of passing data through a diffusion process with an equilibrium distribution, which is a restricted type of connection between the data and the Gaussian. Recently-introduced *stochastic interpolants* [2] lift this constraint and introduce more flexibility in the noise-data connection. In this paper, we systematically explore the effect of this flexibility on performance in large scale image generation.

Intuitively, we expect that the difficulty of the *learning problem* can be related to both the specific connection chosen and the object that is learned. Our aim is to clarify these design choices, so as to simplify the learning problem and thereby improve performance. To understand where potential benefits arise in the learning problem, we start with Denoising Diffusion Probabilistic Models (DDPMs) and sweep through adaptations of: (i) which object to learn, and (ii) which interpolant to choose to reveal best practices.

In addition to the learning problem, there is a *sampling problem* that must be solved at inference time. It has been acknowledged for diffusion models that sampling can be either deterministic or stochastic [63], and the choice of sampling method can be made after the learning process. Yet, the diffusion coefficients used for stochastic sampling are typically presented as intrinsically tied to the forward noising process, which need not be the case in general.



Fig. 1: Selected samples from SiT-XL models trained on ImageNet [55] at 512×512 and 256×256 resolution with $\text{cfg} = 4.0$, respectively.

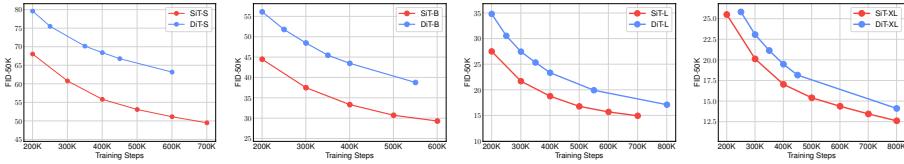


Fig. 2: SiT improves FID across all model sizes. FID-50K over training iterations for both DiT and SiT. All results are produced by a Euler-Maruyama sampler using 250 integration steps. Across all model sizes, SiT converges much faster.

Throughout this paper, we explore how the design of the interpolant and the use of the resulting model as either a deterministic or a stochastic sampler impact performance. We gradually transition from a typical denoising diffusion model to an interpolant model by taking a series of orthogonal steps in the design space. As we progress, we carefully evaluate how each move away from the diffusion model impacts the performance. In summary, our **main contributions** are:

- We systematically study the SiT design space through the combinations of the four key components: **time discretization**, **model prediction**, **interpolant**, and **sampler**.
- We provide theoretical motivation for the choice of each component and study how they lead to improved practical performance.
- We exploit the tunability of the diffusion coefficient of the stochastic sampler, and show that its adaptation can tighten control of the KL-divergence between the model and the target. We show how this leads to empirical benefits without any additional re-training.

- Combining the best design choices identified in each component, our SiT model surpasses Diffusion Transformer(DiT) on both 256×256 and 512×512 image resolution, achieving FID-50K scores of 2.06 and 2.62, respectively, without modifying any structure or hyperparameter of the model.

2 SiT: Scalable Interpolant Transformers

We begin by recalling the main ingredients for building flow-based and diffusion-based generative models.

2.1 Flows and diffusions

Flow and diffusion models both utilize stochastic processes to gradually turn noise $\varepsilon \sim \mathcal{N}(0, \mathbf{I})$ into data $\mathbf{x}_* \sim p(\mathbf{x})$ for the generating task. Such time-dependent processes can be summarized as follow

$$\mathbf{x}_t = \alpha_t \mathbf{x}_* + \sigma_t \varepsilon, \quad (1)$$

where α_t is a decreasing function of t and σ_t is an increasing function of t . Stochastic interpolants and other flow matching methods [2, 4, 41, 43] restrict the process (1) on $t \in [0, 1]$, and set $\alpha_0 = \sigma_1 = 1$, $\alpha_1 = \sigma_0 = 0$, so that \mathbf{x}_t interpolates exactly between \mathbf{x}_* at time $t = 0$ and ε at time $t = 1$. By contrast, score-based diffusion models [33, 37, 64] set both α_t and σ_t indirectly through a forward-time stochastic differential equation (SDE) with $\mathcal{N}(0, \mathbf{I})$ as its equilibrium distribution, i.e. \mathbf{x}_t converges to $\mathcal{N}(0, \mathbf{I})$ only if $t \rightarrow \infty$.

Despite the nuances in formulating the stochastic processes \mathbf{x}_t , common to both stochastic interpolants and score-based diffusion models is the observation that \mathbf{x}_t can be sampled dynamically using either a reverse-time SDE or a probability flow ordinary differential equation (ODE).

Probability flow ODE. The marginal probability distribution $p_t(\mathbf{x})$ of \mathbf{x}_t in (1) coincides with the distribution of the probability flow ODE with a velocity field

$$\dot{\mathbf{X}}_t = \mathbf{v}(\mathbf{X}_t, t), \quad (2)$$

where $\mathbf{v}(\mathbf{x}, t)$ is given by the conditional expectation

$$\begin{aligned} \mathbf{v}(\mathbf{x}, t) &= \mathbb{E}[\dot{\mathbf{x}}_t | \mathbf{x}_t = \mathbf{x}], \\ &= \dot{\alpha}_t \mathbb{E}[\mathbf{x}_* | \mathbf{x}_t = \mathbf{x}] + \dot{\sigma}_t \mathbb{E}[\varepsilon | \mathbf{x}_t = \mathbf{x}]. \end{aligned} \quad (3)$$

The correspondence between $p_t(\mathbf{x})$ and (2) and the formulation of (3) is derived in Appendix A.1. By solving (2) backwards in time from $\mathbf{X}_T = \varepsilon \sim \mathcal{N}(0, \mathbf{I})$, we can generate samples from $p_0(\mathbf{x})$, which approximates the ground-truth data distribution $p(\mathbf{x})$. We refer to (2) as a *flow-based* generative model.

Reverse-time SDE. The time-dependent probability distribution $p_t(\mathbf{x})$ of \mathbf{x}_t also coincides with the distribution of the reverse-time SDE [5]

$$d\mathbf{X}_t = \mathbf{v}(\mathbf{X}_t, t)dt - \frac{1}{2}w_t \mathbf{s}(\mathbf{X}_t, t)dt + \sqrt{w_t} d\bar{\mathbf{W}}_t, \quad (4)$$

where $\bar{\mathbf{W}}_t$ is a reverse-time Wiener process, $w_t > 0$ is an arbitrary time-dependent diffusion coefficient, $\mathbf{v}(\mathbf{x}, t)$ is the velocity defined in (3), and $\mathbf{s}(\mathbf{x}, t) = \nabla \log p_t(\mathbf{x})$ is the score. Similar to \mathbf{v} , this score is given by the conditional expectation

$$\mathbf{s}(\mathbf{x}, t) = -\sigma_t^{-1} \mathbb{E}[\varepsilon | \mathbf{x}_t = \mathbf{x}]. \quad (5)$$

Again, the correspondence between $p_t(\mathbf{x})$ and (4) and the formulation of (5) is derived in Appendix A.3. Solving the reverse SDE (4) backwards in time from $\mathbf{X}_T = \varepsilon \sim N(0, \mathbf{I})$ enables generating samples from the approximated data distribution $p_0(\mathbf{x}) \sim p(\mathbf{x})$. We refer to (4) as a *stochastic* generative model.

Design choices. Score-based diffusion models typically tie the choice of α_t , σ_t , and w_t in (4) to the drift and diffusion coefficients used in the forward SDE that generates \mathbf{x}_t (see (10) below). The stochastic interpolant framework decouples the formulation of \mathbf{x}_t from the forward SDE and shows that there is more flexibility in the choices of α_t , σ_t , and w_t . Below, we will exploit this flexibility to construct generative models that outperform score-based diffusion models on standard benchmarks in image generation task.

2.2 Estimating the score and the velocity

Practical use of the probability flow ODE (2) and the reverse-time SDE (4) as generative models relies on our ability to estimate the velocity $\mathbf{v}(\mathbf{x}, t)$ and/or score $\mathbf{s}(\mathbf{x}, t)$ fields that enter these equations. The key observation made in score-based diffusion models is that the score can be estimated parametrically as $\mathbf{s}_\theta(\mathbf{x}, t)$ using the loss

$$\mathcal{L}_s(\theta) = \int_0^T \mathbb{E}[\|\sigma_t \mathbf{s}_\theta(\mathbf{x}_t, t) + \varepsilon\|^2] dt. \quad (6)$$

This loss can be derived by using (5) along with standard properties of the conditional expectation. Similarly, the velocity in (3) can be estimated parametrically as $\mathbf{v}_\theta(\mathbf{x}, t)$ via the loss

$$\mathcal{L}_v(\theta) = \int_0^T \mathbb{E}[\|\mathbf{v}_\theta(\mathbf{x}_t, t) - \dot{\alpha}_t \mathbf{x}_* - \dot{\sigma}_t \varepsilon\|^2] dt. \quad (7)$$

We note that any time-dependent weight can be included under the integrals in both (6) and (7). These weight factors are key in the context of score-based models when T becomes large [36]; in contrast, with stochastic interpolants where $T = 1$ without any bias, these weights are less important and might impose numerical stability issue (see Appendix B).

Model prediction. We observed that only one of $\mathbf{s}_\theta(\mathbf{x}, t)$ and $\mathbf{v}_\theta(\mathbf{x}, t)$ is needed to be estimated in practice. This follows directly from the constraint

$$\begin{aligned}\mathbf{x} &= \mathbb{E}[\mathbf{x}_t | \mathbf{x}_t = \mathbf{x}], \\ &= \alpha_t \mathbb{E}[\mathbf{x}_* | \mathbf{x}_t = \mathbf{x}] + \sigma_t \mathbb{E}[\boldsymbol{\varepsilon} | \mathbf{x}_t = \mathbf{x}],\end{aligned}\tag{8}$$

which can be used to re-express the score (5) in terms of the velocity (3) as

$$\mathbf{s}(\mathbf{x}, t) = \sigma_t^{-1} \frac{\alpha_t \mathbf{v}(\mathbf{x}, t) - \dot{\alpha}_t \mathbf{x}}{\dot{\alpha}_t \sigma_t - \alpha_t \dot{\sigma}_t}.\tag{9}$$

We include a detailed derivation in Appendix A.4. Notably, given the simply linear relationship posed by (9), we can also express $\mathbf{v}(\mathbf{x}, t)$ in terms of $\mathbf{s}(\mathbf{x}, t)$. We will use this relation to specify our **model prediction**. In our experiments, we typically learn the velocity field $\mathbf{v}(\mathbf{x}, t)$ and use it to express the score $\mathbf{s}(\mathbf{x}, t)$ when using an SDE for sampling.

Note that by our definitions $\dot{\alpha}_t < 0$ and $\dot{\sigma}_t > 0$, so that the denominator of (9) is never zero. Yet, σ_t vanishes at $t = 0$, making the σ_t^{-1} in (9) cause a singularity¹. This suggests the choice $w_t = \sigma_t$ in (4) to cancel this singularity, for which we will explore the performance in the numerical experiments.

Time discretization. The objective functions specified above are defined over a continuous time domain, as opposed to DDPM which couples the time grid used in learning to that used in sampling. Learning in continuous time allows us to specify a discretization used in sampling *a posteriori*, which allows for flexibility in both sampling efficiency and performance.

2.3 Specifying the interpolating process

In Sec. 2.1 we present the general definition of interpolants (α_t and σ_t) for both stochastic interpolant and score-based diffusion. In this section we dive into more details and specify the three choices of interpolants to explore in the experiments.

Score-based diffusion. We follow [64] and use the standard variance-preserving (VP) SDE in forward-time

$$d\mathbf{X}_t = -\frac{1}{2} \beta_t \mathbf{X}_t dt + \sqrt{\beta_t} d\mathbf{W}_t\tag{10}$$

for some $\beta_t > 0$, \mathbf{x}_t 's perturbation kernel $p_t(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\alpha_t \mathbf{x}_t, \sigma_t^2 \mathbf{I})$ is defined by

$$\text{SBDM-VP: } \alpha_t = e^{-\frac{1}{2} \int_0^t \beta_s ds}, \quad \sigma_t = \sqrt{1 - e^{-\int_0^t \beta_s ds}}.\tag{11}$$

¹ We remark that $\mathbf{s}(\mathbf{x}, t)$ can be shown to be non-singular at $t = 0$ analytically if the data distribution $p(\mathbf{x})$ has a smooth density [2], though this singularity appears in numerical implementations and losses in general.

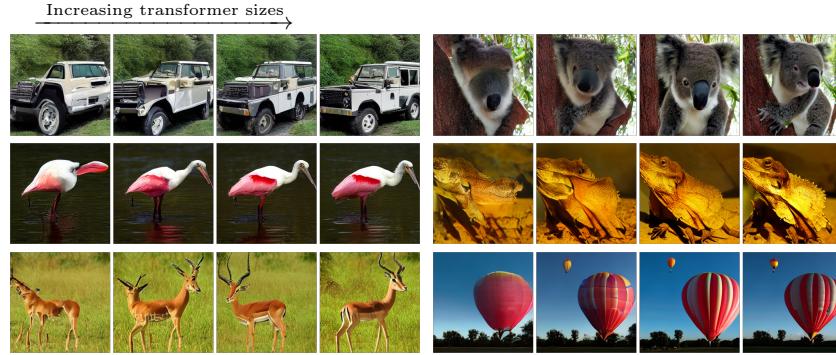


Fig. 3: Increasing transformer size increases sample quality. Best viewed zoomed-in. We sample from all 4 of our SiT model (SiT-S, SiT-B, SiT-L and SiT-XL) after 400K training steps using the same latent noise and class label.

The only design flexibility in (11) comes from the choice of β_t , as it determines both α_t and σ_t ². For example, setting $\beta_t = 1$ leads to $\alpha_t = e^{-t}$ and $\sigma_t = \sqrt{1 - e^{-2t}}$. This choice necessitates taking T sufficiently large [25] or searching for more appropriate choices of β_t [16, 60, 64] to reduce the bias. To be specific, such bias comes from the mismatch between the condition $\varepsilon \sim N(0, \mathbf{I})$ used in practice for sampling and the density of the process $\mathbf{x}_1 \not\sim N(0, \mathbf{I})$, as stated in Sec. 2.1.

General interpolants. In the stochastic interpolant framework, the process (1) is defined explicitly and without any reference to a forward SDE, creating more flexibility in the choice of α_t and σ_t . Specifically, any choice satisfying:

- (i) $\alpha_t^2 + \sigma_t^2 > 0$ for all $t \in [0, 1]$;
- (ii) α_t and σ_t are differentiable for all $t \in [0, 1]$;
- (iii) $\alpha_1 = \sigma_0 = 0$, $\alpha_0 = \sigma_1 = 1$;

gives a process that interpolates without bias between $\mathbf{x}_{t=0} = \mathbf{x}_*$ and $\mathbf{x}_{t=1} = \varepsilon$. In our numerical experiments, we exploit this design flexibility to test, in particular, the choices

$$\begin{aligned} \text{Linear: } \quad \alpha_t &= 1 - t, & \sigma_t &= t, \\ \text{GVP: } \quad \alpha_t &= \cos(\frac{1}{2}\pi t), & \sigma_t &= \sin(\frac{1}{2}\pi t), \end{aligned} \tag{12}$$

where GVP refers to a generalized VP which has constant variance across time for any endpoint distributions with the same variance. We note that the fields $\mathbf{v}(\mathbf{x}, t)$ and $\mathbf{s}(\mathbf{x}, t)$ entering (2) and (4) depend on the choice of α_t and σ_t , and typically must be specified before learning³. This is in contrast to the diffusion coefficient $w(t)$, as we now describe.

² VP is the only linear scalar SDE with an equilibrium distribution [60]; interpolants extend beyond $\alpha_t^2 + \sigma_t^2 = 1$ by foregoing the requirement of an equilibrium distribution.

³ The requirement to learn and sample under one choice of path specified by α_t, σ_t , at training time may be relaxed and is explored in [1].

2.4 Specifying the diffusion coefficient

As stated earlier, the SBDM diffusion coefficient used in (4) is usually taken to match that of (10). That is, one sets $w_t = \beta_t$. In the stochastic interpolant framework, this choice is again subject to greater flexibility: *any* $w_t \geq 0$ can be used. Interestingly, this choice can be made *after* learning, as it does not affect the velocity $\mathbf{v}(\mathbf{x}, t)$ or the score $\mathbf{s}(\mathbf{x}, t)$. In our experiments, we exploit this flexibility by considering the following choices:

- (i) $w_t = \sigma_t$; this is used to eliminate the singularity at $t = 0$ following the explanation at the end of Sec. 2.2;
- (ii) $w_t = \sin^2(\pi t)$; this also eliminates the singularity at $t = 0$, and allows us to explore the effect of removing diffusivity at times close to $t = 1$ in sampling.
- (iii) w_t can be chosen to minimize an upper bound on the KL divergence $D_{\text{KL}}(p(\mathbf{x}) \| p_0(\mathbf{x}))$, where $p(\mathbf{x})$ denotes the true data distribution and $p_0(\mathbf{x})$ refers to the density of \mathbf{x}_t at $t = 0$. Disregarding the simulation cost of integrating the SDE (4), it can be shown (see Appendix A.5) that the following choice of w_t minimizes the KL upper bound:

$$w_t = w_t^{\text{KL}} \equiv 2 \left(\dot{\sigma}_t \sigma_t - \frac{\dot{\alpha}_t \sigma_t^2}{\alpha_t} \right). \quad (13)$$

Under the SBDM-VP interpolant, w_t^{KL} coincides with β_t ; this aligns with the claim made in [63].

- (iv) If the SDE in (iii) becomes hard to integrate because of the magnitude of w_t^{KL} near $t = 1$, one may wish to *regularize* the diffusion coefficient to reduce the integration cost. For example, difficulties may arise for the Linear and GVP interpolants, because $w_t^{\text{KL}} \rightarrow \infty$ as $t \rightarrow 1$ given the presence of α_t in the denominator of (13). Including the integration cost of (4), it can also be shown (see Appendix A.5) that an optimal regularized w_t is given by

$$w_t^{\text{KL}, \eta} \equiv w_t^{\text{KL}} \sqrt{\frac{\mathcal{L}_t}{\mathcal{L}_t + 2\eta(w_t^{\text{KL}})^2}}, \quad (14)$$

where \mathcal{L}_t is the value of \mathcal{L}_v in Sec. 2.2 at time t , and η is any non-negative constant. With $\eta = 0$, we recover w_t^{KL} . For score models, we first convert to a velocity model following (9), then calculate the corresponding \mathcal{L}_v . As $t \rightarrow 1$, $w_t^{\text{KL}, \eta}$ approaches a limit at $\sqrt{\frac{\mathcal{L}_{t \rightarrow 1}}{2\eta}}$. If \mathcal{L}_t is defined everywhere on $[0, 1]$, then $w_t^{\text{KL}, \eta}$ will be well-behaved on $[0, 1]$.

2.5 Interpolant Transformer Architecture

The backbone architecture and capacity of generative models are both crucial for producing high-quality samples. In order to eliminate any confounding factors and focus on our exploration, we strictly follow the standard Diffusion Transformer

(DiT) [50] and its configurations. This way, we can also test the scalability of our model across various model sizes.

Here we briefly introduce the model design. Generating high-resolution images with diffusion models can be computationally expensive. Latent diffusion models (LDMs) [53] address this by first downsampling images into a smaller latent embedding space using an encoder E , and then training a diffusion model on $z = E(x)$. New images are created by sampling z from the model and decoding it back to images using a decoder $x = D(z)$.

Similarly, SiT is a latent generative model, and we use the same pre-trained VAE encoder and decoder models originally used in Stable Diffusion [53]. SiT processes a spatial input z (shape $32 \times 32 \times 4$ for $256 \times 256 \times 3$ images) by first ‘patchifying’ it into T linearly embedded tokens of dimension d . We always use a patch size of 2 in these models as they achieve the best sample quality. We then apply standard ViT [21] sinusoidal positional embeddings to these tokens. We use a series of N SiT transformer blocks, each with hidden dimension d .

Our model configurations—SiT- $\{\text{S,B,L,XL}\}$ —vary in model size (parameters) and compute (flops), allowing for a model scaling analysis. For class-conditional generation on ImageNet, we use the AdaLN-Zero block [50] to process additional conditional information (times and class labels). SiT architectural details are listed in Appendix E.

The complete SiT design space that we explore consists of the choice of time discretization and the model prediction (Sec. 2.2), the choice of the interpolant (Sec. 2.3), the choice of sampler and diffusion coefficient (Sec. 2.4), and the model size (Sec. 2.5).

3 Experiments

To provide a more detailed answer to the question raised in Table 1 and make a fair comparison between DiT and SiT, we gradually transition from a DiT model (discrete, score prediction, VP interpolant) to a SiT model (continuous, velocity prediction, Linear interpolant) and present the impacts on performance.

Experimental setup. In the transition experiments, we use SiT-B models trained on 256×256 image resolution on the ImageNet as our backbone. We fix training steps to be 400K throughout the transition. For solving the ODE (2), we adopt a fixed step second-order Heun integrator; for solving the SDE (4), we used a first-order Euler-Maruyama integrator. With both solver choices we limit the number of function evaluations (NFE) to be 250 to match the number of sampling steps used in DiT. All metrics presented are FID-50K scores evaluated on the ImageNet training set unless otherwise stated.

We also scale up our SiT model to the XL configuration and train on both 256×256 and 512×512 resolution on ImageNet. *We strictly follow the training settings of DiT and did not tune any hyperparameters.*

3.1 Model Parameterization

Discrete- to continuous-time. Continuous time training has been previously studied from the perspective of improved likelihood bounds [37, 64]. As mentioned in Section 2.2, here we focus on the fact that training in continuous time allows us to decouple discretization choices in sampling from the particular training method, which allows for finding the right discretization for various choices of diffusion coefficients that we are free to choose after training. We observe a marginal performance increase in Table 2 by switching to continuous time.

We additionally observe in Figure 5 that flexibility in integration allows one to trade-off number of functional evaluations and FID performance.

Model parameterization. To clarify the role of the model parameterization in the context of SBDM-VP, we now compare learning (i) a score model using (6) (\mathcal{L}_s), (ii) a weighted score model (\mathcal{L}_{s_λ}), or (iii) a velocity model using (7) (\mathcal{L}_v). We observe a significant performance increase with \mathcal{L}_{s_λ} and \mathcal{L}_v in Table 3.

In accordance with the observation made in [36], we carefully choose a $\lambda(t)$ such that λ_{s_λ} is made equivalent to λ_v . We will provide detailed derivations in Appendix A.3, and demonstrate such λ is closely related to the maximum likelihood weighting proposed in [63, 66]. Furthermore, we note that $\lambda(t) \rightarrow \infty$ as $t \rightarrow 0$, thus compensating for the vanishing gradient of the score objective when near the data. This could also account for the performance gain from \mathcal{L}_s to \mathcal{L}_{s_λ} .

Table 2: Discrete vs. continuous.

	Model	Objective	FID
DDPM	Noise	\mathcal{L}_s^N	44.2
SBDM-VP	Score	\mathcal{L}_s	43.6

Table 3: Effect of parameterizations.

Interpolant	Model	Objective	FID
SBDM-VP	Score	\mathcal{L}_s	43.6
SBDM-VP	Score	\mathcal{L}_{s_λ}	39.1
SBDM-VP	Velocity	\mathcal{L}_v	39.8

Choices of interpolant. Sec. 2 highlights that there are many possible ways to build a connection between the data distribution and a Gaussian by varying the choice of α_t and σ_t in the definition of the interpolant (1). To understand the role of this choice, we now study the benefits of moving away from the commonly-used SBDM-VP setup. We consider learning a velocity model $\mathbf{v}(\mathbf{x}, t)$ with the Linear and GVP interpolants presented in (12), which make the interpolation between the Gaussian and the data distribution exact on $[0, 1]$. We benchmark these models against the SBDM-VP in Table 4, where we find that both the GVP and Linear interpolants obtain significantly improved performance.

One possible explanation for this observation is given in Fig. 4, where we see that the path length (transport cost) is reduced when changing from SBDM-VP to GVP or Linear. We note that this is equivalently reducing curvatures in the

ODE trajectories from SBDM-VP to Linear, which is known to reduce the time-discretization errors in sampling [40, 43], and thus contributing to the performance. Numerically, we also note that for SBDM-VP, $\dot{\sigma}_t = \beta_t e^{-\int_0^t \beta_s ds} / (2\sigma_t)$ becomes singular at $t = 0$: this can pose numerical difficulties inside \mathcal{L}_v , leading to difficulty in learning near the data distribution. This issue does not appear with the GVP and Linear interpolants.

Table 4: Effect of interpolant.

Interpolant	Model	Objective	FID
SBDM-VP	Velocity	\mathcal{L}_v	39.8
Linear	Velocity	\mathcal{L}_v	34.8
GVP	Velocity	\mathcal{L}_v	34.6

Table 5: ODE vs. SDE, $w_t = w_t^{\text{KL}}$.

Interpolant	Model	Objective	ODE	SDE
SBDM-VP	Velocity	\mathcal{L}_v	39.8	37.8
Linear	Velocity	\mathcal{L}_v	34.8	33.6
GVP	Velocity	\mathcal{L}_v	34.6	32.9

3.2 Deterministic vs stochastic sampling

As shown in Sec. 2, given a learned model, we can sample using either the probability flow equation (2) or an SDE (4). In Tab. 5 we illustrate the discrepancy between the two methods when using the same trained velocity model. We find performance improvements by sampling with an SDE over the ODE, which is in line with the bounds given in [2]: the SDE has better control over the KL divergence between the model density at $t = 0$ and the ground truth data distribution. We also note that the performance of ODE and SDE integrators may differ under different computation budgets. As shown in Fig. 5, the ODE converges faster with fewer NFE, while the SDE is capable of reaching a much lower final FID score when given a larger computational budget.

Tunable diffusion coefficient. Motivated by the improved performance of SDE sampling, we now consider the effect of tuning the diffusion coefficient in inference. As shown in Table 6, we sweep through all different combinations of our model prediction and interpolant, and present the result. We find that the optimal choice for sampling is both *model prediction* and *interpolant* dependent.

According to Sec. 2.4, the choice of $w_t = w_t^{\text{KL}}$ would ideally minimize the upper bound for the KL divergence $D_{\text{KL}}(p(\mathbf{x}) || p_0(\mathbf{x}))$ and make the SDE approximate the data distribution more closely, barring integration costs. This theoretical result is supported by empirical observation for the SBDM-VP and GVP interpolants presented in Table 6. For Linear interpolants, the cost-regularized version $w_t^{\text{KL}, \eta}$ provides the best FID, because the SDE for the Linear interpolant with w_t^{KL} becomes hard to integrate at the endpoint. Generally speaking, the score models perform worse than the velocity models, which may be due to the singularity of the objective in (6). Moreover, the efficacy of using w_t^{KL} in this context is also reduced, for similar reason. For example, reverting (9) to obtain $\mathbf{v}_\theta(\mathbf{x}, t)$

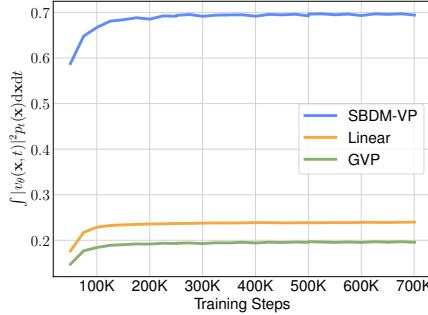


Fig. 4: Path length. The path length $C(v) = \int_0^1 \mathbb{E}[\|\mathbf{v}(\mathbf{x}, t)\|^2] dt$ arising from the velocity field at different training steps; each curve is approximated by 10000 datapoints at each training step.

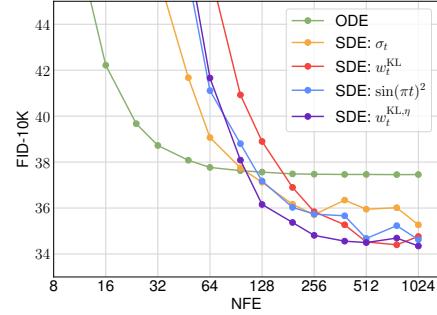


Fig. 5: Comparison of ODE and SDE w/ choices of diffusion coefficients. We evaluate each sampler using a 400K steps trained SiT-B model with Linear interpolant and learning the $\mathbf{v}(\mathbf{x}, t)$.

Table 6: Evaluation of our SDE samplers. The last three columns specify different diffusion coefficients w_t . To make the SBDM-VP competitive, we perform evaluation on the weighted score model \mathcal{L}_{s_λ} . We mark the optimal w_t for each interpolant.

Interpolant	Model	Objective	$w_t = w_t^KL$	$w_t = \sigma_t$	$w_t = \sin^2(\pi t)$	$w_t = w_t^{KL,\eta}$
SBDM-VP	velocity	\mathcal{L}_v	37.8	38.7	39.2	41.1
	score	\mathcal{L}_{s_λ}	35.7	37.1	37.7	38.9
GVP	velocity	\mathcal{L}_v	32.9	33.4	33.6	33.2
	score	\mathcal{L}_s	37.8	33.5	33.2	33.3
Linear	velocity	\mathcal{L}_v	33.6	33.5	33.3	33.0
	score	\mathcal{L}_s	41.0	35.3	34.4	34.9

from $s_\theta(\mathbf{x}, t)$ will result in a singularity at $t = 1$ in \mathcal{L}_t used to choose $w_t^{KL,\eta}$ in (14). Lastly, for SBDM-VP we observe worse result from $w_t^{KL,\eta}$ as opposed to w_t^KL . Different from Linear and GVP, as stated in Sec. 2.4 and Sec. 3.1, w_t^KL is well-defined everywhere on $[0, 1]$ for SBDM-VP, whereas $w_t^{KL,\eta}$ suffers from the singularity issue posed by \mathcal{L}_v near $t = 0$. These observations supports our claim made before, that the optimal choice of w_t will always be *model prediction* and *interpolant* dependent.

We also note that the influences of different diffusion coefficients can vary across different model sizes. Empirically, we observe the best choice for our SiT-XL is a velocity model with Linear interpolant and sampled with $w_t^{KL,\eta}$.

3.3 Classifier-free guidance

Classifier-free guidance (CFG) [27] often leads to improved performance for score-based models. In this section, we give a concise justification for adopting

it on the velocity model, and then empirically show that the drastic gains in performance for DiT case carry across to SiT.

Guidance for a velocity field means that: (i) that the velocity model $\mathbf{v}_\theta(\mathbf{x}, t; \mathbf{y})$ takes class labels y during training, where y is occasionally masked with a null token \emptyset ; and (ii) during sampling the velocity used is $\mathbf{v}_\theta^\zeta(\mathbf{x}, t; \mathbf{y}) = \zeta \mathbf{v}_\theta(\mathbf{x}, t; \mathbf{y}) + (1 - \zeta) \mathbf{v}_\theta(\mathbf{x}, t; \emptyset)$ for a fixed $\zeta > 0$. In Appendix C, we show that this indeed corresponds to sampling the tempered density $p(\mathbf{x}_t)p(\mathbf{y}|\mathbf{x}_t)^\zeta$ as proposed in [48]. Given this observation, one can leverage the usual argument for classifier-free guidance of score-based models.

We observed similar performance improvement with our SiT-XL models under identical computation budget and CFG scale as DiT-X: models. For SiT-XL 256×256 , we follow identical settings in DiT and train the model for 7M steps. We show samples in Fig. 1, and report the result in Table 7. For SiT-XL 512×512 , we train the model for 3M steps under the same setting and report the result in Table 7. Under both training settings we observe performance advantage of SiT. We display more samples in Fig. 1 and in Appendix F.

Table 7: Benchmarking class-conditional image generation on ImageNet 256 × 256 and 512 × 512. SiT-XL surpasses DiT-XL in both resolutions.

Class-Conditional ImageNet 256 × 256						Class-Conditional ImageNet 512 × 512					
Model	FID↓	sFID↓	IS↑	Precision↑	Recall↑	Model	FID↓	sFID↓	IS↑	Precision↑	Recall↑
BigGAN-deep [10]	6.95	7.36	171.4	0.87	0.28	BigGAN-deep [10]	8.43	8.13	177.90	0.88	0.29
StyleGAN-XL [57]	2.30	4.02	265.12	0.78	0.53	StyleGAN-XL [57]	2.41	4.06	267.75	0.77	0.52
Mask-GIT [12]	6.18	-	182.1	-	-	Mask-GIT [12]	7.32	-	156.0	-	-
ADM [19]	10.94	6.02	100.98	0.69	0.63	ADM [19]	23.24	10.19	58.06	0.73	0.60
ADM-G, ADM-U	3.94	6.14	215.84	0.83	0.53	ADM-G, ADM-U	3.85	5.86	221.72	0.84	0.53
CDM [26]	4.88	-	158.71	-	-	Simple Diffusion(U-Net) [28]	4.28	-	171.0	-	-
RIN [31]	3.42	-	182.0	-	-	Simple Diffusion(U-ViT, L)	4.53	-	205.3	-	-
Simple Diffusion(U-Net) [28]	3.76	-	171.6	-	-	VDM++ [36]	2.65	-	278.1	-	-
Simple Diffusion(U-ViT, L)	2.77	-	211.8	-	-	DiT-XL(cfg = 1.5) [50]	3.04	5.02	240.82	0.84	0.54
VDM++ [36]	2.12	-	267.7	-	-	SiT-XL(cfg = 1.5, SDE)	2.62	4.18	252.21	0.84	0.57
DiT-XL(cfg = 1.5) [50]	2.27	4.60	278.24	0.83	0.57						
SiT-XL(cfg = 1.5, ODE)	2.15	4.60	258.09	0.81	0.60						
SiT-XL(cfg = 1.5, SDE)	2.06	4.49	277.50	0.83	0.59						

4 Related Work

Transformers. The transformer architecture [67] has emerged as a powerful tool for application domains as diverse as vision [21, 49], language [68, 69], quantum chemistry [23], active matter systems [9], and biology [11]. Several works have built on DiT and have made improvements by modifying the architecture to internally include masked prediction layers [22, 70]; these choices are orthogonal to this work and may be fruitfully combined in future work.

Training and Sampling in Diffusions. Diffusion models arose from [25, 61, 64] and have close historical relationship with denoising methods [29, 30, 59]. Various

efforts have gone into improving the sampling algorithms behind these methods in the context of DDPM [62] and SBDM [33, 63]; these are also orthogonal to our studies and may be combined to push for better performance in future work. Improved Diffusion ODE [71] also studies several combinations of model parameterizations (velocity versus noise) and paths (VP versus Linear). Unlike our work, they focus on lower dimensional experiments, benchmark with likelihoods, and do not consider SDE sampling.

Interpolants and flow matching. Velocity parameterizations using the Linear interpolant were also studied in [41, 43], and were generalized to the manifold setting in [6]. A trade-off in bounds on the KL divergence between the target distribution and the model arises when considering sampling with SDEs versus ODE; [2] shows that minimizing the objectives presented in this work controls KL for SDEs, but not for ODEs. Error bounds for SDE-based sampling with score-based diffusion models are studied in [13, 14, 38, 39], for ODE-base sampling are explored in [7, 15], in addition to the Wasserstein bounds provided in [4].

Other related works make improvements by changing how noise and data are sampled during training. [52, 65] compute mini-batch optimal couplings between the Gaussian and data distribution to reduce the transport cost and gradient variance; [3] instead build the coupling by flowing directly from the conditioning variable to the data for image-conditional tasks. Finally, various work considers learning a stochastic bridge connecting two arbitrary distributions [18, 44, 51, 58]. These directions are compatible with our investigations; they specify the learning problem for which one can vary the choices of model parameterizations, interpolant schedules, and sampling algorithms.

Diffusion in Latent Space. Generative modeling in latent space [53, 66] is a tractable approach for modeling high-dimensional data. The approach has been applied beyond images to video generation [8], which is a yet-to-be explored and promising application area for velocity trained models. [17] also train velocity models in the latent space of the pre-trained Stable Diffusion VAE. They demonstrate promising results for the DiT-B backbone with a final FID-50K of 4.46; their study was one motivation for the investigation in this work regarding which aspects of these models contribute to the gains in performance over DiT.

5 Conclusion

In this work, we have presented Scalable Interpolant Transformers, a simple and powerful framework for image generation tasks. Within the framework, we explored the tradeoffs between a number of key design choices: the choice of a continuous or discrete-time model, the choice of interpolant, the choice of model prediction, and the choice of diffusion coefficient. We highlighted the advantages and disadvantages of each choice and demonstrated how careful decisions can lead to significant performance improvements. Many concurrent works [24, 32, 42, 47] explore similar approaches in a wide variety of downstream tasks, and we leave the application of SiT to these tasks for future works.

Acknowledgements

We would like to thank Adithya Iyer, Sai Charitha Akula, Fred Lu, Jiatao Gu, and Edwin P. Gerber for helpful discussions and feedback. The research is partly supported by the Google TRC program.

References

1. Albergo, M.S., Boffi, N.M., Lindsey, M., Vanden-Eijnden, E.: Multimarginal generative modeling with stochastic interpolants. arXiv preprint arXiv:2310.03695 (2023)
2. Albergo, M.S., Boffi, N.M., Vanden-Eijnden, E.: Stochastic interpolants: A unifying framework for flows and diffusions. arXiv preprint arXiv:2303.08797 (2023)
3. Albergo, M.S., Goldstein, M., Boffi, N.M., Ranganath, R., Vanden-Eijnden, E.: Stochastic interpolants with data-dependent couplings. arXiv preprint arXiv:2310.03725 (2023)
4. Albergo, M.S., Vanden-Eijnden, E.: Building normalizing flows with stochastic interpolants. In: ICLR (2023)
5. Anderson, B.D.: Reverse-time diffusion equation models. Stochastic Processes and their Applications (1982)
6. Ben-Hamu, H., Cohen, S., Bose, J., Amos, B., Grover, A., Nickel, M., Chen, R.T., Lipman, Y.: Matching normalizing flows and probability paths on manifolds. In: ICML (2022)
7. Benton, J., Deligiannidis, G., Doucet, A.: Error bounds for flow matching methods. arXiv preprint arXiv:2305.16860 (2023)
8. Blattmann, A., Rombach, R., Ling, H., Dockhorn, T., Kim, S.W., Fidler, S., Kreis, K.: Align your latents: High-resolution video synthesis with latent diffusion models. In: CVPR (2023)
9. Boffi, N.M., Vanden-Eijnden, E.: Deep learning probability flows and entropy production rates in active matter. arXiv preprint arXiv:2309.12991 (2023)
10. Brock, A., Donahue, J., Simonyan, K.: Large scale gan training for high fidelity natural image synthesis. In: ICLR (2019)
11. Chandra, A., Tünnermann, L., Löfstedt, T., Gratz, R.: Transformer-based deep learning for predicting protein properties in the life sciences. Elife **12**, e82819 (2023)
12. Chang, H., Zhang, H., Jiang, L., Liu, C., Freeman, W.T.: Maskgit: Masked generative image transformer. In: CVPR (2022)
13. Chen, H., Lee, H., Lu, J.: Improved analysis of score-based generative modeling: User-friendly bounds under minimal smoothness assumptions. In: ICML (2023)
14. Chen, S., Chewi, S., Li, J., Li, Y., Salim, A., Zhang, A.: Sampling is as easy as learning the score: theory for diffusion models with minimal data assumptions. In: ICLR (2023)
15. Chen, S., Daras, G., Dimakis, A.: Restoration-degradation beyond linear diffusions: A non-asymptotic analysis for DDIM-type samplers. In: ICML (2023)
16. Chen, T.: On the importance of noise scheduling for diffusion models. arXiv preprint arXiv:2301.10972 (2023)
17. Dao, Q., Phung, H., Nguyen, B., Tran, A.: Flow matching in latent space. arXiv preprint arXiv:2307.08698 (2023)
18. De Bortoli, V., Thornton, J., Heng, J., Doucet, A.: Diffusion schrödinger bridge with applications to score-based generative modeling. In: NeurIPS (2021)

19. Dhariwal, P., Nichol, A.: Diffusion models beat gans on image synthesis. In: NIPS (2021)
20. Dockhorn, T., Vahdat, A., Kreis, K.: Score-based generative modeling with critically-damped langevin diffusion. In: ICLR (2022)
21. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: ICLR (2021)
22. Gao, S., Zhou, P., Cheng, M.M., Yan, S.: Masked diffusion transformer is a strong image synthesizer. arXiv preprint arXiv:2303.14389 (2023)
23. von Glehn, I., Spencer, J.S., Pfau, D.: A Self-Attention Ansatz for Ab-initio Quantum Chemistry. In: ICLR (2023)
24. Gupta, A., Yu, L., Sohn, K., Gu, X., Hahn, M., Fei-Fei, L., Essa, I., Jiang, L., Lezama, J.: Photorealistic video generation with diffusion models. arXiv preprint arXiv:2312.06662 (2023)
25. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. In: NeurIPS (2020)
26. Ho, J., Saharia, C., Chan, W., Fleet, D.J., Norouzi, M., Salimans, T.: Cascaded diffusion models for high fidelity image generation. arXiv preprint arXiv:2106.15282 (2021)
27. Ho, J., Salimans, T.: Classifier-free diffusion guidance. arXiv preprint arXiv:2207.12598 (2022)
28. Hoogeboom, E., Heek, J., Salimans, T.: simple diffusion: End-to-end diffusion for high resolution images. In: ICML (2023)
29. Hyvärinen, A.: Estimation of non-normalized statistical models by score matching. JMLR (2005)
30. Hyvärinen, A.: Sparse code shrinkage: Denoising of nongaussian data by maximum likelihood estimation. Neural Computation (1999)
31. Jabri, A., Fleet, D., Chen, T.: Scalable adaptive computation for iterative generation. In: ICML (2023)
32. Jakab, T., Li, R., Wu, S., Rupprecht, C., Vedaldi, A.: Farm3D: Learning articulated 3d animals by distilling 2d diffusion. In: 3DV (2024)
33. Karras, T., Aittala, M., Aila, T., Laine, S.: Elucidating the design space of diffusion-based generative models. In: NeurIPS (2022)
34. Kidger, P.: On Neural Differential Equations. Ph.D. thesis, University of Oxford (2021)
35. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015)
36. Kingma, D.P., Gao, R.: Understanding the diffusion objective as a weighted integral of elbos. arXiv preprint arXiv:2303.00848 (2023)
37. Kingma, D.P., Salimans, T., Poole, B., Ho, J.: Variational diffusion models. In: NeurIPS (2021)
38. Lee, H., Lu, J., Tan, Y.: Convergence for score-based generative modeling with polynomial complexity. In: NeurIPS (2022)
39. Lee, H., Lu, J., Tan, Y.: Convergence of score-based generative modeling for general data distributions. In: ALT (2023)
40. Lee, S., Kim, B., Ye, J.C.: Minimizing trajectory curvature of ode-based generative models. In: ICML (2023)
41. Lipman, Y., Chen, R.T.Q., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. In: ICLR (2023)
42. Liu, R., Wu, R., Hoorick, B.V., Tokmakov, P., Zakharov, S., Vondrick, C.: Zero-1-to-3: Zero-shot one image to 3d object. In: ICCV (2023)

43. Liu, X., Gong, C., Liu, Q.: Flow straight and fast: Learning to generate and transfer data with rectified flow. In: ICLR (2023)
44. Liu, X., Wu, L., Ye, M., Liu, Q.: Let us build bridges: Understanding and extending diffusion generative models. arXiv preprint arXiv:2208.14699 (2022)
45. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: ICLR (2019)
46. Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., Zhu, J.: Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. In: NeurIPS (2022)
47. Meng, C., He, Y., Song, Y., Song, J., Wu, J., Zhu, J.Y., Ermon, S.: Sdedit: Guided image synthesis and editing with stochastic differential equations. In: ICLR (2022)
48. Nichol, A., Dhariwal, P.: Improved denoising diffusion probabilistic models. In: ICML (2021)
49. Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, L., Shazeer, N., Ku, A., Tran, D.: Image Transformer. In: ICML (2018)
50. Peebles, W., Xie, S.: Scalable diffusion models with transformers. In: ICCV (2023)
51. Peluchetti, S.: Non-denoising forward-time diffusions. In: ICLR (2022)
52. Pooladian, A.A., Ben-Hamu, H., Domingo-Enrich, C., Amos, B., Lipman, Y., Chen, R.T.Q.: Multisample flow matching: Straightening flows with minibatch couplings. In: ICML (2023)
53. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: CVPR (2022)
54. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: MICCAI (2015)
55. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. IJCV (2015)
56. Salimans, T., Ho, J.: Progressive distillation for fast sampling of diffusion models. In: ICLR (2022)
57. Sauer, A., Schwarz, K., Geiger, A.: Stylegan-xl: Scaling stylegan to large diverse datasets. In: SIGGRAPH (2022)
58. Shi, Y., Bortoli, V.D., Campbell, A., Doucet, A.: Diffusion schrödinger bridge matching. In: NIPS (2023)
59. Simoncelli, E.P., Adelson, E.H.: Noise removal via bayesian wavelet coring. In: ICIP (1996)
60. Singhal, R., Goldstein, M., Ranganath, R.: Where to diffuse, how to diffuse, and how to get back: Automated learning for multivariate diffusions. In: ICLR (2023)
61. Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using nonequilibrium thermodynamics. In: ICML (2015)
62. Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. In: ICLR (2021)
63. Song, Y., Durkan, C., Murray, I., Ermon, S.: Maximum likelihood training of score-based diffusion models. In: NeurIPS (2021)
64. Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. In: ICLR (2021)
65. Tong, A., Malkin, N., Huguet, G., Zhang, Y., Rector-Brooks, J., Fatras, K., Wolf, G., Bengio, Y.: Improving and generalizing flow-based generative models with minibatch optimal transport. In: ICML Workshop on New Frontiers in Learning, Control, and Dynamical Systems (2023)
66. Vahdat, A., Kreis, K., Kautz, J.: Score-based generative modeling in latent space. In: NIPS (2021)
67. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NIPS (2017)

68. Wang, Q., Li, B., Xiao, T., Zhu, J., Li, C., Wong, D.F., Chao, L.S.: Learning deep transformer models for machine translation. In: ACL (2019)
69. Zaheer, M., Guruganesh, G., Dubey, A., Ainslie, J., Alberti, C., Ontanon, S., Pham, P., Ravula, A., Wang, Q., Yang, L., Ahmed, A.: Big Bird: Transformers for Longer Sequences. In: NeurIPS (2020)
70. Zheng, H., Nie, W., Vahdat, A., Anandkumar, A.: Fast training of diffusion models with masked transformers. arXiv preprint arXiv:2306.09305 (2023)
71. Zheng, K., Lu, C., Chen, J., Zhu, J.: Improved techniques for maximum likelihood estimation for diffusion odes. In: ICML (2023)

A Proofs

In all proofs below, we use \cdot for dot product and assume all bold notations (\mathbf{x} , $\boldsymbol{\varepsilon}$, etc.) are real-valued vectors in \mathbb{R}^d . Most proofs are derived from [2].

A.1 Proof of the probability flow ODE (2) with the velocity in Eq. (3).

Consider the time-dependent probability density function (PDF) $p_t(\mathbf{x})$ of $\mathbf{x}_t = \alpha_t \mathbf{x}_* + \sigma_t \boldsymbol{\varepsilon}$ defined in Eq. (1). By definition, its characteristic function $\hat{p}_t(\mathbf{k}) = \int_{\mathbb{R}^d} e^{i\mathbf{k}\cdot\mathbf{x}} p_t(\mathbf{x}) d\mathbf{x}$ is given by

$$\hat{p}_t(\mathbf{k}) = \mathbb{E}[e^{i\mathbf{k}\cdot\mathbf{x}_t}] \quad (15)$$

where \mathbb{E} denotes expectation over \mathbf{x}_* and $\boldsymbol{\varepsilon}$. Taking time derivative on both sides, and using the tower property of conditional expectation, we have

$$\partial_t \hat{p}_t(\mathbf{k}) = i\mathbf{k} \cdot \mathbb{E}[\dot{\mathbf{x}}_t e^{i\mathbf{k}\cdot\mathbf{x}_t}] \quad (16)$$

$$= i\mathbf{k} \cdot \mathbb{E}_{\mathbf{x} \sim p_t} [\mathbb{E}[\dot{\mathbf{x}}_t e^{i\mathbf{k}\cdot\mathbf{x}_t} | \mathbf{x}_t = \mathbf{x}]] \quad (17)$$

$$= i\mathbf{k} \cdot \mathbb{E}_{\mathbf{x} \sim p_t} [\mathbb{E}[(\dot{\alpha}_t \mathbf{x}_* + \dot{\sigma}_t \boldsymbol{\varepsilon}) e^{i\mathbf{k}\cdot\mathbf{x}_t} | \mathbf{x}_t = \mathbf{x}]] \quad (18)$$

$$= i\mathbf{k} \cdot \mathbb{E}_{\mathbf{x} \sim p_t} [\mathbb{E}[(\dot{\alpha}_t \mathbf{x}_* + \dot{\sigma}_t \boldsymbol{\varepsilon}) | \mathbf{x}_t = \mathbf{x}] e^{i\mathbf{k}\cdot\mathbf{x}}] \quad (19)$$

$$= i\mathbf{k} \cdot \mathbb{E}_{\mathbf{x} \sim p_t} [\mathbf{v}(\mathbf{x}, t) e^{i\mathbf{k}\cdot\mathbf{x}}] \quad (20)$$

where $\mathbf{v}(\mathbf{x}, t) = \mathbb{E}[(\dot{\alpha}_t \mathbf{x}_* + \dot{\sigma}_t \boldsymbol{\varepsilon}) | \mathbf{x}_t = \mathbf{x}] = \dot{\alpha}_t \mathbb{E}[\mathbf{x}_* | \mathbf{x}_t = \mathbf{x}] + \dot{\sigma}_t \mathbb{E}[\boldsymbol{\varepsilon} | \mathbf{x}_t = \mathbf{x}]$ is the velocity defined in Eq. (3). Explicitly, Eq. (20) reads

$$\partial_t \int_{\mathbb{R}^d} e^{i\mathbf{k}\cdot\mathbf{x}} p_t(\mathbf{x}) d\mathbf{x} = i\mathbf{k} \cdot \int_{\mathbb{R}^d} \mathbf{v}(\mathbf{x}, t) e^{i\mathbf{k}\cdot\mathbf{x}} p_t(\mathbf{x}) d\mathbf{x} \quad (21)$$

from which we deduce

$$\int_{\mathbb{R}^d} e^{i\mathbf{k}\cdot\mathbf{x}} \partial_t p_t(\mathbf{x}) d\mathbf{x} = \int_{\mathbb{R}^d} \mathbf{v}(\mathbf{x}, t) \cdot \nabla_{\mathbf{x}} [e^{i\mathbf{k}\cdot\mathbf{x}}] p_t(\mathbf{x}) d\mathbf{x} \quad (22)$$

$$= - \int_{\mathbb{R}^d} \nabla_{\mathbf{x}} \cdot [\mathbf{v}(\mathbf{x}, t) p_t(\mathbf{x})] e^{i\mathbf{k}\cdot\mathbf{x}} d\mathbf{x} \quad (23)$$

where $\nabla_{\mathbf{x}} \cdot [\mathbf{v} p_t] = \sum_{i=1}^d \frac{\partial}{\partial x_i} [v_i p_t]$ is the divergence operator and we used integration by parts to get the second equality. By the properties of Fourier transform, Eq. (23) implies that $p_t(\mathbf{x})$ satisfies the transport equation

$$\partial_t p_t(\mathbf{x}) + \nabla_{\mathbf{x}} \cdot (\mathbf{v}(\mathbf{x}, t) p_t(\mathbf{x})) = 0. \quad (24)$$

Solving this equation by the method of characteristic leads to probability flow ODE (2).

A.2 Proof of the SDE (4)

We show that the SDE (4) has marginal density $p_t(\mathbf{x})$ with any choice of $w_t \geq 0$. To this end, recall that solution to the SDE

$$d\mathbf{X}_t = [\mathbf{v}(\mathbf{X}_t, t) - \frac{1}{2}w_t \mathbf{s}(\mathbf{X}_t, t)]dt + \sqrt{w_t} d\bar{\mathbf{W}}_t$$

has a PDF that satisfies the Fokker-Planck equation

$$\partial_t p_t(\mathbf{x}) = -\nabla_{\mathbf{x}} \cdot ([\mathbf{v}(\mathbf{x}, t) - \frac{1}{2}w_t \mathbf{s}(\mathbf{x}, t)]p_t(\mathbf{x})) - \frac{1}{2}w_t \Delta_{\mathbf{x}} p_t(\mathbf{x}) \quad (25)$$

where $\Delta_{\mathbf{x}}$ is the Laplace operator defined as $\Delta_{\mathbf{x}} = \nabla_{\mathbf{x}} \cdot \nabla_{\mathbf{x}} = \sum_{i=0}^d \frac{\partial^2}{\partial x_i^2}$. Reorganizing the equation and using the definition of the score $\mathbf{s}(\mathbf{x}, t) = \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) = p_t^{-1}(\mathbf{x}) \nabla_{\mathbf{x}} p_t(\mathbf{x})$, we have

$$\begin{aligned} \partial_t p_t(\mathbf{x}) &= \underbrace{-\nabla_{\mathbf{x}} \cdot [\mathbf{v}(\mathbf{x}, t)p_t(\mathbf{x})]}_{= \partial_t p_t(\mathbf{x}) \text{ by Eq. (24)}} + \frac{1}{2}w_t \nabla_{\mathbf{x}} \cdot [\underbrace{\nabla_{\mathbf{x}} \log p_t(\mathbf{x})p_t(\mathbf{x})}_{= \nabla_{\mathbf{x}} p_t(\mathbf{x})}] - \frac{1}{2}w_t \Delta_{\mathbf{x}} p_t(\mathbf{x}) \\ &\quad (26) \end{aligned}$$

$$\implies 0 = \frac{1}{2}w_t \nabla_{\mathbf{x}} \cdot \nabla_{\mathbf{x}} p_t(\mathbf{x}) - \frac{1}{2}w_t \Delta_{\mathbf{x}} p_t(\mathbf{x}) \quad (27)$$

By definition of Laplace operator, the last equation holds for any $w_t \geq 0$. When $w_t = 0$, the Fokker-Planck equation reduces to a continuity equation, and the SDE reduces to an ODE, so the connection trivially holds.

A.3 Proof of the expression for the score in Eq. (5)

We show that $\mathbf{s}(\mathbf{x}, t) = -\sigma_t^{-1} \mathbb{E}[\boldsymbol{\varepsilon} | \mathbf{x}_t = \mathbf{x}]$. Letting $\hat{f}(\mathbf{k}, t) = \mathbb{E}[\boldsymbol{\varepsilon} e^{i\sigma_t \mathbf{k} \cdot \boldsymbol{\varepsilon}}]$, we have

$$\hat{f}(\mathbf{k}, t) = -\frac{i}{\sigma_t} \nabla_{\mathbf{k}} \mathbb{E}[e^{i\sigma_t \mathbf{k} \cdot \boldsymbol{\varepsilon}}] \quad (28)$$

Since $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, \mathbf{I})$, we can compute the expectation explicitly to obtain

$$\hat{f}(\mathbf{k}, t) = -\frac{i}{\sigma_t} (\nabla_{\mathbf{k}} e^{-\frac{1}{2}\sigma_t^2 |\mathbf{k}|^2}) \quad (29)$$

$$= i\sigma_t \mathbf{k} e^{-\frac{1}{2}\sigma_t^2 |\mathbf{k}|^2} \quad (30)$$

Since \mathbf{x}_* and $\boldsymbol{\varepsilon}$ are independent random variable, we have

$$\mathbb{E}[\boldsymbol{\varepsilon} e^{i\mathbf{k} \cdot \mathbf{x}_t}] = \hat{f}(\mathbf{k}, t) \mathbb{E}[e^{i\alpha_t \mathbf{k} \cdot \mathbf{x}_*}] = i\sigma_t \mathbf{k} \underbrace{e^{-\frac{1}{2}\sigma_t^2 |\mathbf{k}|^2} \mathbb{E}[e^{i\alpha_t \mathbf{k} \cdot \mathbf{x}_*}]}_{\text{combine this}} = i\sigma_t \mathbf{k} \hat{p}_t(\mathbf{k}) \quad (31)$$

where $\hat{p}_t(\mathbf{k})$ is the characteristic function of $\mathbf{x}_t = \alpha_t \mathbf{x}_* + \sigma_t \boldsymbol{\varepsilon}$ defined in Eq. (15). The left hand-side of this equation can also be written as:

$$\mathbb{E}[\boldsymbol{\varepsilon} e^{i\mathbf{k} \cdot \mathbf{x}_t}] = \int_{\mathbb{R}^d} \mathbb{E}[\boldsymbol{\varepsilon} e^{i\mathbf{k} \cdot \mathbf{x}_t} | \mathbf{x}_t = \mathbf{x}] p_t(\mathbf{x}) d\mathbf{x} \quad (32)$$

$$= \int_{\mathbb{R}^d} \mathbb{E}[\boldsymbol{\varepsilon} | \mathbf{x}_t = \mathbf{x}] e^{i\mathbf{k} \cdot \mathbf{x}} p_t(\mathbf{x}) d\mathbf{x}, \quad (33)$$

whereas the right hand-side is

$$i\sigma_t \mathbf{k} \hat{p}_t(\mathbf{k}) = i\sigma_t \mathbf{k} \int_{\mathbb{R}^d} e^{i\mathbf{k} \cdot \mathbf{x}} p_t(\mathbf{x}) d\mathbf{x} \quad (34)$$

$$= \sigma_t \int_{\mathbb{R}^d} \nabla_{\mathbf{x}}[e^{i\mathbf{k} \cdot \mathbf{x}}] p_t(\mathbf{x}) d\mathbf{x} \quad (35)$$

$$= -\sigma_t \int_{\mathbb{R}^d} e^{i\mathbf{k} \cdot \mathbf{x}} \nabla_{\mathbf{x}} p_t(\mathbf{x}) d\mathbf{x} \quad (36)$$

$$= -\sigma_t \int_{\mathbb{R}^d} e^{i\mathbf{k} \cdot \mathbf{x}} \mathbf{s}(\mathbf{x}, t) p_t(\mathbf{x}) d\mathbf{x} \quad (37)$$

where we used integration by parts to get the third equality, and again the definition of the score to get the last.

Comparing Eq. (33) and Eq. (37) we deduce that, when $\sigma_t \neq 0$,

$$\mathbf{s}(\mathbf{x}, t) = \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) = -\sigma_t^{-1} \mathbb{E}[\boldsymbol{\varepsilon} | \mathbf{x}_t = \mathbf{x}] \quad (38)$$

Further, setting w_t to σ_t in Eq. (4) gives

$$\frac{1}{2} w_t \mathbf{s}(\mathbf{x}_t, t) = -\frac{1}{2} \mathbb{E}[\boldsymbol{\varepsilon} | \mathbf{x}_t = \mathbf{x}] \quad (39)$$

for all $t \in [0, 1]$. This bypass the constraint of $\sigma_t \neq 0$ and effectively eliminate the singularity at $t = 0$.

A.4 Proof of Eq. (9)

We note that there exists a straightforward connection between $\mathbf{v}(\mathbf{x}, t)$ and $\mathbf{s}(\mathbf{x}, t)$. From Eq. (1), we have

$$\mathbf{v}(\mathbf{x}, t) = \dot{\alpha}_t \mathbb{E}[\mathbf{x}_* | \mathbf{x}_t = \mathbf{x}] + \dot{\sigma}_t \mathbb{E}[\boldsymbol{\varepsilon} | \mathbf{x}_t = \mathbf{x}] \quad (40)$$

$$= \dot{\alpha}_t \mathbb{E}\left[\frac{\mathbf{x}_t - \sigma_t \boldsymbol{\varepsilon}}{\alpha_t} | \mathbf{x}_t = \mathbf{x}\right] + \dot{\sigma}_t \mathbb{E}[\boldsymbol{\varepsilon} | \mathbf{x}_t = \mathbf{x}] \quad (41)$$

$$= \frac{\dot{\alpha}_t}{\alpha_t} \mathbf{x} + (\dot{\sigma}_t - \frac{\dot{\alpha}_t \sigma_t}{\alpha_t}) \mathbb{E}[\boldsymbol{\varepsilon} | \mathbf{x}_t = \mathbf{x}] \quad (42)$$

$$= \frac{\dot{\alpha}_t}{\alpha_t} \mathbf{x} + (\dot{\sigma}_t - \frac{\dot{\alpha}_t \sigma_t}{\alpha_t})(-\sigma_t \mathbf{s}(\mathbf{x}, t)) \quad (43)$$

$$= \frac{\dot{\alpha}_t}{\alpha_t} \mathbf{x} - \lambda_t \sigma_t \mathbf{s}(\mathbf{x}, t) \quad (44)$$

where we defined

$$\lambda_t = \dot{\sigma}_t - \frac{\dot{\alpha}_t \sigma_t}{\alpha_t}. \quad (45)$$

Given Eq. (44) is linear in terms of \mathbf{s} , reverting it will lead to Eq. (9).

Note that we can also plug Eq. (44) into the loss $\mathcal{L}_{\mathbf{v}}$ in Eq. (7) to deduce that

$$\mathcal{L}_{\mathbf{v}}(\theta) = \int_0^T \mathbb{E}[\|\underbrace{\dot{\alpha}_t}_{\alpha_t} \mathbf{x} + \lambda_t(-\sigma_t \mathbf{s}_\theta(\mathbf{x}_t, t)) - \dot{\alpha}_t \mathbf{x}_* - \dot{\sigma}_t \boldsymbol{\varepsilon}\|^2] dt \quad (46)$$

Expand to $\mathbf{x}_t = \alpha_t \mathbf{x}_* + \sigma_t \boldsymbol{\varepsilon}$

$$= \int_0^T \mathbb{E}[\|\dot{\alpha}_t \mathbf{x}_* + \frac{\dot{\alpha}_t \sigma_t}{\alpha_t} \boldsymbol{\varepsilon} + \lambda_t(-\sigma_t \mathbf{s}_\theta(\mathbf{x}_t, t)) - \dot{\alpha}_t \mathbf{x}_* - \dot{\sigma}_t \boldsymbol{\varepsilon}\|^2] dt \quad (47)$$

$$= \int_0^T \mathbb{E}[\|\lambda_t(-\sigma_t \mathbf{s}_\theta(\mathbf{x}_t, t)) - \lambda_t \boldsymbol{\varepsilon}\|^2] dt \quad (48)$$

$$= \int_0^T \lambda_t^2 \mathbb{E}[\|\sigma_t \mathbf{s}_\theta(\mathbf{x}_t, t) + \boldsymbol{\varepsilon}\|^2] dt \quad (49)$$

$$\equiv \mathcal{L}_{s_\lambda}(\theta) \quad (50)$$

which defines the weighted score objective $\mathcal{L}_{s_\lambda}(\theta)$. This observation is consistent with the claim made in [36] that the score objective with different monotonic weighting functions coincides with losses for different model parameterizations. In Appendix B we show that λ_t corresponds to the square of the maximum likelihood weighting proposed in [63] and [66].

A.5 Proof for the optimal w_t for tightening the KL bound

Lemma 2.22 in [2] asserts that:

$$D_{\text{KL}}(p(\mathbf{x}) \| p_\theta(\mathbf{x})) \leq \frac{1}{2} \int_0^1 w_t^{-1} \int_{\Omega} |b(\mathbf{x}, t) - b_\theta(\mathbf{x}, t)|^2 p_t(\mathbf{x}) dt d\mathbf{x} \quad (51)$$

where $p(\mathbf{x})$ denotes the true data distribution, $p_\theta(\mathbf{x})$ denotes the approximated data distribution by our model at time $t = 0$, and p_t corresponds to the marginal density in Appendix A.2. We further use b and b_θ to refer to the ground truth and approximated drift for the reverse SDE, respectively; that is, $b(\mathbf{x}, t) = v(\mathbf{x}, t) - \frac{1}{2} w_t s(\mathbf{x}, t)$. Following Appendix A.4, $b(\mathbf{x}, t)$ can be expressed in terms of $v(\mathbf{x}, t)$

$$b(\mathbf{x}, t) = (1 + \frac{1}{2} \frac{w_t}{\lambda_t \sigma_t}) v(\mathbf{x}, t) - \frac{1}{2} \frac{w_t \dot{\alpha}_t}{\alpha_t \lambda_t \sigma_t} x \quad (52)$$

and similarly for $b_\theta(\mathbf{x}, t)$. Plug back, Eq. (51) becomes

$$D_{\text{KL}}(p(\mathbf{x}) \| p_\theta(\mathbf{x})) \leq \frac{1}{2} \int_0^1 w_t^{-1} (1 + \frac{1}{2} \frac{w_t}{\lambda_t \sigma_t})^2 \int_{\Omega} |v(\mathbf{x}, t) - v_\theta(\mathbf{x}, t)|^2 p_t(\mathbf{x}) dt d\mathbf{x} \quad (53)$$

Since $v(\mathbf{x}, t) = \mathbb{E}[\dot{\mathbf{x}} | \mathbf{x}_t = \mathbf{x}]$ from Eq. (1), we have

$$\begin{aligned} \int_{\Omega} |v(\mathbf{x}, t) - v_{\theta}(\mathbf{x}, t)|^2 p_t(\mathbf{x}) d\mathbf{x} &= \mathbb{E}[|v(\mathbf{x}, t) - v_{\theta}(\mathbf{x}, t)|^2] \\ &\leq \mathbb{E}[|\dot{\mathbf{x}} - v_{\theta}(\mathbf{x}, t)|^2] \equiv \mathcal{L}_t \end{aligned} \quad (54)$$

where \mathcal{L}_t is the loss at time t of our model after optimization. With Eq. (55) we can further simplify Eq. (54) to be

$$D_{\text{KL}}(p(\mathbf{x}) \| p_{\theta}(\mathbf{x})) \leq \frac{1}{2} \int_0^1 w_t^{-1} \left(1 + \frac{1}{2} \frac{w_t}{\lambda_t \sigma_t}\right)^2 \mathcal{L}_t dt \quad (55)$$

We note the minimum of the integrand in Eq. (55) is achieved at $w_t = 2\lambda_t \sigma_t$ with a value of $2\frac{\mathcal{L}_t}{\lambda_t \sigma_t}$. We note that such w_t is the exact choice of w_t^{KL} in Sec. 2.4.

For SBDM-VP interpolant, such w_t^{KL} coincides with β_t in [64], and is well defined and positive everywhere on $[0, 1]$. For GVP and Linear interpolant however, this diffusion coefficient is zero at $t = 0$ and infinity at $t = 1$. Since $\sigma_t = O(t)$ at $t = 0$, the integrand $2\frac{\mathcal{L}_t}{\lambda_t \sigma_t}$ is not integrable at $t = 0$, making the bound in Eq. (55) trivially ∞ unless $\lim_{t \rightarrow 0} \mathcal{L}_t = 0$.

We note the bound proposed in Eq. (55) does not account for the cost of time-integration of the SDE. Assuming that the non-uniform integration step Δt one must take to maintain a given precision is inversely proportional to w_t , that is $\Delta t = O(w_t^{-1})$, we can account for the integration cost by adding a term to Eq. (55)

$$D_{\text{KL}}(p(\mathbf{x}) \| p_{\theta}(\mathbf{x})) \leq \frac{1}{2} \int_0^1 w_t^{-1} \left(1 + \frac{1}{2} \frac{w_t}{\lambda_t \sigma_t}\right)^2 \mathcal{L}_t dt + \eta \int_0^1 w_t dt \quad (56)$$

where $\eta > 0$ is a parameter that controls the integration error: the higher the η , the smaller the cost but the higher the error, and vice-versa. The minimum of the integrand in Eq. (56) is

$$\min_{w_t} \left(w_t^{-1} \left(1 + \frac{1}{2} \frac{w_t}{\lambda_t \sigma_t}\right)^2 \mathcal{L}_t + \eta w_t \right) = \frac{2\mathcal{L}_t}{\lambda_t \sigma_t} \left(1 + \sqrt{\frac{4\eta \lambda_t^2 \sigma_t^2 + \mathcal{L}_t}{\mathcal{L}_t}}\right) \quad (57)$$

and it is achieved at

$$w_t = 2\lambda_t \sigma_t \sqrt{\frac{\mathcal{L}_t}{4\eta \lambda_t^2 \sigma_t^2 + \mathcal{L}_t}} \quad (58)$$

This is exactly the choice of $w_t^{\text{KL}, \eta}$ in Sec. 2.4. We note that such diffusion coefficient is well defined everywhere on $[0, 1]$ if \mathcal{L}_t is also well defined everywhere, and as $t \rightarrow 1$, it approaches a finite limit at $\sqrt{\frac{\mathcal{L}_{t \rightarrow 1}}{2\eta}}$.

We also note that the integrand in Eq. (56) would still be ∞ at time 0 given the $\frac{1}{\lambda_t \sigma_t}$, unless $\lim_{t \rightarrow 0} \mathcal{L}_t = 0$. Theoretically, this is not an unreasonable assumption for both coefficients, as we know the closed form of $v(\mathbf{x}, 0) = \mathbb{E}[\dot{\mathbf{x}}_0 | \mathbf{x}_{t=0} = \mathbf{x}] = \dot{\mathbf{x}}_0 \mathbb{E}[\mathbf{x}_*]$ and could optimize our model $v_{\theta}(\mathbf{x}, t)$ to directly approximate this value at $t = 0$. In practice, we found the numerical stability of $w_t^{\text{KL}, \eta}$ could lead to better results.

B Connection with Score-based Diffusion

As shown in [64], the reverse-time SDE from Eq. (10) is

$$d\mathbf{X}_t = [-\frac{1}{2}\beta_t \mathbf{X}_t - \beta_t \mathbf{s}(\mathbf{X}_t, t)]dt + \sqrt{\beta_t} d\bar{\mathbf{W}}_t \quad (59)$$

Let us show this SDE is Eq. (4) for the specific choice $w_t = \beta_t$. To this end, notice that the solution \mathbf{X}_t to Eq. (59) for the initial condition $\mathbf{X}_{t=0} = \mathbf{x}_*$ with \mathbf{x}_* fixed is Gaussian distributed with mean and variance given respectively by

$$\mathbb{E}[\mathbf{X}_t] = e^{-\frac{1}{2} \int_0^t \beta_s ds} \mathbf{x}_* \equiv \alpha_t \mathbf{x}_* \quad (60)$$

$$\text{var}[\mathbf{X}_t] = 1 - e^{-\int_0^t \beta_s ds} \equiv \sigma_t^2 \quad (61)$$

Using Eq. (44), the velocity of the score-based diffusion model can therefore be expressed as

$$\mathbf{v}(\mathbf{x}, t) = -\frac{1}{2}\beta_t \mathbf{x} + (-\frac{1}{2}\beta_t(1 - e^{-\int_0^t \beta_s ds}) - \frac{1}{2}\beta_t e^{-\int_0^t \beta_s ds}) \mathbf{s}(\mathbf{x}, t) \quad (62)$$

$$= -\frac{1}{2}\beta_t \mathbf{x} - \frac{1}{2}\beta_t \mathbf{s}(\mathbf{x}, t) \quad (63)$$

we see that $2\lambda_t \sigma_t$ is precisely β_t , making λ_t correspond to the square of maximum likelihood weighting proposed in [64]. Further, if we plug Eq. (63) into Eq. (4), we arrive at Eq. (59).

A useful observation for choosing velocity versus noise model. We see that in the velocity model, all of the path-dependent terms (α_t , σ_t) are inside the squared loss, and in the score model, the terms are pulled out (apart from the necessary σ_t in score matching loss) and get squared due to coming out of the norm. So which is more stable depends on the interpolant. In the paper we see that for SBDM-VP, due to the blowing up behavior of $\dot{\sigma}_t$ near $t = 0$, both \mathcal{L}_v and \mathcal{L}_{s_λ} are unstable.

Yet, shown in Tab. 3, we observed better performance with \mathcal{L}_{s_λ} for SBDM-VP, as the blowing up λ_t near $t = 0$ will compensate for the diminishing gradient inside the squared norm, where \mathcal{L}_v would simply experience gradient explosion resulted from $\dot{\sigma}_t$. The behavior is different for the Linear and GVP interpolant, where the source of instability is α_t^{-1} near $t = 1$. We note \mathcal{L}_v is stable since α_t^{-1} gets cancelled out inside the squared norm, while in \mathcal{L}_{s_λ} it remains in λ_t outside the norm.

C Sampling with Guidance

Let $p_t(\mathbf{x}|\mathbf{y})$ be the density of $\mathbf{x}_t = \alpha_t \mathbf{x}_* + \sigma_t \boldsymbol{\varepsilon}$ conditioned on some extra variable \mathbf{y} . By argument similar to the one given in Appendix A.1, it is easy to see that $p_t(\mathbf{x}|\mathbf{y})$ satisfies the transport equation (compare Eq. (24))

$$\partial_t p_t(\mathbf{x}|\mathbf{y}) + \nabla_{\mathbf{x}} \cdot (\mathbf{v}(\mathbf{x}, t|\mathbf{y}) p_t(\mathbf{x}, |\mathbf{y})) = 0, \quad (64)$$

where (compare Eq. (3))

$$\mathbf{v}(\mathbf{x}, t|\mathbf{y}) = \mathbb{E}[\dot{\mathbf{x}}_t | \mathbf{x}_t = \mathbf{x}, \mathbf{y}] = \dot{\alpha}_t \mathbb{E}[\mathbf{x}_* | \mathbf{x}_t = \mathbf{x}, \mathbf{y}] + \dot{\sigma}_t \mathbb{E}[\boldsymbol{\varepsilon} | \mathbf{x}_t = \mathbf{x}, \mathbf{y}] \quad (65)$$

Proceeding as in Appendix A.3 and Appendix A.4, it is also easy to see that the score $\mathbf{s}(\mathbf{x}, t|\mathbf{y}) = \nabla_{\mathbf{x}} \log p_t(\mathbf{x}|\mathbf{y})$ is given by (compare Eq. (5))

$$\mathbf{s}(\mathbf{x}, t|\mathbf{y}) = -\sigma_t^{-1} \mathbb{E}[\boldsymbol{\varepsilon} | \mathbf{x}_t = \mathbf{x}, \mathbf{y}] \quad (66)$$

and that $\mathbf{v}(\mathbf{x}, t|\mathbf{y})$ and $\mathbf{s}(\mathbf{x}, t|\mathbf{y})$ are related via (compare Eq. (44))

$$\mathbf{v}(\mathbf{x}, t|\mathbf{y}) = \frac{\dot{\alpha}_t}{\alpha_t} \mathbf{x} - \lambda_t \sigma_t \mathbf{s}(\mathbf{x}, t|\mathbf{y}) \quad (67)$$

Consider now

$$\mathbf{s}^\zeta(\mathbf{x}, t|\mathbf{y}) \equiv (1 - \zeta) \mathbf{s}(\mathbf{x}, t) + \zeta \mathbf{s}(\mathbf{x}, t|\mathbf{y}) \quad (68)$$

$$= \nabla \log p_t(\mathbf{x}) - \zeta \nabla \log p_t(\mathbf{x}) + \zeta \nabla \log p_t(\mathbf{x}|\mathbf{y}) \quad (69)$$

$$= \nabla \log p_t(\mathbf{x}) - \zeta \nabla \log p_t(\mathbf{x}) + (\zeta \nabla \log p_t(\mathbf{y}|\mathbf{x}) + \zeta \nabla \log p_t(\mathbf{x})) \quad (70)$$

$$= \nabla \log p_t(\mathbf{x}) + \zeta \nabla \log p_t(\mathbf{y}|\mathbf{x}) \quad (71)$$

$$= \nabla \log [p_t(\mathbf{x}) p_t^\zeta(\mathbf{y}|\mathbf{x})] \quad (72)$$

where we have used the fact $\nabla_{\mathbf{x}} \log p_t(\mathbf{x}|\mathbf{y}) = \nabla_{\mathbf{x}} \log p_t(\mathbf{y}|\mathbf{x}) + \nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ that follows from $p_t(\mathbf{x}|\mathbf{y})p(\mathbf{y}) = p_t(\mathbf{y}|\mathbf{x})p_t(\mathbf{x})$, and ζ to be some constant greater than 1. Eq. (72) shows that using the score mixture $\mathbf{s}^\zeta(\mathbf{x}, t|\mathbf{y}) = (1 - \zeta) \mathbf{s}(\mathbf{x}, t) + \zeta \mathbf{s}(\mathbf{x}, t|\mathbf{y})$, and the velocity mixture associated with it, namely,

$$\mathbf{v}^\zeta(\mathbf{x}, t|\mathbf{y}) = (1 - \zeta) \mathbf{v}(\mathbf{x}, t) + \zeta \mathbf{v}(\mathbf{x}, t|\mathbf{y}) \quad (73)$$

$$= \frac{\dot{\alpha}_t}{\alpha_t} \mathbf{x} - \lambda_t \sigma_t [(1 - \zeta) \mathbf{s}(\mathbf{x}, t) + \zeta \mathbf{s}(\mathbf{x}, t|\mathbf{y})] \quad (74)$$

$$= \frac{\dot{\alpha}_t}{\alpha_t} \mathbf{x} - \lambda_t \sigma_t \mathbf{s}^\zeta(\mathbf{x}, t|\mathbf{y}), \quad (75)$$

allows one to construct generative models that sample the tempered distribution $p_t(\mathbf{x}_t)p_t^\zeta(\mathbf{y}|\mathbf{x}_t)$ following classifier guidance [19]. Note that $p_t(\mathbf{x})p_t^\zeta(\mathbf{y}|\mathbf{x}) \propto p_t^\zeta(\mathbf{x}|\mathbf{y})p_t^{1-\zeta}(\mathbf{x})$, so we can also perform classifier free guidance sampling [27]. Empirically, we observe significant performance boost by applying classifier free guidance, as showed in Tab. 1 and Tab. 7.

D Sampling with ODE and SDE

In the main body of the paper, we used a second order Heun integrator for solving the ODE in Eq. (2) and a first order Euler-Maruyama integrator for solving the SDE in Eq. (4). We summarize all results in Tab. 8, and present the implementations below.

Table 8: FID-50K scores produced by ODE and SDE. We demonstrate the comparison between ODE and SDE across all of our model sizes. All statistics are produced without classifier free guidance. Each cell in the table is showing [ODE results] / [SDE results]. We note the better performances of SDE observed in all model sizes are in line with the bounds given in [2], and that ODE has its advantage in lower NFE region, as shown in Fig. 5

Model	Training Steps(K)	FID↓	sFID↓	IS↑	Precision↑	Recall↑
SiT-S	400	58.97 / 57.64	8.95 / 9.05	23.34 / 24.78	0.40 / 0.41	0.59 / 0.60
SiT-B	400	34.84 / 33.02	6.59 / 6.46	41.53 / 43.71	0.52 / 0.53	0.64 / 0.63
SiT-L	400	20.01 / 18.79	5.31 / 5.29	67.76 / 72.02	0.62 / 0.64	0.64 / 0.64
SiT-XL	400	18.04 / 17.19	5.17 / 5.07	73.90 / 76.52	0.63 / 0.65	0.64 / 0.63
SiT-XL	7000	9.35 / 8.26	6.38 / 6.32	126.06 / 131.65	0.67 / 0.68	0.68 / 0.67

It is feasible to use either a velocity model \mathbf{v}_θ or a score model \mathbf{s}_θ in applying the above two samplers. If learning the score for the deterministic Heun sampler, we could always convert the learned \mathbf{s}_θ to \mathbf{v}_θ following Appendix A.4. However, as there exists potential numerical instability (depending on interpolants) in $\dot{\sigma}_t$, α_t^{-1} and λ_t , it's recommended to learn \mathbf{v}_θ in sampling with deterministic sampler instead of \mathbf{s}_θ . For the stochastic sampler, it's required to have both \mathbf{v}_θ and \mathbf{s}_θ in integration, so we always need to convert from one (either learning velocity or score) to obtain the other. Under this scenario, the numerical issue from Appendix A.4 can only be avoided by clipping the time interval near $t = 0$. Empirically we found clipping the interval by $h = 0.04$ and doing a long last step from $t = 0.04$ to 0 can greatly benefit the performance. A detailed summary of sampler configuration is provided in Appendix E.

Additionally, we could replace \mathbf{v}_θ and \mathbf{s}_θ by \mathbf{v}_θ^ζ and \mathbf{s}_θ^ζ presented in Appendix C as inputs of the two samplers and enjoy the performance improvements coming along with guidance. As guidance requires evaluating both conditional and unconditional model output in a single step, it will impose twice the computational cost when sampling.

We also note that our models are compatible with more advanced samplers [37, 46]. We do not include the evaluations of those samplers in our work for the sake of direct comparison with the DDPM model, and we leave the investigation of potential performance improvements to future work.

Comparison between DDPM and Euler-Maruyama We primarily investigate and report the performance comparison between DDPM and Euler-Maruyama samplers. We set our Euler sampler's number of steps to be 250 to match that of DDPM during evaluation. This comparison is made direct and fair, as the DDPM method can be viewed as a discretized version of Euler's method.

Comparison between DDIM and Heun We also investigate the performance difference produced by deterministic samplers between DiT and our models. In Fig. 6, we show the FID-50K results for both DiT models sampled with

DDIM and SiT models sampled with Heun. We note that this is not directly an apples-to-apples comparison, as DDIM can be viewed as a discretized version of the first order Euler’s method, while we use the second order Heun’s method in sampling SiT models, due to the large discretization error with Euler’s method in continuous time. Nevertheless, we control the NFEs for both DDIM (250 sampling steps) and Heun (250 NFE).

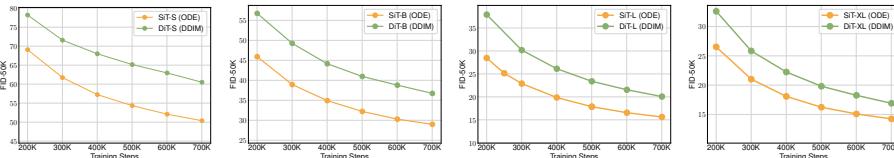


Fig. 6: SiT observes improvement in FID across all model sizes. We show FID-50K over training iterations for both DiT and SiT models. Across all model sizes, SiT converges faster. We acknowledge this is *not directly an apples-to-apples comparison*. This is because DDIM is essentially a discrete form of the first-order Euler’s method, whereas in sampling SiT, we employ the second-order Heun’s method. Nevertheless, both the SiT and DiT results are produced by a deterministic sampler with a 250 NFE.

Algorithm 1 Deterministic Heun Sampler

```

procedure HEUNSAMPLER( $\mathbf{v}_\theta(\mathbf{x}, t, \mathbf{y})$ ,  $t_{i \in \{0, \dots, N\}}$ ,  $\alpha_t$ ,  $\sigma_t$ )
    sample  $\mathbf{x}_0 \sim \mathcal{N}(0, \mathbf{I})$             $\triangleright$  Generate initial sample
     $\Delta t \leftarrow t_1 - t_0$                    $\triangleright$  Determine fixed step size
    for  $i \in \{0, \dots, N - 1\}$  do
         $\mathbf{d}_i \leftarrow \mathbf{v}_\theta(\mathbf{x}_i, t_i, \mathbf{y})$ 
         $\tilde{\mathbf{x}}_{i+1} \leftarrow \mathbf{x}_i + \Delta t \mathbf{d}_i$            $\triangleright$  Euler Step at  $t_i$ 
         $\mathbf{d}_{i+1} \leftarrow \mathbf{v}_\theta(\tilde{\mathbf{x}}_{i+1}, t_{i+1}, \mathbf{y})$ 
         $\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + \frac{\Delta t}{2} [\mathbf{d}_i + \mathbf{d}_{i+1}]$   $\triangleright$  Explicit trapezoidal
        rule at  $t_{i+1}$ 
    end for
    return  $\mathbf{x}_N$ 
end procedure

```

E Additional Implementation Details

We implemented our models in JAX following the DiT PyTorch codebase by [50]⁴, and referred to [2]⁵, [64]⁶, and [20]⁷ for our implementation of the Euler-Maruyama sampler. For the Heun sampler, we directly used the one from `diffra` [34]⁸, a JAX-based numerical differential equation solver library.

⁴ <https://github.com/facebookresearch/DiT>

⁵ <https://github.com/malbergo/stochastic-interpolants>

⁶ https://github.com/yang-song/score_sde

⁷ <https://github.com/nv-tlabs/CLD-SGM>

⁸ <https://github.com/patrick-kidger/diffra>

Algorithm 2 Stochastic Euler-Maruyama Sampler

```

procedure EULERSAM-
    PLER( $\mathbf{v}_\theta(\mathbf{x}, t, \mathbf{y})$ ,  $w_t, t_i \in \{0, \dots, N\}$ ,  $T, \alpha_t, \sigma_t$ )
        sample  $\mathbf{x}_0 \sim \mathcal{N}(0, \mathbf{I})$   $\triangleright$  Generate initial sample
         $\mathbf{s}_\theta \leftarrow \text{convert}$  from  $\mathbf{v}_\theta$  following Appendix A.4
         $\triangleright$  Obtain  $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$  in Eq. (4)
         $\Delta t \leftarrow t_1 - t_0$   $\triangleright$  Determine fixed step size
        for  $i \in \{0, \dots, N-1\}$  do
            sample  $\boldsymbol{\varepsilon}_i \sim \mathcal{N}(0, \mathbf{I})$ 
             $d\boldsymbol{\varepsilon}_i \leftarrow \boldsymbol{\varepsilon}_i * \sqrt{\Delta t}$ 
             $\mathbf{d}_i \leftarrow \mathbf{v}_\theta(\mathbf{x}_i, t_i, \mathbf{y}) + \frac{1}{2} w_{t_i} \mathbf{s}_\theta(\mathbf{x}_i, t_i, \mathbf{y})$   $\triangleright$  Evaluate drift term at  $t_i$ 
             $\bar{\mathbf{x}}_{i+1} \leftarrow \mathbf{x}_i + \Delta t \mathbf{d}_i$ 
             $\mathbf{x}_{i+1} \leftarrow \bar{\mathbf{x}}_{i+1} + \sqrt{w_{t_i}} d\boldsymbol{\varepsilon}_i$   $\triangleright$  Evaluate diffusion term at  $t_i$ 
        end for
         $h \leftarrow T - t_N$   $\triangleright$  Last step size;  $T$  denotes the time where  $\mathbf{x}_T = \mathbf{x}_*$ 
         $\mathbf{d} \leftarrow \mathbf{v}_\theta(\mathbf{x}_N, t_N, \mathbf{y}) + \frac{1}{2} w_{t_N} \mathbf{s}_\theta(\mathbf{x}_N, t_N, \mathbf{y})$ 
         $\mathbf{x} \leftarrow \mathbf{x}_N + h * \mathbf{d}$   $\triangleright$  Last step; output noiseless sample without diffusion
        return  $\mathbf{x}$ 
end procedure

```

Architectural Configurations We follow the identical transformer architectures in DiT and have four different configurations: SiT- $\{\text{S,B,L,XL}\}$, varying in model size (parameters) and compute (flops). A detailed summarization is presented below.

Table 9: Details of SiT models. We follow DiT [50] for the Small (S), Base (B), Large (L) and XLarge (XL) model configurations.

Model	Layers N	Hidden size d	Heads
SiT-S	12	384	6
SiT-B	12	768	12
SiT-L	24	1024	16
SiT-XL	28	1152	16

Training configurations We trained all of our models following identical structure and hyperparameters retained from DiT [50]. We used AdamW [35,45] as optimizer for all models. We use a constant learning rate of 1×10^{-4} and a batch size of 256. We used random horizontal flip with probability of 0.5 in data augmentation. *We did not tune the learning rates, decay/warm up schedules, AdamW parameters, nor use any extra data augmentation or gradient clipping during training.* Our

largest model, SiT-XL, trains at approximately 6.8 iters/sec on a TPU v4-64 pod following the above configurations. This speed is slightly faster compared to DiT-XL, which trains at 6.4 iters/sec under identical settings.

Sampling configurations We maintain an exponential moving average (EMA) of all models weights over training with a decay of 0.9999. All results are sampled from the EMA checkpoints, which is empirically observed to yield better performance. We summarize the start and end points of our deterministic and stochastic samplers with different interpolants below, where each t_0 and t_N are carefully tuned to optimize performance and avoid numerical instability during integration.

Table 10: Sampler configurations

<i>Interpolant</i>	<i>Model</i>	<i>Objective</i>	Heun		Euler-Maruyama	
			t_0	t_N	t_0	t_N
SBDM-VP	velocity	\mathcal{L}_v	1	1e-5	1	4e-2
		\mathcal{L}_{s_λ}	1	1e-5	1	4e-2
GVP	velocity	\mathcal{L}_v	1	0	1	4e-2
		\mathcal{L}_s	1 - 1e-5	0 1 - 1e-3	4e-2	
LIN	velocity	\mathcal{L}_v	1	0	1	4e-2
		\mathcal{L}_s	1 - 1e-5	0 1 - 1e-3	4e-2	

FID calculation We calculate FID scores between generated images (10K or 50K) and all available real images in ImageNet training dataset. We observe small performance variations between TPU-based FID evaluation and GPU-based FID evaluation (ADM’s TensorFlow evaluation suite [19]⁹). To ensure consistency with the baseline DiT, we sample all of our models on GPU and obtain FID scores using the ADM evaluation suite.

⁹ <https://github.com/openai/guided-diffusion/tree/main/evaluations>

F Additional Visual results



Fig. 7: Uncurated 512×512 SiT-XL samples.

Classifier-free guidance scale = 4.0
Class label = "volcano"(980)



Fig. 8: Uncurated 512×512 SiT-XL samples.

Classifier-free guidance scale = 4.0
Class label = "arctic fox"(279)



Fig. 9: Uncurated 512×512 SiT-XL samples.

Classifier-free guidance scale = 4.0
Class label = "loggerhead turtle"(33)



Fig. 10: Uncurated 512×512 SiT-XL samples.

Classifier-free guidance scale = 4.0
Class label = "balloon"(417)



Fig. 11: Uncurated 512×512 SiT-XL samples.

Classifier-free guidance scale = 4.0
Class label = "red panda"(387)



Fig. 12: Uncurated 512×512 SiT-XL samples.

Classifier-free guidance scale = 4.0
Class label = "geyser"(974)



Fig. 13: Uncurated 256×256 SiT-XL samples.

Classifier-free guidance scale = 4.0
Class label = "macaw"(88)



Fig. 14: Uncurated 256×256 SiT-XL samples.

Classifier-free guidance scale = 4.0
Class label = "golden retriever"(207)



Fig. 15: Uncurated 256×256 SiT-XL samples.

Classifier-free guidance scale = 4.0
Class label = "ice cream"(928)

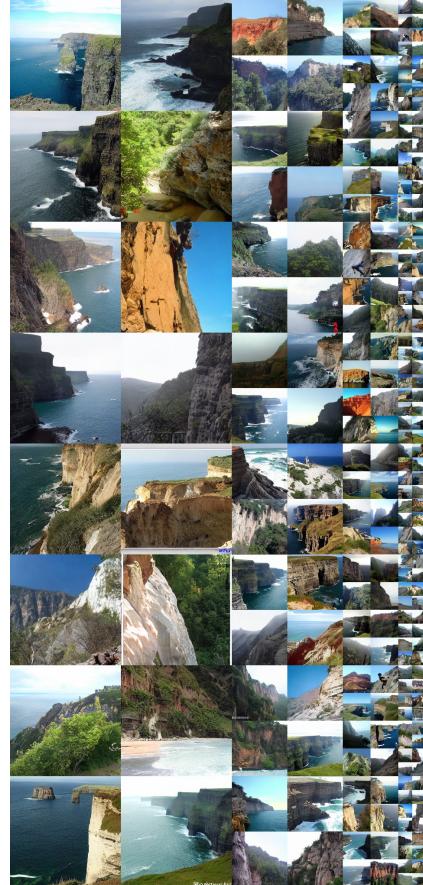


Fig. 16: Uncurated 256×256 SiT-XL samples.

Classifier-free guidance scale = 4.0
Class label = "cliff"(972)



Fig. 17: Uncurated 256×256 SiT-XL samples.

Classifier-free guidance scale = 4.0
Class label = "husky"(250)



Fig. 18: Uncurated 256×256 SiT-XL samples.

Classifier-free guidance scale = 4.0
Class label = "valley"(979)