GENERALIZATION THROUGH VARIANCE: HOW NOISE SHAPES INDUCTIVE BIASES IN DIFFUSION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

How diffusion models generalize beyond their training set is not known, and is somewhat mysterious given two facts: the optimum of the denoising score matching (DSM) objective usually used to train diffusion models is the score function of the training distribution; and the networks usually used to learn the score function are expressive enough to learn this score to high accuracy. We claim that a certain feature of the DSM objective—the fact that its target is not the training distribution's score, but a noisy quantity only equal to it in expectation—strongly impacts whether and to what extent diffusion models generalize. In this paper, we develop a mathematical theory that partly explains this 'generalization through variance' phenomenon. Our theoretical analysis exploits a physics-inspired path integral approach to compute the distributions typically learned by a few paradigmatic under- and overparameterized diffusion models. We find that the distributions diffusion models effectively learn to sample from resemble their training distributions, but with 'gaps' filled in, and that this inductive bias is due to the covariance structure of the noisy target used during training. We also characterize how this inductive bias interacts with feature-related inductive biases.

1 INTRODUCTION

Diffusion models (Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Ho et al., 2020; Yang et al., 2023) have proven effective at producing high-quality samples (e.g., images) *like* those from some training distribution, but not overwhelmingly so. This ability to generalize is somewhat surprising for two reasons. First, the optimum of the denoising score matching (DSM) objective usually used to train them is the score function of the training distribution (Vincent, 2011; Song & Ermon, 2019), and sampling using this score only reproduces training examples (see Appendix A). Second, the network architectures usually used for score function approximation are highly expressive. Two near-SOTA models developed by Karras et al. (2022) have $\sim$ 56 million (CIFAR-10, trained on 200 million samples) and $\sim$ 296 million parameters (ImageNet-64, trained on 2500 million samples), respectively. Sufficiently expressive models can fit even random noise (Zhang et al., 2017).

A body of empirical work bears on the question of when and to what extent diffusion models generalize. Training data is more likely to be memorized when training sets are small (Somepalli et al., 2023a; Stein et al., 2023; Dar et al., 2024; Kadkhodaie et al., 2024), contain duplicates (Somepalli et al., 2023a; Carlini et al., 2023; Somepalli et al., 2023b), or feature low 'complexity' (Somepalli et al., 2023b; Stein et al., 2023). The specific training examples more likely to be memorized are either highly duplicated or outliers (Carlini et al., 2023). Whether generalization happens also strongly depends on model capacity, with Yoon et al. (2023) and Zhang et al. (2024) observing a sharp transition from memorization to generalization as the number of training examples used somewhat outstrips model capacity. However, the relationship between model performance (e.g., FID score) and model size, given a fixed number of training examples, is not monotonic; Karras et al. (2024) observe that their ImageNet models strictly improve (and hence generalize better) as model size increases.

At present, there is arguably no theory that describes when diffusion models generalize and characterizes how the associated inductive biases depend on details like training set structure, the choice of forward/reverse processes, and model architecture. Most existing theoretical work focuses on orthogonal questions: given a *known ground truth*, can one mathematically guarantee that in some limit (e.g., a large or infinite number of samples from the ground truth distribution) diffusion models recover the ground truth, and bound how score approximation error impacts agreement (Bortoli, 2022; Chen et al., 2023a;c; Han et al., 2024)? The question we are interested in is qualitatively different: given $M \geq 1$ examples from a data distribution p_{data}, how do samples from a model trained on those examples differ from them? For example, does the model effectively interpolate training data? If so, when, and what details does this depend on?

In this paper, we argue that six factors substantially impact how diffusion models generalize.

1. **Noisy objective.** The target of the DSM objective is not the score of the training distribution, but a noisy quantity *only equal to it in expectation*. This quantity, which we call the 'proxy score', introduces additional randomness to training, and has extremely high variance at low noise levels (infinite variance, in fact, at zero noise). Intuitively, this makes score function estimates, especially at low noise levels, inaccurate (this is well-known; Karras et al. (2022) remark on this when they discuss their choice of loss weighting). Moreover, this variance is not uniform in state space, but higher in 'boundary regions', e.g., regions of state space close to multiple training examples. This provides a useful inductive bias.
2. **Forward process.** Details of the forward process (e.g., when noise is added, asymmetry in how noise is added along different directions of state space) affect generalization through their influence on the covariance structure of the proxy score.
3. **Nonlinear score-dependence.** The learned distribution depends nonlinearly on the learned score function through the dynamics of the reverse process. This implies that the average learned distribution is *not* the training distribution, even if the score estimator is unbiased.
4. **Model capacity.** Models generalize better when they are somewhat underparameterized.
5. **Model features.** Feature-related inductive biases interact with, and can enhance, inductive biases due to the covariance structure of the proxy score.
6. **Training set structure.** Nontrivial generalization (e.g., interpolation) is substantially more likely when a large number of training examples are near each other in state space; outliers are less likely to be meaningfully generalized.

Hence, details of training (1, 2), sampling (3), model architecture (4, 5), and the training set (6) all interact to determine the details of generalization. Other aspects, like learning dynamics, also almost certainly play a role, but we mostly neglect them here. The first factor is particularly important, and without it we will see that diffusion models do not generalize well; for this reason, we refer to the phenomenon enabled by (1) and affected by (2-6) as **generalization through variance**.

We support this claim using physics-inspired theory. The Martin-Siggia-Rose (MSR) path integral description of stochastic dynamics (Martin et al., 1973), which has also been exploited to characterize random neural networks (Crisanti & Sompolinsky, 2018) and learning dynamics (Mignacco et al., 2020; Bordelon & Pehlevan, 2022; 2023), plays a pivotal role in our analysis. First, we use the MSR path integral to derive the generic form of 'generalization through variance', and then we discuss in specific, analytically tractable cases of interest (e.g., linear models, lazy infinite-width neural networks) how the details change and the role of each of the aforementioned factors. To keep our theoretical analysis tractable, we focus on unconditional, non-latent models.

2 Preliminaries

Data distribution. Let $p_{data}(\boldsymbol{x}_0)$ denote a data distribution on $\mathbb{R}^D$. We are especially interested in the case that p_{data} consists of a discrete set of $1 \leq M < \infty$ examples (e.g., images), so that $p_{data}(\boldsymbol{x}_0) = \sum_{m=1}^{M} \delta(\boldsymbol{x}_0 - \boldsymbol{\mu}_m)/M$, where δ is the Dirac delta function. However, we do not restrict ourselves to this case.

Forward/reverse diffusion. Training a diffusion model involves learning to convert samples from some other distribution $p_{noise}(\boldsymbol{x}_T)$ (e.g., a normal distribution) to samples from $p_{data}(\boldsymbol{x}_0)$ via

$$\dot{\boldsymbol{x}}_t = -\beta_t \boldsymbol{x}_t + \boldsymbol{G}_t \boldsymbol{\eta}_t \qquad t = 0 \to t = T \qquad \text{forward process, } p_{data} \text{ to } p_{noise} \tag{1}$$

$$\dot{\boldsymbol{x}}_t = -\beta_t \boldsymbol{x}_t - \boldsymbol{D}_t \boldsymbol{s}(\boldsymbol{x}_t, t) \qquad t = T \to t = \epsilon \qquad \text{reverse process, } p_{noise} \text{ to } p_{data} \tag{2}$$

where $\boldsymbol{\eta}_t \in \mathbb{R}^K$ is Gaussian white noise, $\boldsymbol{G}_t \in \mathbb{R}^{D \times K}$ is a nonnegative matrix that controls the noise amplitude, $\boldsymbol{D}_t := \boldsymbol{G}_t \boldsymbol{G}_t^T/2$ is the corresponding diffusion tensor, $\beta_t \geq 0$ controls decay to the

Table 1: Popular forward processes in our parameterization. For these, $\boldsymbol{G}_t := g_t \boldsymbol{I}_D$ and $\boldsymbol{S}_t = \sigma_t^2 \boldsymbol{I}_D$.

	β_t	g_t	α_t	σ_t	end time
VP-SDE	$\beta_{min} + \beta_d t$	$\sqrt{2\beta_t}$	$e^{-\int_0^t \beta_{t'} \, dt'}$	$\sqrt{1 - e^{-2\int_0^t \beta_{t'} \, dt'}}$	1
EDM	0	$\sqrt{2t}$	1	t	T

origin, $\epsilon > 0$ is a parameter that helps ensure numerical stability, and $\boldsymbol{s}(\boldsymbol{x}, t) := \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}|t)$ is the score function. We allow $\boldsymbol{G}_t$ to be a matrix so we can study how asymmetries affect generalization later. The forward process' marginals are $p(\boldsymbol{x}|t) := \int p(\boldsymbol{x}|\boldsymbol{x}_0, t) p_{data}(\boldsymbol{x}_0) \, d\boldsymbol{x}_0$. The transition probabilities are $p(\boldsymbol{x}|\boldsymbol{x}_0, t) = \mathcal{N}(\boldsymbol{x}; \alpha_t \boldsymbol{x}_0, \boldsymbol{S}_t)$, where $\alpha_t := e^{-\int_0^t \beta_{t'} \, dt'}$ and $\boldsymbol{S}_t := \int_0^t 2\boldsymbol{D}_{t'} \alpha_{t'}^2 \, dt'$.

The forward process assumed here is fairly general, and includes popular choices like the VP-SDE (Song et al., 2021) and EDM formulation (Karras et al., 2022) (Table 1). This choice of reverse process is called the probability flow ODE (PF-ODE), and has been shown to have both practical (Song et al., 2021) and theoretical (Chen et al., 2023b) advantages. Since $\boldsymbol{s}(\boldsymbol{x}, t)$ is required to run the reverse process but is a priori unknown, "training" a model means approximating $\boldsymbol{s}(\boldsymbol{x}, t)$.

Denoising score matching. One could in principle use a naive mean-squared-error objective

$$J_0(\boldsymbol{\theta}) := \mathbb{E}_{t,\boldsymbol{x}} \left\{ \frac{\lambda_t}{2} \| \hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}, t) - \boldsymbol{s}(\boldsymbol{x}, t) \|_2^2 \right\} = \int \frac{\lambda_t}{2} \| \hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}, t) - \boldsymbol{s}(\boldsymbol{x}, t) \|_2^2 \, p(\boldsymbol{x}|t) p(t) \, d\boldsymbol{x} dt \quad (3)$$

to learn a parameterized score estimator $\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}, t)$. Here, $\lambda_t > 0$ is a positive weighting function and $p(t)$ is a time-sampling distribution. The DSM objective (Vincent, 2011; Song & Ermon, 2019)

$$J_1(\boldsymbol{\theta}) := \mathbb{E}_{t,\boldsymbol{x}_0,\boldsymbol{x}} \left\{ \frac{\lambda_t}{2} \| \hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}, t) - \tilde{\boldsymbol{s}}(\boldsymbol{x}, t; \boldsymbol{x}_0) \|_2^2 \right\} = \int \frac{\lambda_t}{2} \, \| \hat{\boldsymbol{s}}_{\boldsymbol{\theta}} - \tilde{\boldsymbol{s}} \|_2^2 \, p(\boldsymbol{x}, \boldsymbol{x}_0, t) \, d\boldsymbol{x} d\boldsymbol{x}_0 dt \quad (4)$$

where $p(\boldsymbol{x}, \boldsymbol{x}_0, t) := p(\boldsymbol{x}|\boldsymbol{x}_0, t) p_{data}(\boldsymbol{x}_0) p(t)$, is usually used instead. While the folklore justifying this choice is that the score function is not known, this is not true; both J_0 and J_1 are optimized when $\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}$ equals the score of the training distribution (see Appendix A), which is known.

We will argue that the real difference between J_0 and J_1 is that J_1 generalizes better, and that this is in part because the **proxy score** $\tilde{\boldsymbol{s}}(\boldsymbol{x}, t; \boldsymbol{x}_0) := \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}|\boldsymbol{x}_0, t) = \boldsymbol{S}_t^{-1}(\alpha_t \boldsymbol{x}_0 - \boldsymbol{x})$ is used as the target instead of the true score. It is a 'noisy' version of the true score (see Appendix B), since

$$\mathbb{E}_{\boldsymbol{x}_0|\boldsymbol{x},t}[\tilde{\boldsymbol{s}}(\boldsymbol{x}, t; \boldsymbol{x}_0)] = \boldsymbol{s}(\boldsymbol{x}, t) \qquad \text{Cov}_{\boldsymbol{x}_0|\boldsymbol{x},t}[\tilde{s}_i, \tilde{s}_j] = S_{t,ij}^{-1} + \partial_{ij}^2 \log p(\boldsymbol{x}|t) \; . \quad (5)$$

Although the proxy score is equal to the score of the training distribution in expectation, neural networks trained on J_1 empirically learn a different distribution and generalize better. We claim that this fact is closely related to the *covariance structure* of the proxy score. Two relevant observations about its form are as follows. First, it is large at small times, since $\boldsymbol{S}_t \to 0$ as $t \to 0$. Second, it is large where the log-likelihood $\log p(\boldsymbol{x}|t)$ has substantial curvature. In the typical case, where p_{data} consists of a discrete set of M examples, regions of high curvature precisely correspond to the location of training examples and the boundaries between them (Fig. 1).

In what follows, we assume training involves $P \gg 1$ independent samples from $p(\boldsymbol{x}, \boldsymbol{x}_0, t)$ (note that P is different than M, the number of points in discrete p_{data}).

Generalization and inductive biases. In a typical supervised learning setting, one trains a model on one set of data and tests it on another, and defines 'generalization error' as performance on the held-out data. Here, we are interested in a different type of problem: *given* a model trained on samples from $p(\boldsymbol{x}, \boldsymbol{x}_0, t)$, *to what extent* does the learned distribution differ from p_{data}, and what are the associated inductive biases? Of particular interest in whether models do three things: (i) interpolation (filling in gaps in the training data), (ii) extrapolation (extending patterns in the training data), and (iii) feature blending (generating samples which include both feature X and feature Y even when training examples only involve one of the two features).

In our setting, a subtle but important point is that there is generally no ground truth. For example, the smooth distribution that CIFAR-10 or MNIST images are drawn from does not exist, except in a 'Platonic' sense; we are interested in the extent to which diffusion models learn a distribution plausibly *like* a smoothed version of the training distribution.

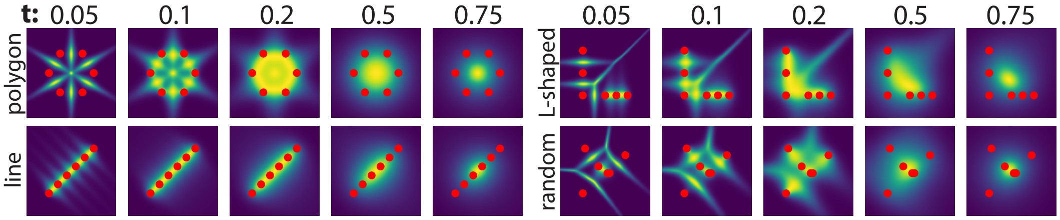

Figure 1: Visualization of proxy score variance for four example 2D data distributions. Each distribution is supported on six point masses (red dots). Note that as t changes (left: small t, right: large t), boundary regions at different scales are emphasized.

3 Approach: Computing typical learned distributions

The distribution $q(\boldsymbol{x}_0|\boldsymbol{\theta})$ learned by a diffusion model depends on the learned score $\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}$ nonlinearly through PF-ODE dynamics; importantly, we are less interested in how well the score is estimated, and more interested in how estimation errors impact q. The learned score can be viewed as a random variable, since it depends on the P samples $\boldsymbol{x}^{(i)}, \boldsymbol{x}_0^{(i)}, t^{(i)} \sim p(\boldsymbol{x}, \boldsymbol{x}_0, t)$ used during training. In order to theoretically understand how diffusion models generalize, we aim to obtain an analytic expression for the 'typical' learned distribution by averaging q over sample realizations.

How do we do the required averaging? One of our major contributions is to introduce a theoretical approach for averaging $q(\boldsymbol{x}_0)$ over variation due to $\hat{\boldsymbol{s}}$. Below, we describe our approach.

Writing PF-ODE dynamics in terms of a path integral. How does one average over the result of an ODE given that, in the case of PF-ODE dynamics, there is generally no closed-form expression for the result? To address this issue, we use a novel **stochastic path integral** representation of PF-ODE dynamics that makes the required average easy to do. If $q(\boldsymbol{x}_0|\boldsymbol{x}_T;\boldsymbol{\theta})$ denotes the distribution of PF-ODE outputs given a score estimator $\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}, t)$ and a fixed noise seed $\boldsymbol{x}_T$,

$$q(\boldsymbol{x}_0|\boldsymbol{x}_T;\boldsymbol{\theta}) = \int \mathcal{D}[\boldsymbol{p}_t]\mathcal{D}[\boldsymbol{x}_t] \ \exp\left\{ \int_\epsilon^T i\boldsymbol{p}_t \cdot [\dot{\boldsymbol{x}}_t + \beta_t \boldsymbol{x}_t + \boldsymbol{D}_t \hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)] \ dt \right\} \tag{6}$$

where the integral is over all possible paths from $\boldsymbol{x}_T$ to $\boldsymbol{x}_0$. (To avoid technical issues, we assume a particular time discretization in all calculations. See Appendix C.) This type of path integral is a time-reversed version of the Martin-Siggia-Rose (MSR) path integral (Martin et al., 1973).

Averaging over possible sample realizations. Because the argument of the exponential depends linearly on the score, the required ensemble average is now easy to do. Using $[\cdots]$ to denote it,

$$[q(\boldsymbol{x}_0|\boldsymbol{x}_T)] = \int \mathcal{D}[\boldsymbol{p}_t]\mathcal{D}[\boldsymbol{x}_t] \exp\left\{ M_1 - \frac{1}{2}M_2 + \cdots \right\} \tag{7}$$

$$M_1 := \int_\epsilon^T i\boldsymbol{p}_t \cdot [\dot{\boldsymbol{x}}_t + \beta_t \boldsymbol{x}_t + \boldsymbol{D}_t \boldsymbol{s}_{avg}(\boldsymbol{x}_t, t)] \, dt \qquad M_2 := \int_\epsilon^T \int_\epsilon^T \boldsymbol{p}_t^T \boldsymbol{V}(\boldsymbol{x}_t, t; \boldsymbol{x}_{t'}, t') \boldsymbol{p}_{t'} \ dt dt'$$

where $\boldsymbol{s}_{avg}(\boldsymbol{x}_t, t) := [\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)]$ is the ensemble's average score estimator, and $\boldsymbol{V}(\boldsymbol{x}_t, t; \boldsymbol{x}_{t'}, t') := \boldsymbol{D}_t \text{Cov}_{\boldsymbol{\theta}}[\hat{\boldsymbol{s}}(\boldsymbol{x}_t, t), \hat{\boldsymbol{s}}(\boldsymbol{x}_{t'}, t')]\boldsymbol{D}_{t'}$ measures ensemble variance. Assuming higher-order terms can be neglected—and hence that the estimator distribution is approximately Gaussian—one can show (see Appendix C) that sampling from $[q(\boldsymbol{x}_0|\boldsymbol{x}_T)]$ is equivalent to integrating an (Ito-interpreted) SDE:

Proposition 3.1 (Effective SDE description of typical learned distribution)**.** *Sampling from* $[q(\boldsymbol{x}_0|\boldsymbol{x}_T)]$ *is equivalent to integrating the (Ito-interpreted) SDE*

$$\dot{\boldsymbol{x}}_t = -\beta_t \boldsymbol{x}_t - \boldsymbol{D}_t \boldsymbol{s}_{avg}(\boldsymbol{x}_t, t) + \boldsymbol{\xi}(\boldsymbol{x}_t, t) \qquad\qquad t = T \to t = \epsilon \tag{8}$$

with initial condition $\boldsymbol{x}_T$*, where* $\boldsymbol{s}_{avg}(\boldsymbol{x}_t, t) := [\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)]$ *and where the noise term* $\boldsymbol{\xi}(\boldsymbol{x}_t, t)$ *has mean zero and autocorrelation* $\boldsymbol{V}(\boldsymbol{x}_t, t; \boldsymbol{x}_{t'}, t') := \boldsymbol{D}_t \mathit{Cov}_{\boldsymbol{\theta}}[\hat{\boldsymbol{s}}(\boldsymbol{x}_t, t), \hat{\boldsymbol{s}}(\boldsymbol{x}_{t'}, t')]\boldsymbol{D}_{t'}$*.*

If $\hat{s}$ is unbiased and M is finite, then the noise term is solely responsible for the difference between true PF-ODE dynamics (which reproduces training examples) and a model's 'typical' sampling

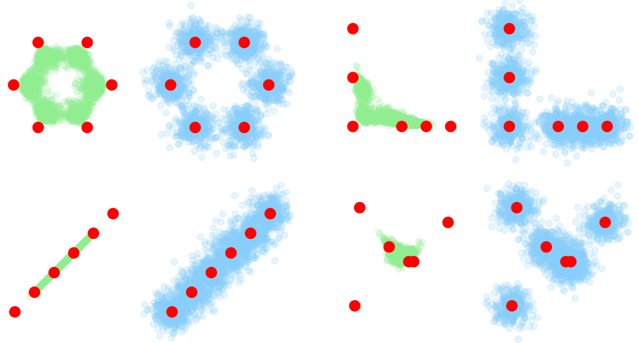

Figure 2: Samples from naive score model (green, EDM) compared to comparable kernel density estimate model (blue) for the four distributions depicted in Fig. 1. The naive score model does not add probability mass uniformly about training data (red), but adds more mass to boundary regions.

dynamics—i.e., generalization occurs if and only if $\boldsymbol{V} \neq \boldsymbol{0}$. This makes characterizing $\boldsymbol{V}$, which we call the *V-kernel* since it reflects ensemble variance, crucially important for understanding how diffusion models generalize. Our remaining theoretical work is to complete two tasks: first, to compute $\boldsymbol{s}_{avg}$ and $\boldsymbol{V}$ for a few paradigmatic and theoretically tractable architectures; and second, to study how their precise forms affect $[q(\boldsymbol{x}_0)]$.

4 Diffusion models that memorize training data still generalize

It is instructive to first consider an extreme case: do diffusion models generalize in the complete *absence* of any model-related inductive biases? Perhaps surprisingly, the answer is yes. In this section, we make this point using a toy model in which training and sampling are interleaved.

Suppose that the PF-ODE is integrated backward in time from an initial point $\boldsymbol{x}_T$ (e.g., using first-order Euler updates). At each time step, suppose one samples R times from $p(\boldsymbol{x}_0|\boldsymbol{x}_t, t)$ and constructs the 'naive' score estimator $\hat{\boldsymbol{s}}(\boldsymbol{x}_t, t) := \sum_{r=1}^{R} \tilde{\boldsymbol{s}}(\boldsymbol{x}_t, t; \boldsymbol{x}_{0t}^{(r)})/R$. Then suppose this estimator is used as the score in that step's PF-ODE update. Assume this process continues (with R new samples drawn at each time step) until $t = \epsilon$. Despite this approach using the proxy score directly (so that training data is 'memorized'), one obtains a nontrivial V-kernel, and hence generalization:

Proposition 4.1 (Naive score estimator generalizes)**.** *Consider the result of integrating the PF-ODE (Eq. 2) from $t = T$ to $t = \epsilon$ using $R \geq 1$ independent proxy score samples at each time step, i.e.,*

$$\dot{\boldsymbol{x}}_t = -\beta_t \boldsymbol{x}_t - \boldsymbol{D}_t \left(\sum_{r=1}^{R} \frac{\tilde{\boldsymbol{s}}(\boldsymbol{x}_t, t; \boldsymbol{x}_{0t}^{(r)})}{R} \right) \qquad \boldsymbol{x}_{0t}^{(r)} \sim p(\boldsymbol{x}_0|\boldsymbol{x}_t, t) := \frac{p(\boldsymbol{x}_t|\boldsymbol{x}_0, t)\, p_{data}(\boldsymbol{x}_0)}{p(\boldsymbol{x}_t|t)} .$$

Then $[q(\boldsymbol{x}_0|\boldsymbol{x}_T)]$ is described by an effective SDE (Eq. 8) with $\boldsymbol{s}_{avg} = \boldsymbol{s}$ and V-kernel

$$V_{ij}(\boldsymbol{x}_t, t; \boldsymbol{x}_{t'}, t') := \frac{1}{R} \sum_{a,b} D_{t,ia} \left[S_{t,ab}^{-1} + \partial_{ab}^2 \log p(\boldsymbol{x}_t|t) \right] D_{t,bj}\, \delta(t - t') . \tag{9}$$

See Appendix D for details and Fig. 2 for illustrative examples. Notably, the effective SDE is noisier when the covariance of the proxy score is high, e.g., in boundary regions between training examples. We will see in the next section that this is also true for less trivial models, but that the proxy score's covariance interacts with feature-related biases in order to determine the SDE's overall noise term.

5 Feature-related inductive biases enhance generalization

Model architecture is known to produce certain inductive biases, with spectral bias being a well-known example (Rahaman et al., 2019; Bordelon et al., 2020; Canatar et al., 2021). How do model-feature-related inductive biases affect the V-kernel? We answer this question below in two interesting but tractable cases: linear models, and (lazy regime) infinite-width neural networks. To ease notation, let $\boldsymbol{z} := (\boldsymbol{x}, t)$ and $C_{ij}(\boldsymbol{z}) := \mathrm{Cov}_{\boldsymbol{x}_0|\boldsymbol{x},t}[\tilde{s}_i(\boldsymbol{x}, t; \boldsymbol{x}_0), \tilde{s}_j(\boldsymbol{x}, t; \boldsymbol{x}_0)]$.

5.1 The V-kernel of expressive linear models

Consider a linear score estimator

$$\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}, t) = \boldsymbol{w}_0 + \boldsymbol{W}\boldsymbol{\phi}(\boldsymbol{x}, t) \ , \tag{10}$$

where the F feature maps $\boldsymbol{\phi} := (\phi_1, ..., \phi_F)^T$ are linearly independent, smooth functions from $\mathbb{R}^D \times (0, T]$ to $\mathbb{R}$ that are square-integrable with respect to the measure $\lambda_t p(\boldsymbol{x}, t)$. The parameters to be estimated are $\boldsymbol{\theta} := \{\boldsymbol{w}_0, \boldsymbol{W}\}$, with $\boldsymbol{w}_0 \in \mathbb{R}^D$ and $\boldsymbol{W} \in \mathbb{R}^{D \times F}$. Note that this estimator is linear in its features, but not necessarily in $\boldsymbol{x}$ or t. The weights that optimize Eq. 4 are (see Appendix E)

$$\boldsymbol{W}^* = -\boldsymbol{J}^T \boldsymbol{\Sigma}_{\boldsymbol{\phi}}^{-1} \qquad \boldsymbol{w}_0^* = \boldsymbol{J}^T \boldsymbol{\Sigma}_{\boldsymbol{\phi}}^{-1} \langle \boldsymbol{\phi} \rangle + \langle \tilde{\boldsymbol{s}} \rangle \tag{11}$$

where we define $\langle \cdots \rangle := \mathbb{E}_{\boldsymbol{x}, \boldsymbol{x}_0, t}[\lambda_t \ \cdots]/\mathbb{E}_t[\lambda_t]$ and matrices

$$\boldsymbol{J} := -\langle [\boldsymbol{\phi}(\boldsymbol{x}, t) - \langle \boldsymbol{\phi} \rangle] \, [\tilde{\boldsymbol{s}}(\boldsymbol{x}, t; \boldsymbol{x}_0) - \langle \tilde{\boldsymbol{s}} \rangle]^T \rangle \qquad \boldsymbol{\Sigma}_{\boldsymbol{\phi}} := \langle [\boldsymbol{\phi}(\boldsymbol{x}, t) - \langle \boldsymbol{\phi} \rangle] \, [\boldsymbol{\phi}(\boldsymbol{x}, t) - \langle \boldsymbol{\phi} \rangle]^T \rangle \ .$$

When averaged over $\boldsymbol{x}_0$ sample realizations, the estimator $\hat{\boldsymbol{s}}_*(\boldsymbol{x}, t) = \boldsymbol{w}_0^* + \boldsymbol{W}^* \boldsymbol{\phi}(\boldsymbol{x}, t)$ is unbiased as long as the set of feature maps is sufficiently expressive. Interestingly, this is true regardless of the $\boldsymbol{x}$ or t samples used, provided $F \leq P$. The following result characterizes $[q(\boldsymbol{x}_0)]$ for linear models:

Proposition 5.1 (Expressive linear models asymptotically generalize)**.** *Suppose the parameters of an expressive linear score estimator (Eq. 10) with F features are perfectly optimized according to the DSM objective (Eq. 4) using $P \geq F$ independent samples from $p(\boldsymbol{x}, \boldsymbol{x}_0, t)$, and define matrices*

$$\tilde{\boldsymbol{C}}_{ij} := \int \frac{\lambda_{t''}^2}{\mathbb{E}_t[\lambda_t]^2} [\boldsymbol{\phi}(\boldsymbol{z}'') - \langle \boldsymbol{\phi} \rangle] \, [\boldsymbol{\phi}(\boldsymbol{z}'') - \langle \boldsymbol{\phi} \rangle]^T C_{ij}(\boldsymbol{z}'') \, p(\boldsymbol{z}'') \, d\boldsymbol{z}'' \ . \tag{12}$$

Provided that the limit exists and is finite, in the $P \to \infty$ limit (where F may scale with P) we have

$$V_{ij}(\boldsymbol{z}; \boldsymbol{z}') = \lim_{P \to \infty} \frac{1}{P} \sum_{a,b} D_{t,ia} [\boldsymbol{\phi}(\boldsymbol{z}) - \langle \boldsymbol{\phi} \rangle]^T \boldsymbol{\Sigma}_{\boldsymbol{\phi}}^{-1} \tilde{\boldsymbol{C}}_{ab} \boldsymbol{\Sigma}_{\boldsymbol{\phi}}^{-1} [\boldsymbol{\phi}(\boldsymbol{z}') - \langle \boldsymbol{\phi} \rangle] \, D_{t',bj} \ . \tag{13}$$

On the other hand, if the number of features F does not scale with P, $\boldsymbol{V} \equiv \boldsymbol{0}$. See Appendix E for the details of our argument.

The V-kernel for linear models differs from the naive score's V-kernel (Eq. 9) via the presence of feature-related factors. In particular, the effective SDE is noisier *where features take atypical values*. One expects that these factors can either enhance or compete with noise due to the covariance structure (e.g., noise should be higher if features take atypical values in boundary regions).

5.2 The V-kernel of Lazy infinite-width neural networks

Neural networks in the neural tangent kernel (NTK) regime (Jacot et al., 2018; Bietti & Mairal, 2019) provide another interesting but tractable model. Such networks exhibit 'lazy' learning (Chizat et al., 2019) in the sense that weights do not move much from their initial values. Moreover, it is known that they interpolate training data in the absence of parameter regularization or early stopping (Bordelon et al., 2020). If they precisely interpolated their samples, we would expect to recover a V-kernel like the one we computed in Sec. 4; more generally, we expect something similar modified by the spectral inductive biases associated with the architecture (Canatar et al., 2021).

For simplicity, we consider fully-connected networks whose hidden layers all have width N, which is taken to infinity together with P (see Appendix F for details). The associated NTK has a Mercer decomposition with respect to the measure $\lambda_t p(\boldsymbol{x}, t)/\mathbb{E}[\lambda_t]$, so K can be written in terms of F orthonormal features $\{\phi_i\}$:

$$K(\boldsymbol{x}, t; \boldsymbol{x}', t') = \sum_k \lambda_k \phi_k(\boldsymbol{x}, t) \phi_k(\boldsymbol{x}', t') \qquad \int \frac{\lambda_t}{\mathbb{E}[\lambda_t]} \phi_k(\boldsymbol{x}, t) \phi_\ell(\boldsymbol{x}, t) \, p(\boldsymbol{x}, t) \, d\boldsymbol{x} dt = \delta_{k\ell} \ . \tag{14}$$

We assume training involves full-batch gradient descent on P samples from $p(\boldsymbol{x}, \boldsymbol{x}_0, t)$, so that the learned score function after training for an amount of 'time' τ has the closed-form solution

$$\hat{\boldsymbol{s}}(\boldsymbol{z}) = \hat{\boldsymbol{s}}_0(\boldsymbol{z}) + [\tilde{\boldsymbol{S}} - \hat{\boldsymbol{S}}_0]^T (\boldsymbol{I} - e^{-\boldsymbol{\Lambda}_T \boldsymbol{K} \tau / P}) \boldsymbol{K}^{-1} \boldsymbol{k}(\boldsymbol{z})$$

where $\hat{\boldsymbol{s}}_0$ is the network's initial output, $\tilde{\boldsymbol{S}} \in \mathbb{R}^{D \times P}$ contains proxy score samples, $\hat{\boldsymbol{S}}_0 \in \mathbb{R}^{D \times P}$ contains the network's initial outputs given the samples, $\boldsymbol{K} \in \mathbb{R}^{P \times P}$ is the kernel Gram matrix, $\boldsymbol{\Lambda}_T \in \mathbb{R}^{P \times P}$ is a diagonal matrix containing the weighting function $\lambda_t/\mathbb{E}[\lambda_t]$ evaluated on samples, and $\boldsymbol{k}(\boldsymbol{x}, t)$ is an input-dependent vector whose i-th component is $K(\boldsymbol{x}^{(i)}, t^{(i)}; \boldsymbol{x}, t)$. We have:

Proposition 5.2 (Lazy neural networks asymptotically generalize)**.** *Suppose the parameters of a fully-connected, infinite-width neural network characterized by a rank F NTK are optimized according to the DSM objective (Eq. 4) using P independent samples from $p(\boldsymbol{x}, \boldsymbol{x}_0, t)$, and define*

$$\tilde{\boldsymbol{C}}_{ij} := \int \frac{\lambda_{t''}^2}{\mathbb{E}[\lambda_t]^2} \boldsymbol{\phi}(\boldsymbol{z}'') \boldsymbol{\phi}(\boldsymbol{z}'')^T C_{ij}(\boldsymbol{z}'') \, p(\boldsymbol{z}'') \, d\boldsymbol{z}'' \; . \tag{15}$$

Provided that the limit exists and is finite, in the $P \to \infty$ limit (where F may scale with P) we have

$$V_{ij}(\boldsymbol{z}; \boldsymbol{z}') = \lim_{P \to \infty} \frac{1}{P} \sum_{a,b} D_{t,ia} \boldsymbol{\phi}(\boldsymbol{z})^T (\boldsymbol{I}_F - e^{-\boldsymbol{\Lambda}\tau}) \tilde{\boldsymbol{C}}_{ab} (\boldsymbol{I}_F - e^{-\boldsymbol{\Lambda}\tau}) \boldsymbol{\phi}(\boldsymbol{z}') D_{t',bj} \; . \tag{16}$$

In the infinite training time ($\tau \to \infty$) limit, we recover the Sec. 4 result with a prefactor $\kappa \propto F/P$:

$$V_{ij}(\boldsymbol{z}; \boldsymbol{z}') = \kappa \sum_{a,b} D_{t,ia} \boldsymbol{C}_{ab} D_{t,bj} \; \delta(\boldsymbol{z} - \boldsymbol{z}') \; . \tag{17}$$

See Appendix F for the full details of our argument. Interestingly, although the network is not assumed to be in the feature-learning regime, this result interpolates between our pure memorization (Prop. 4.1) and linear model (Prop. 5.1) results as we change the value of the training time τ.

6 Discussion

We used a novel path-integral approach to quantitatively characterize the ‘typical’ distribution learned by diffusion models, and find that generalization is influenced by a combination of factors related to training (the DSM objective and forward process; Sec. 2 and 4), sampling (the learned distribution depends nonlinearly on score estimates; Sec. 3), model architecture (Sec. 5), and the data distribution. Below, we use our theory to comment on various previous observations.

DSM produces noisy estimators, but stable distributions. Various forms of score ‘mislearning’ are well-known. At small times, scores are hard to learn due to the noisiness of the proxy score target, leading authors like Karras et al. (2022) to suggest a $p(t)$ that emphasizes intermediate noise scales. Chao et al. (2022) discuss how score estimation errors affect conditional scores. Xu et al. (2023) explicitly study the variance-near-mode-boundaries issue we discussed, and propose a strategy for mitigating it. On the other hand, it is well-known that despite noisy score estimates, diffusion models generally produce smooth output distributions (see, e.g., Luzi et al. (2024)). Moreover, two diffusion models trained on non-overlapping subsets of a data set are often highly similar (Kadkhodaie et al., 2024). These facts are due to noisy score estimates contributing to sample generation through the PF-ODE, which effectively ‘averages’ over estimator noise. Our theory is consistent with these observations: even the interleaved training-sampling procedure discussed in Sec. 4 produces a well-behaved, smooth distribution.

DSM produces a boundary-smearing inductive bias. This has been previously pointed out by authors like Xu et al. (2023). Where we differ from previous authors is in considering this issue a potential strength. Integrating the PF-ODE using the true score reproduces training examples, so it is in some sense beneficial to ‘mislearn’ the score. This particular kind of mislearning is useful for several ways of generalizing point clouds, including interpolation, extrapolation, and feature blending. Moreover, producing this inductive bias is an interesting way diffusion models differ from something like kernel density estimation: boundary regions *across different noise scales* are smeared out, with different scales linked via PF-ODE dynamics, which may provide better generalization than convolving the training distribution with any single kernel.

Architecture-related inductive biases play a role. As we showed in Sec. 5, feature/architecture-related inductive biases interact with DSM’s boundary-smearing bias in order to determine how diffusion models generalize. This appears to be consistent, for example, with the Kadkhodaie et al. (2024) finding that diffusion models effectively exhibit ‘adaptive geometric harmonic priors’; their finding is specifically in the context of score estimation using a convolutional neural network (CNN) architecture. It is plausible that this choice encourages a harmonic inductive bias, since CNNs more generally exhibit inductive biases related to spatial translation invariance.

Generalization through variance harmful and helpful. It is important to note that this kind of generalization is not always helpful. A trivial example is that unconditional models trained on MNIST digit images tend to learn to produce non-digits as output in the absence of label information (see, e.g., Bortoli et al. (2021)). More generally, blending modes may or may not be desirable, since it can produce (e.g.) images very qualitatively different from those of the training distribution.

Other forms of generalization are possible. Factors we did not study, like learning dynamics, most likely also partly determine how diffusion models generalize. For example, the use of stochastic gradient descent introduces additional randomness that disfavors converging on sharp local optima (Smith & Le, 2018; Smith et al., 2020). It would be interesting to utilize recent theoretical tools (Bordelon & Pehlevan, 2023) to characterize how learning dynamics impacts generalization, especially in the rich (Geiger et al., 2020; Woodworth et al., 2020) rather than lazy learning regime.

Comment on memorization. Determining whether diffusion models memorize data (Somepalli et al., 2023a; Carlini et al., 2023), and if so how to address the issue (Vyas et al., 2023), has become a significant technical and societal issue. Our theory suggests that since generalization through variance happens primarily in boundary regions, diffusion models are unlikely to substantially generalize outliers. Since conditional models involve distributions of much higher effective dimension, one may expect that more training examples are ‘outlier-like’, and hence memorization should happen more often; this is consistent with the observations of Somepalli et al. (2023b). Our theory also suggests why duplications increase memorization: the existence of a strong boundary between modes, which requires modes to have comparable probability mass, is degraded.

Limitations of theoretical approach. Our theory is simplified in at least two ways. First, only a simple formulation of training (via DSM) and sampling (via the PF-ODE) from diffusion models is considered. There exist alternatives to DSM, like sliced score matching (Song et al., 2020), and alternative ways of sampling, including using auxiliary momentum-like variables (Dockhorn et al., 2022b). Also, our theoretical analysis neglects variation due to numerical integration schemes, even though these may matter in practice (Liu et al., 2022; Karras et al., 2022; Dockhorn et al., 2022a).

Second, we study only unconditional models for simplicity. This means that in particular do not consider diffusion coupled to attention layers, which enables the text-conditioning behind many of the most striking diffusion-model-related successes (Rombach et al., 2022; Blattmann et al., 2023).

Finally, we do not consider realistic architectures (like U-nets) and rich learning dynamics due to theoretical tractability. However, these challenges are not unique to the current setting. Despite our contribution’s simplicity, we hope that it nonetheless provides a foundation for others to more rigorously understand the inductive biases and generalization capabilities of diffusion models.

REFERENCES

Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/dbc4d84bfcfe2284ba11beffb853a8c4-Paper.pdf.

Alberto Bietti and Julien Mairal. On the inductive bias of neural tangent kernels. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/c4ef9c39b300931b69a36fb3dbb8d60e-Paper.pdf.

Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22563–22575, June 2023.

Blake Bordelon and Cengiz Pehlevan. Self-consistent dynamical field theory of kernel evolution in wide neural networks. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 32240–32256. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/d027a5c93d484a4312cc486d399c62c1-Paper-Conference.pdf.

Blake Bordelon and Cengiz Pehlevan. Self-consistent dynamical field theory of kernel evolution in wide neural networks*. *Journal of Statistical Mechanics: Theory and Experiment*, 2023(11): 114009, nov 2023. doi: 10.1088/1742-5468/ad01b0. URL https://dx.doi.org/10.1088/1742-5468/ad01b0.

Blake Bordelon, Abdulkadir Canatar, and Cengiz Pehlevan. Spectrum dependent learning curves in kernel regression and wide neural networks. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1024–1034. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/bordelon20a.html.

Valentin De Bortoli. Convergence of denoising diffusion models under the manifold hypothesis. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL https://openreview.net/forum?id=MhK5aXo3gB. Expert Certification.

Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion schrödinger bridge with applications to score-based generative modeling. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=9BnCwiXB0ty.

Abdulkadir Canatar, Blake Bordelon, and Cengiz Pehlevan. Spectral bias and task-model alignment explain generalization in kernel regression and infinitely wide neural networks. *Nature Communications*, 12(1):2914, May 2021. ISSN 2041-1723. doi: 10.1038/s41467-021-23103-1. URL https://doi.org/10.1038/s41467-021-23103-1.

Nicolas Carlini, Jamie Hayes, Milad Nasr, Matthew Jagielski, Vikash Sehwag, Florian Tramèr, Borja Balle, Daphne Ippolito, and Eric Wallace. Extracting training data from diffusion models. In *32nd USENIX Security Symposium (USENIX Security 23)*, pp. 5253–5270, Anaheim, CA, August 2023. USENIX Association. ISBN 978-1-939133-37-3. URL https://www.usenix.org/conference/usenixsecurity23/presentation/carlini.

Chen-Hao Chao, Wei-Fang Sun, Bo-Wun Cheng, Yi-Chen Lo, Chia-Che Chang, Yu-Lun Liu, Yu-Lin Chang, Chia-Ping Chen, and Chun-Yi Lee. Denoising likelihood score matching for conditional score-based data generation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=LcF-EEt8cCC.

Minshuo Chen, Kaixuan Huang, Tuo Zhao, and Mengdi Wang. Score approximation, estimation and distribution recovery of diffusion models on low-dimensional data. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 4672–4712. PMLR, 23–29 Jul 2023a. URL https://proceedings.mlr.press/v202/chen23o.html.

Sitan Chen, Sinho Chewi, Holden Lee, Yuanzhi Li, Jianfeng Lu, and Adil Salim. The probability flow ode is provably fast. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 68552–68575. Curran Associates, Inc., 2023b. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/d84a27ff694345aacc21c72097a69ea2-Paper-Conference.pdf.

Sitan Chen, Sinho Chewi, Jerry Li, Yuanzhi Li, Adil Salim, and Anru Zhang. Sampling is as easy as learning the score: theory for diffusion models with minimal data assumptions. In *The Eleventh International Conference on Learning Representations*, 2023c. URL https://openreview.net/forum?id=zyLVMgsZ0U_.

Lénaïc Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/ae614c557843b1df326cb29c57225459-Paper.pdf.

A. Crisanti and H. Sompolinsky. Path integral approach to random neural networks. *Phys. Rev. E*, 98:062120, Dec 2018. doi: 10.1103/PhysRevE.98.062120. URL https://link.aps.org/doi/10.1103/PhysRevE.98.062120.

Salman Ul Hassan Dar, Arman Ghanaat, Jannik Kahmann, Isabelle Ayx, Theano Papavassiliu, Stefan O. Schoenberg, and Sandy Engelhardt. Investigating data memorization in 3d latent diffusion models for medical image synthesis. In *Deep Generative Models: Third MICCAI Workshop, DGM4MICCAI 2023, Held in Conjunction with MICCAI 2023, Vancouver, BC, Canada, October 8, 2023, Proceedings*, pp. 56–65, Berlin, Heidelberg, 2024. Springer-Verlag. ISBN 978-3-031-53766-0. doi: 10.1007/978-3-031-53767-7_6. URL https://doi.org/10.1007/978-3-031-53767-7_6.

Tim Dockhorn, Arash Vahdat, and Karsten Kreis. GENIE: Higher-order denoising diffusion solvers. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022a. URL https://openreview.net/forum?id=LKEYuYNOqx.

Tim Dockhorn, Arash Vahdat, and Karsten Kreis. Score-based generative modeling with critically-damped langevin diffusion. In *International Conference on Learning Representations*, 2022b. URL https://openreview.net/forum?id=CzceR82CYc.

Mario Geiger, Stefano Spigler, Arthur Jacot, and Matthieu Wyart. Disentangling feature and lazy training in deep neural networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2020 (11):113301, nov 2020. doi: 10.1088/1742-5468/abc4de. URL https://dx.doi.org/10.1088/1742-5468/abc4de.

Yinbin Han, Meisam Razaviyayn, and Renyuan Xu. Neural network-based score estimation in diffusion models: Optimization and generalization. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=h8GeqOxtd4.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Arthur Jacot, Franck Gabriel, and Clement Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/5a4be1fa34e62bb8a6ec6b91d2462f5a-Paper.pdf.

Zahra Kadkhodaie, Florentin Guth, Eero P Simoncelli, and Stéphane Mallat. Generalization in diffusion models arises from geometry-adaptive harmonic representations. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=ANvmVS2Yr0.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=k7FuTOWMOc7.

Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. Analyzing and improving the training dynamics of diffusion models, 2024.

Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. *arXiv preprint arXiv:2202.09778*, 2022.

Lorenzo Luzi, Paul M Mayer, Josue Casco-Rodriguez, Ali Siahkoohi, and Richard Baraniuk. Boomerang: Local sampling on image manifolds using diffusion models. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=NYdThkjNW1.

P. C. Martin, E. D. Siggia, and H. A. Rose. Statistical dynamics of classical systems. *Phys. Rev. A*, 8:423–437, Jul 1973. doi: 10.1103/PhysRevA.8.423. URL https://link.aps.org/doi/10.1103/PhysRevA.8.423.

Francesca Mignacco, Florent Krzakala, Pierfrancesco Urbani, and Lenka Zdeborová. Dynamical mean-field theory for stochastic gradient descent in gaussian mixture classification. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 9540–9550. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/6c81c83c4bd0b58850495f603ab45a93-Paper.pdf.

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5301–5310. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/rahaman19a.html.

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*, pp. 10684–10695, 2022.

Haozhe Shan and Blake Bordelon. A theory of neural tangent kernel alignment and its influence on training, 2022.

Samuel Smith, Erich Elsen, and Soham De. On the generalization benefit of noise in stochastic gradient descent. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9058–9067. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/smith20a.html.

Samuel L. Smith and Quoc V. Le. A bayesian perspective on generalization and stochastic gradient descent. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=BJij4yg0Z.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2256–2265, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/sohl-dickstein15.html.

Gowthami Somepalli, Vasu Singla, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Diffusion art or digital forgery? investigating data replication in diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6048–6058, June 2023a.

Gowthami Somepalli, Vasu Singla, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Understanding and mitigating copying in diffusion models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023b. URL https://openreview.net/forum?id=HtMXRGbUMt.

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 (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/3001ef257407d5a371a96dcd947c7d93-Paper.pdf.

Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. Sliced score matching: A scalable approach to density and score estimation. In Ryan P. Adams and Vibhav Gogate (eds.), *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference*, volume 115 of *Proceedings of Machine Learning Research*, pp. 574–584. PMLR, 22–25 Jul 2020. URL https://proceedings.mlr.press/v115/song20a.html.

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.

George Stein, Jesse C. Cresswell, Rasa Hosseinzadeh, Yi Sui, Brendan Leigh Ross, Valentin Villecroze, Zhaoyan Liu, Anthony L. Caterini, Eric Taylor, and Gabriel Loaiza-Ganem. Exposing flaws of generative model evaluation metrics and their unfair treatment of diffusion models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=08zf7kTOoh.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011. doi: 10.1162/NECO_a_00142.

Nikhil Vyas, Sham M. Kakade, and Boaz Barak. On provable copyright protection for generative models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 35277–35299. PMLR, 2023. URL https://proceedings.mlr.press/v202/vyas23b.html.

Blake Woodworth, Suriya Gunasekar, Jason D. Lee, Edward Moroshko, Pedro Savarese, Itay Golan, Daniel Soudry, and Nathan Srebro. Kernel and rich regimes in overparametrized models. In Jacob Abernethy and Shivani Agarwal (eds.), *Proceedings of Thirty Third Conference on Learning Theory*, volume 125 of *Proceedings of Machine Learning Research*, pp. 3635–3673. PMLR, 09–12 Jul 2020. URL https://proceedings.mlr.press/v125/woodworth20a.html.

Yilun Xu, Shangyuan Tong, and Tommi S. Jaakkola. Stable target field for reduced variance score estimation in diffusion models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WmIwYTd0YTF.

Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *ACM Comput. Surv.*, 56(4), nov 2023. ISSN 0360-0300. doi: 10.1145/3626235. URL https://doi.org/10.1145/3626235.

TaeHo Yoon, Joo Young Choi, Sehyun Kwon, and Ernest K. Ryu. Diffusion probabilistic models generalize when they fail to memorize. In *ICML 2023 Workshop on Structured Probabilistic Inference & Generative Modeling*, 2023. URL https://openreview.net/forum?id=shciCbSk9h.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Sy8gdB9xx.

Huijie Zhang, Jinfan Zhou, Yifu Lu, Minzhe Guo, Peng Wang, Liyue Shen, and Qing Qu. The emergence of reproducibility and consistency in diffusion models. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=HsliOqZkc0.

A Optimizing objective reproduces training distribution

In this appendix, we characterize the optima of the naive and DSM objectives introduced in Sec. 2, and in particular show that one (naively) theoretically expects diffusion models to reproduce the training distribution in the absence of expressivity-related constraints.

A.1 Denoising score matching preserves optima of naive objective

First, we reestablish the well-known fact that the optima of the naive objective

$$J_0(\boldsymbol{\theta}) := \frac{1}{2}\,\mathbb{E}_{t,\boldsymbol{x}}\left\{\lambda_t\|\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t)-\boldsymbol{s}(\boldsymbol{x},t)\|_2^2\right\} = \int \frac{\lambda_t}{2}\|\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t)-\boldsymbol{s}(\boldsymbol{x},t)\|_2^2\; p(\boldsymbol{x}|t)p(t)\; d\boldsymbol{x}dt \tag{18}$$

and DSM objective

$$\begin{aligned} J_1(\boldsymbol{\theta}) &:= \frac{1}{2}\mathbb{E}_{t,\boldsymbol{x}_0,\boldsymbol{x}}\left\{\lambda_t\|\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t)-\tilde{\boldsymbol{s}}(\boldsymbol{x},t;\boldsymbol{x}_0)\|_2^2\right\} \\ &= \int \frac{\lambda_t}{2}\;\|\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t)-\tilde{\boldsymbol{s}}(\boldsymbol{x},t;\boldsymbol{x}_0)\|_2^2\; p(\boldsymbol{x}|\boldsymbol{x}_0,t)p_{data}(\boldsymbol{x}_0)p(t)\; d\boldsymbol{x}d\boldsymbol{x}_0dt \end{aligned} \tag{19}$$

are the same (Vincent, 2011; Song & Ermon, 2019; Song et al., 2021). Assume that $\boldsymbol{x},\boldsymbol{x}_0 \in \mathbb{R}^D$ and that $\boldsymbol{\theta}\in\mathbb{R}^F$. The gradient of J_0 with respect to $\boldsymbol{\theta}$ is

$$\frac{\partial J_0}{\partial \boldsymbol{\theta}} = \int \lambda_t \frac{\partial \hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t)^T}{\partial \boldsymbol{\theta}}\;[\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t)-\boldsymbol{s}(\boldsymbol{x},t)]\;\, p(\boldsymbol{x}|t)p(t)\; d\boldsymbol{x}dt \tag{20}$$

where $\partial\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t)/\partial\boldsymbol{\theta}$ is the $D\times F$ Jacobian matrix of the score estimator. The gradient of J_1 is

$$\frac{\partial J_1}{\partial \boldsymbol{\theta}} = \int \lambda_t \frac{\partial \hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t)^T}{\partial \boldsymbol{\theta}}\;[\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t)-\tilde{\boldsymbol{s}}(\boldsymbol{x},t;\boldsymbol{x}_0)]\;\, p(\boldsymbol{x}|\boldsymbol{x}_0,t)p_{data}(\boldsymbol{x}_0)p(t)\; d\boldsymbol{x}d\boldsymbol{x}_0dt\;. \tag{21}$$

At this point, we make two observations about the gradient of J_1. First, the term on the left does not depend on $\boldsymbol{x}_0$, so we can marginalize over $\boldsymbol{x}_0$. Explicitly,

$$\int \lambda_t \frac{\partial \hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t)^T}{\partial \boldsymbol{\theta}}\,\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t)\; p(\boldsymbol{x}|\boldsymbol{x}_0,t)p_{data}(\boldsymbol{x}_0)p(t)\; d\boldsymbol{x}d\boldsymbol{x}_0dt = \int \lambda_t \frac{\partial \hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t)^T}{\partial \boldsymbol{\theta}}\,\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t)\; p(\boldsymbol{x}|t)p(t)\; d\boldsymbol{x}dt\;.$$

Second, the term on the right only depends on $\boldsymbol{x}_0$ through the proxy score target. Moreover,

$$\begin{aligned} \int \tilde{\boldsymbol{s}}(\boldsymbol{x},t;\boldsymbol{x}_0)\; p(\boldsymbol{x}|\boldsymbol{x}_0,t)p_{data}(\boldsymbol{x}_0)\; d\boldsymbol{x}_0 &= \int \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}|\boldsymbol{x}_0,t)\; p(\boldsymbol{x}|\boldsymbol{x}_0,t)p_{data}(\boldsymbol{x}_0)\; d\boldsymbol{x}_0 \\ &= \int \nabla_{\boldsymbol{x}} p(\boldsymbol{x}|\boldsymbol{x}_0,t)p_{data}(\boldsymbol{x}_0)\; d\boldsymbol{x}_0 \\ &= \nabla_{\boldsymbol{x}} \int p(\boldsymbol{x}|\boldsymbol{x}_0,t)p_{data}(\boldsymbol{x}_0)\; d\boldsymbol{x}_0 \\ &= \nabla_{\boldsymbol{x}} p(\boldsymbol{x}|t) \\ &= \boldsymbol{s}(\boldsymbol{x},t)p(\boldsymbol{x}|t)\;. \end{aligned} \tag{22}$$

Hence, the gradient of J_0 is the same as the gradient of J_1, so they have the same optima. If the score approximator is arbitrarily expressive and smooth in its parameters, we in particular have that the true score (a global minimum of J_0) is an optimum of the DSM objective.

This optimum is *also* the global minimum of J_1. Note that J_1 can be written as

$$\begin{aligned} &\mathbb{E}_{t,\boldsymbol{x}_0,\boldsymbol{x}}\left\{\frac{\lambda_t}{2}\|\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t)-\boldsymbol{s}(\boldsymbol{x},t)+\boldsymbol{s}(\boldsymbol{x},t)-\tilde{\boldsymbol{s}}(\boldsymbol{x},t;\boldsymbol{x}_0)\|_2^2\right\} \\ &= \mathbb{E}_{t,\boldsymbol{x}_0,\boldsymbol{x}}\left\{\frac{\lambda_t}{2}\left(\|\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t)-\boldsymbol{s}(\boldsymbol{x},t)\|_2^2 + 2[\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t)-\boldsymbol{s}(\boldsymbol{x},t)]\cdot[\boldsymbol{s}(\boldsymbol{x},t)-\tilde{\boldsymbol{s}}(\boldsymbol{x},t;\boldsymbol{x}_0)] + \|\boldsymbol{s}(\boldsymbol{x},t)-\tilde{\boldsymbol{s}}(\boldsymbol{x},t;\boldsymbol{x}_0)\|_2^2\;\right)\right\}\;. \end{aligned}$$

The first term is precisely equal to J_0. The second term vanishes, since (as shown by Eq. 22)

$$\mathbb{E}_{\boldsymbol{x}_0|\boldsymbol{x},t}[\tilde{\boldsymbol{s}}(\boldsymbol{x},t;\boldsymbol{x}_0)] = \boldsymbol{s}(\boldsymbol{x},t)\;. \tag{23}$$

Hence, we have that

$$J_1 = J_0 + \frac{1}{2}\mathbb{E}_{t,\boldsymbol{x}}\left\{\lambda_t\;\mathrm{tr}(\;\mathrm{Cov}_{\boldsymbol{x}_0|\boldsymbol{x},t}(\tilde{\boldsymbol{s}})\;)\right\}\;. \tag{24}$$

In words: J_1 is equal to J_0 up to a $\boldsymbol{\theta}$-independent term that is a weighted combination of proxy score variances.

A.2 Training distribution reproduction

In practice, the training set consists of $1 \leq M < \infty$ examples (e.g., images) which together define

$$p_{data}(\boldsymbol{x}_0) = \frac{1}{M} \sum_{m=1}^{M} \delta(\boldsymbol{x}_0 - \boldsymbol{\mu}_m) \ . \tag{25}$$

The corresponding 'corrupted' distribution, given our choice of forward process (see Sec. 2), is

$$p(\boldsymbol{x}|t) = \frac{1}{M} \sum_{m=1}^{M} \mathcal{N}(\boldsymbol{x}; \alpha_t \boldsymbol{\mu}_m, \boldsymbol{S}_t) \ . \tag{26}$$

Usually, model updates utilize batches of samples from $p(\boldsymbol{x}, \boldsymbol{x}_0, t)$ (Song et al., 2021; Karras et al., 2022). As training proceeds, the model sees an ever larger number P of samples from this distribution, making the empirical objective

$$J_1(\boldsymbol{\theta}; P) := \frac{1}{P} \sum_{n=1}^{P} \frac{\lambda(t^{(n)})}{2} \| \hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}^{(n)}, t^{(n)}) - \tilde{\boldsymbol{s}}(\boldsymbol{x}^{(n)}, t^{(n)}; \boldsymbol{x}_0^{(n)}) \|_2^2 \ , \tag{27}$$

where the n superscripts index different (independent) samples from $p(\boldsymbol{x}, \boldsymbol{x}_0, t) = p(\boldsymbol{x}|\boldsymbol{x}_0, t) p_{data}(\boldsymbol{x}_0) p(t)$. For P sufficiently large, by the central limit theorem, we expect the empirical objective to be extremely close to the true objective, and hence share its global minimum. But the global minimum is the true score, i.e.,

$$\boldsymbol{s}(\boldsymbol{x}, t) = \sum_{m=1}^{M} \boldsymbol{S}_t^{-1} (\alpha_t \boldsymbol{\mu}_m - \boldsymbol{x}) \frac{\mathcal{N}(\boldsymbol{x}; \alpha_t \boldsymbol{\mu}_m, \boldsymbol{S}_t)}{\sum_{m'} \mathcal{N}(\boldsymbol{x}; \alpha_t \boldsymbol{\mu}_{m'}, \boldsymbol{S}_t)} \ .$$

Since integrating the PF-ODE using this score produces samples from $p_{data}(\boldsymbol{x}_0)$—as $t \to 0$, $\boldsymbol{S}_t \to \boldsymbol{0}_D$, so the asymptotic 'force' pushing $\boldsymbol{x}_t$ towards an example becomes infinitely strong—we expect expressive diffusion models trained on the DSM objective using a large number of samples to reproduce training examples.

B Covariance of proxy score

In this appendix, we compute the covariance of the proxy score $\tilde{\boldsymbol{s}}(\boldsymbol{x}, t; \boldsymbol{x}_0) := \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}|\boldsymbol{x}_0, t)$ with respect to $p(\boldsymbol{x}_0|\boldsymbol{x}, t)$. We also show how this covariance is connected to Fisher information, and explicitly compute it in the case that $p_{data}(\boldsymbol{x}_0)$ is an isotropic Gaussian mixture.

B.1 Computing covariance of proxy score

Note that

$$\frac{\partial^2}{\partial x_i \partial x_j} p(\boldsymbol{x}|\boldsymbol{x}_0, t) = \left[-S^{-1}_{t,ij} + \tilde{s}_i \tilde{s}_j \ \right] \ p(\boldsymbol{x}|\boldsymbol{x}_0, t) \ . \tag{28}$$

Using this fact, we can write

$$\begin{aligned}
\mathrm{Cov}_{\boldsymbol{x}_0|\boldsymbol{x},t}(\tilde{s}_i, \tilde{s}_j) &= \int \tilde{s}_i \tilde{s}_j \ \frac{p(\boldsymbol{x}|\boldsymbol{x}_0, t) p_{data}(\boldsymbol{x}_0)}{p(\boldsymbol{x}|t)} \ d\boldsymbol{x}_0 - s_i s_j \\
&= \int \frac{1}{p(\boldsymbol{x}|t)} \left[S^{-1}_{t,ij} + \frac{\partial^2}{\partial x_i \partial x_j} \right] \ p(\boldsymbol{x}|\boldsymbol{x}_0, t) p_{data}(\boldsymbol{x}_0) \ d\boldsymbol{x}_0 - s_i s_j \\
&= \int \frac{1}{p(\boldsymbol{x}|t)} \left[S^{-1}_{t,ij} + \frac{\partial^2}{\partial x_i \partial x_j} \right] p(\boldsymbol{x}|t) \ \frac{p(\boldsymbol{x}|\boldsymbol{x}_0, t) p_{data}(\boldsymbol{x}_0)}{p(\boldsymbol{x}|t)} \ d\boldsymbol{x}_0 - s_i s_j \qquad (29) \\
&= S^{-1}_{t,ij} + \frac{1}{p(\boldsymbol{x}|t)} \frac{\partial^2 p(\boldsymbol{x}|t)}{\partial x_i \partial x_j} - s_i s_j \\
&= S^{-1}_{t,ij} + \frac{\partial^2}{\partial x_i \partial x_j} \log p(\boldsymbol{x}|t) \ .
\end{aligned}$$

B.2 Connection to Fisher information

By definition, if $p(\boldsymbol{x}_0|\boldsymbol{x}, t)$ is viewed as a distribution with parameter vector $\boldsymbol{x}$, and t is viewed as a hyperparameter, the Fisher information $\mathcal{I}_F$ is defined as

$$\begin{aligned}
\mathcal{I}_F(\boldsymbol{x}|t) &:= \int \frac{\partial \log p(\boldsymbol{x}_0|\boldsymbol{x}, t)}{\partial x_i} \cdot \frac{\partial \log p(\boldsymbol{x}_0|\boldsymbol{x}, t)}{\partial x_j} \ p(\boldsymbol{x}_0|\boldsymbol{x}, t) \ d\boldsymbol{x}_0 \\
&= \int \left[\frac{\partial \log p(\boldsymbol{x}|\boldsymbol{x}_0, t)}{\partial x_i} - \frac{\partial \log p(\boldsymbol{x}|t)}{\partial x_i} \right] \left[\frac{\partial \log p(\boldsymbol{x}|\boldsymbol{x}_0, t)}{\partial x_j} - \frac{\partial \log p(\boldsymbol{x}|t)}{\partial x_j} \right] \ p(\boldsymbol{x}_0|\boldsymbol{x}, t) \ d\boldsymbol{x}_0 \\
&= \int \left[\tilde{s}_i - s_i \right] \left[\tilde{s}_j - s_j \right] \ p(\boldsymbol{x}_0|\boldsymbol{x}, t) \ d\boldsymbol{x}_0 \\
&= \mathrm{Cov}_{\boldsymbol{x}_0|\boldsymbol{x},t} \left(\tilde{s}_i, \tilde{s}_j \right) \ .
\end{aligned} \tag{30}$$

B.3 Explicit covariance for isotropic Gaussian mixture training distribution

Suppose that $p(\boldsymbol{x}_0)$ and $p(\boldsymbol{x}|t)$ are

$$p_{data}(\boldsymbol{x}_0) = \frac{1}{M} \sum_m \mathcal{N}(\boldsymbol{x}_0; \boldsymbol{\mu}_m, \sigma_0^2 \boldsymbol{I}) \qquad\qquad p(\boldsymbol{x}|t) = \frac{1}{M} \sum_m \mathcal{N}(\boldsymbol{x}; \alpha_t \boldsymbol{\mu}_m, \alpha_t^2 \sigma_0^2 \boldsymbol{I} + \boldsymbol{S}_t) \ . \tag{31}$$

Note that the delta mixture case is an example ($\sigma_0^2 = 0$). Define the softmax distribution

$$p(m|\boldsymbol{x}, t) := \frac{\mathcal{N}(\boldsymbol{x}; \alpha_t \boldsymbol{\mu}_m, \alpha_t^2 \sigma_0^2 \boldsymbol{I} + \boldsymbol{S}_t)}{\sum_{m'} \mathcal{N}(\boldsymbol{x}; \alpha_t \boldsymbol{\mu}_{m'}, \alpha_t^2 \sigma_0^2 \boldsymbol{I} + \boldsymbol{S}_t)} \tag{32}$$

on $\mathcal{M} = \{1, ..., M\}$. The first and second derivatives of $p(\boldsymbol{x}|t)$ can be written in terms of expectations with respect to this distribution, since

$$\frac{1}{p(\boldsymbol{x}|t)} \frac{\partial p(\boldsymbol{x}|t)}{\partial \boldsymbol{x}} = \sum_m (\alpha_t^2 \sigma_0^2 \boldsymbol{I} + \boldsymbol{S}_t)^{-1} (\alpha_t \boldsymbol{\mu}_m - \boldsymbol{x}) p(m|\boldsymbol{x}, t) = (\alpha_t^2 \sigma_0^2 \boldsymbol{I} + \boldsymbol{S}_t)^{-1} (\alpha_t \langle \boldsymbol{\mu} \rangle_{\mathcal{M}} - \boldsymbol{x})$$

and the Hessian matrix ($H_{ij} := \partial^2_{ij} p(\boldsymbol{x}|t)$) is

$$
\begin{aligned}
\frac{\boldsymbol{H}}{p(\boldsymbol{x}|t)} &= \sum_m \left[-(\alpha_t^2\sigma_0^2\boldsymbol{I}+\boldsymbol{S}_t)^{-1} + (\alpha_t^2\sigma_0^2\boldsymbol{I}+\boldsymbol{S}_t)^{-1}(\alpha_t\boldsymbol{\mu}_m-\boldsymbol{x})(\alpha_t\boldsymbol{\mu}_m-\boldsymbol{x})^T(\alpha_t^2\sigma_0^2\boldsymbol{I}+\boldsymbol{S}_t)^{-1}\right] p(m|\boldsymbol{x},t) \\
&= -(\alpha_t^2\sigma_0^2\boldsymbol{I}+\boldsymbol{S}_t)^{-1} + (\alpha_t^2\sigma_0^2\boldsymbol{I}+\boldsymbol{S}_t)^{-1}\mathbb{E}_{\mathcal{M}}\left\{(\alpha_t\boldsymbol{\mu}-\boldsymbol{x})(\alpha_t\boldsymbol{\mu}-\boldsymbol{x})^T\right\}(\alpha_t^2\sigma_0^2\boldsymbol{I}+\boldsymbol{S}_t)^{-1} \\
&= -(\alpha_t^2\sigma_0^2\boldsymbol{I}+\boldsymbol{S}_t)^{-1} + (\alpha_t^2\sigma_0^2\boldsymbol{I}+\boldsymbol{S}_t)^{-1}\left[\alpha_t^2\mathrm{Cov}_{\mathcal{M}}(\boldsymbol{\mu}) + (\alpha_t\langle\boldsymbol{\mu}\rangle_{\mathcal{M}}-\boldsymbol{x})(\alpha_t\langle\boldsymbol{\mu}\rangle_{\mathcal{M}}-\boldsymbol{x})^T\right](\alpha_t^2\sigma_0^2\boldsymbol{I}+\boldsymbol{S}_t)^{-1} \ .
\end{aligned}
$$

Then we have

$$
\frac{\partial^2 \log p(\boldsymbol{x}|t)}{\partial x_i \partial x_j} = -(\alpha_t^2\sigma_0^2\boldsymbol{I}+\boldsymbol{S}_t)^{-1} + \alpha_t^2(\alpha_t^2\sigma_0^2\boldsymbol{I}+\boldsymbol{S}_t)^{-1}\mathrm{Cov}_{\mathcal{M}}(\boldsymbol{\mu})(\alpha_t^2\sigma_0^2\boldsymbol{I}+\boldsymbol{S}_t)^{-1} \tag{33}
$$

and hence that

$$
\mathrm{Cov}_{\boldsymbol{x}_0|\boldsymbol{x},t}(\tilde{\boldsymbol{s}}) = \boldsymbol{S}_t^{-1} - (\alpha_t^2\sigma_0^2\boldsymbol{I}+\boldsymbol{S}_t)^{-1} + \alpha_t^2(\alpha_t^2\sigma_0^2\boldsymbol{I}+\boldsymbol{S}_t)^{-1}\mathrm{Cov}_{\mathcal{M}}(\boldsymbol{\mu})(\alpha_t^2\sigma_0^2\boldsymbol{I}+\boldsymbol{S}_t)^{-1} \ .
$$

For a delta mixture training distribution, since $\sigma_0^2 = 0$, the covariance simplifies to

$$
\mathrm{Cov}_{\boldsymbol{x}_0|\boldsymbol{x},t}(\tilde{\boldsymbol{s}}) = \alpha_t^2\boldsymbol{S}_t^{-1}\,\mathrm{Cov}_{\mathcal{M}}(\boldsymbol{\mu})\boldsymbol{S}_t^{-1} \ . \tag{34}
$$

C PATH-INTEGRAL REPRESENTATION OF LEARNED DISTRIBUTION

In this appendix, we derive a path-integral description of the 'typical' distribution learned by diffusion models. We do this in three stages. First, we derive a path-integral description of the PF-ODE. Next, we derive a path-integral description of a more general kind of stochastic process. Finally, we show that averaging the path-integral representation of the PF-ODE over sample realizations produces a path integral whose dynamics correspond to those of the aforementioned stochastic process.

C.1 WARM-UP: DERIVING A PATH-INTEGRAL REPRESENTATION OF THE PF-ODE

A general ODE can be written as

$$\dot{\boldsymbol{x}}_t = \boldsymbol{f}(\boldsymbol{x}_t, t) \tag{35}$$

where $\boldsymbol{x}_t \in \mathbb{R}^D$ and $t \in [0, T]$. We will assume that $\boldsymbol{f}$ is smooth to avoid technical issues. If we discretize time, and slightly abuse notation by using t and T to refer to integer-valued indices instead of real-valued times, we can write the trajectory as $\{x_T, x_{T-1}, ..., x_1, x_0\}$ and the corresponding updates in the form

$$\boldsymbol{x}_t = \boldsymbol{x}_{t+1} - \boldsymbol{f}(\boldsymbol{x}_{t+1}, t+1)\Delta t \, . \tag{36}$$

Note that our discretization corresponds to a first-order Euler update scheme. In the small Δt limit, this specific choice does not matter, even if it matters in practice; we use it to slightly simplify our argument. Conditional on the initial point $\boldsymbol{x}_T$, the probability of reaching another point $\boldsymbol{x}_0$ after T backwards-time steps is

$$p(\boldsymbol{x}_0|\boldsymbol{x}_T) = \int \delta(\boldsymbol{x}_0 - \boldsymbol{x}_1 + \boldsymbol{f}(\boldsymbol{x}_1, 1)\Delta t) \cdots \delta(\boldsymbol{x}_{T-1} - \boldsymbol{x}_T + \boldsymbol{f}(\boldsymbol{x}_T, T)\Delta t) \; d\boldsymbol{x}_1 \cdots d\boldsymbol{x}_{T-1} \tag{37}$$

where δ is the Dirac delta function. Here, we will employ a well-known integral representation of the Dirac delta function:

$$\delta(\boldsymbol{x} - \boldsymbol{x}') = \int \frac{d\boldsymbol{p}}{(2\pi)^D} \; \exp\left\{-i\boldsymbol{p} \cdot (\boldsymbol{x} - \boldsymbol{x}')\right\} \tag{38}$$

where $\boldsymbol{p}$ is integrated over all of $\mathbb{R}^D$. Our expression for $p(\boldsymbol{x}_0|\boldsymbol{x}_T)$ becomes

$$p(\boldsymbol{x}_0|\boldsymbol{x}_T) = \int \frac{d\boldsymbol{p}_0}{(2\pi)^D} \frac{d\boldsymbol{x}_1 d\boldsymbol{p}_1}{(2\pi)^D} \cdots \frac{d\boldsymbol{x}_{T-1} d\boldsymbol{p}_{T-1}}{(2\pi)^D} \; \exp\left\{ \sum_{t=0}^{T-1} -i\boldsymbol{p}_t \cdot [\boldsymbol{x}_t - \boldsymbol{x}_{t+1} + \boldsymbol{f}(\boldsymbol{x}_{t+1}, t+1)\Delta t] \right\} . \tag{39}$$

Schematically, we can write this path integral as a 'sum over paths'

$$p(\boldsymbol{x}_0|\boldsymbol{x}_T) = \int \mathcal{D}[\boldsymbol{p}_t]\mathcal{D}[\boldsymbol{x}_t] \; \exp\left\{ \int_0^T -i\boldsymbol{p}_t \cdot [-\dot{\boldsymbol{x}}_t + \boldsymbol{f}(\boldsymbol{x}_t, t)] \; dt \right\} , \tag{40}$$

although explicitly using this form is unnecessary for our purposes. (This is good, since remaining in discrete time allows us to avoid various thorny mathematical issues.) For the particular choice of $\boldsymbol{f}$ associated with the PF-ODE, we have discrete and schematic forms

$$p(\boldsymbol{x}_0|\boldsymbol{x}_T) = \int \frac{d\boldsymbol{p}_0}{(2\pi)^D} \frac{d\boldsymbol{x}_1 d\boldsymbol{p}_1}{(2\pi)^D} \cdots \frac{d\boldsymbol{x}_{T-1} d\boldsymbol{p}_{T-1}}{(2\pi)^D} \; e^{\sum_{t=0}^{T-1} -i\boldsymbol{p}_t \cdot [\boldsymbol{x}_t - \boldsymbol{x}_{t+1} - (\beta_{t+1}\boldsymbol{x}_{t+1} + \boldsymbol{D}_{t+1}\boldsymbol{s}(\boldsymbol{x}_{t+1}, t+1))\Delta t]}$$

$$p(\boldsymbol{x}_0|\boldsymbol{x}_T) = \int \mathcal{D}[\boldsymbol{p}_t]\mathcal{D}[\boldsymbol{x}_t] \; \exp\left\{ \int_0^T i\boldsymbol{p}_t \cdot [\dot{\boldsymbol{x}}_t + \beta_t \boldsymbol{x}_t + \boldsymbol{D}_t \boldsymbol{s}(\boldsymbol{x}_t, t)] \; dt \right\} .$$

C.2 Deriving a path-integral representation of a more general process

Consider a more general type of backwards, discrete-time stochastic process. Once again, suppose that a variable $\boldsymbol{x}_t \in \mathbb{R}^D$ evolves backwards in time from an initial point $\boldsymbol{x}_T$. But this time, suppose that the transition between $\boldsymbol{x}_{t+1}$ and $\boldsymbol{x}_t$ depends upon some set of K independent standard normal random variables $\{\xi_k\}$. In particular, suppose that discrete-time updates have the form

$$x_{tj} = x_{t+1,j} - f_j(\boldsymbol{x}_{t+1}, t+1)\Delta t + \sum_{k=1}^{K} G_{jk}(\boldsymbol{x}_{t+1}, t+1)\ \xi_k\ \Delta t\ , \tag{41}$$

i.e., updates are the same as before except for the new noise term. In general, the noise term is quite complicated; $\boldsymbol{G}$ is a $D \times K$ matrix which can depend explicitly on both the current state and the current time. The process described by the above updates is generally not Markov, since noise added at different time steps can depend on some of the same ξ_k variables, and hence the amount of noise added at one time step can be correlated with the amount of noise added at some other time step.

What is the distribution of $\boldsymbol{x}_0$, the result of T steps of this process, conditional on a starting point $\boldsymbol{x}_T$? We know that each update depends only on the previous state and the noise variables, so

$$p(\boldsymbol{x}_0|\boldsymbol{x}_T) = \int p(\boldsymbol{x}_0|\boldsymbol{x}_1, \{\xi_k\}) p(\boldsymbol{x}_1|\boldsymbol{x}_2, \{\xi_k\}) \cdots p(\boldsymbol{x}_{T-1}|\boldsymbol{x}_T, \{\xi_k\})\ p(\{\xi_k\})\ d\boldsymbol{x}_1 \cdots d\boldsymbol{x}_{T-1} d\{\xi_k\}\ .$$

In particular, conditional on the previous state and the noise variables, updates are deterministic. This allows us to write the above transition probability as

$$\int \left[\prod_{j=1}^{D} \prod_{t=0}^{T-1} \delta\left(x_{t,j} - x_{t+1,j} + f_j(\boldsymbol{x}_{t+1}, t+1)\Delta t + \sum_{k=1}^{K} G_{jk}(\boldsymbol{x}_{t+1}, t+1)\ \xi_k\ \Delta t \right) \right]\ p(\{\xi_k\})\ d\boldsymbol{x}_1 \cdots d\boldsymbol{x}_{T-1} d\{\xi_k\}\ .$$

Using the same integral representation of the Dirac delta function that we used above, this becomes

$$\int e^{\sum_{t,j} -ip_{t,j}\left[x_{t,j} - x_{t+1,j} + f_j(\boldsymbol{x}_{t+1}, t+1)\Delta t + \sum_{k=1}^{K} G_{jk}(\boldsymbol{x}_{t+1}, t+1)\ \xi_k\ \Delta t\right]}\ p(\{\xi_k\})\ \frac{d\boldsymbol{p}_0}{(2\pi)^D} \frac{d\boldsymbol{x}_1 d\boldsymbol{p}_1}{(2\pi)^D} \cdots \frac{d\boldsymbol{x}_{T-1} d\boldsymbol{p}_{T-1}}{(2\pi)^D} d\{\xi_k\}\ .$$

Although this appears to be extremely complicated, it can be considerably simplified by doing the integral over the noise variables. Since the noise variables are all independent and standard normal,

$$p(\{\xi_k\}) = \frac{1}{(2\pi)^{k/2}} \exp\left\{ -\frac{\xi_1^2}{2} - \cdots - \frac{\xi_K^2}{2} \right\}\ . \tag{42}$$

Hence, the integral over the noise variables is a typical Gaussian integral with a linear term. We can save time by recognizing the integral as essentially computing the characteristic function of a standard normal; more precisely, we have

$$\begin{aligned} I_k &= \int \exp\left\{ -i\xi_k \sum_{t=0}^{T-1} \sum_{j=1}^{D} p_{t,j} G_{jk}(\boldsymbol{x}_{t+1}, t+1)\Delta t \right\} \frac{e^{-\xi_k^2/2}}{\sqrt{2\pi}}\ d\xi_k \\ &= \exp\left\{ -\frac{1}{2} \sum_{t=0}^{T-1} \sum_{t'=0}^{T-1} \sum_{j=1}^{D} \sum_{j'=1}^{D} p_{t,j} G_{jk}(\boldsymbol{x}_{t+1}, t+1) G_{j'k}(\boldsymbol{x}_{t'+1}, t'+1) p_{t',j'} \Delta t \Delta t \right\} \end{aligned} \tag{43}$$

for each ξ_k. Putting everything together, we find that $p(\boldsymbol{x}_0|\boldsymbol{x}_T)$ can be written

$$\int e^{\sum_{t,j} -ip_{t,j}[x_{t,j} - x_{t+1,j} + f_j(\boldsymbol{x}_{t+1}, t+1)\Delta t] - \frac{1}{2} \sum_{t,t',j,j'} \sum_{k=1}^{K} p_{t,j} G_{jk}(\boldsymbol{x}_{t+1}, t+1) G_{j'k}(\boldsymbol{x}_{t'+1}, t'+1) p_{t',j'} \Delta t \Delta t}\ \frac{d\{\boldsymbol{x}_t\} d\{\boldsymbol{p}_t\}}{(2\pi)^{DT}}$$

where we have used the shorthand $d\{\boldsymbol{x}_t\} d\{\boldsymbol{p}_t\} := d\boldsymbol{p}_0\ d\boldsymbol{x}_1 d\boldsymbol{p}_1 \cdots d\boldsymbol{x}_{T-1} d\boldsymbol{p}_{T-1}$. This is our final answer, although it is more enlightening to write it in its schematic continuous-time form. We obtain

$$p(\boldsymbol{x}_0|\boldsymbol{x}_T) = \int \mathcal{D}[\boldsymbol{p}_t] \mathcal{D}[\boldsymbol{x}_t]\ \exp\left\{ \int_0^T -i\boldsymbol{p}_t \cdot [-\dot{\boldsymbol{x}}_t + \boldsymbol{f}(\boldsymbol{x}_t, t)]\ dt\ - \frac{1}{2} \int_0^T \int_0^T \boldsymbol{p}_t^T \boldsymbol{V}(\boldsymbol{x}_t, t; \boldsymbol{x}_{t'}, t') \boldsymbol{p}_{t'}\ dt dt' \right\}$$

where we have defined the state- and time-dependent $D \times D$ V-kernel $V_{ij}(\boldsymbol{x}_t, t; \boldsymbol{x}_{t'}, t')$ via

$$V_{ij}(\boldsymbol{x}_t, t; \boldsymbol{x}_{t'}, t') := \sum_{k=1}^{K} G_{ik}(\boldsymbol{x}_t, t) G_{jk}(\boldsymbol{x}_{t'}, t')\ , \tag{44}$$

or equivalently via $\boldsymbol{V}(\boldsymbol{x}_t, t; \boldsymbol{x}_{t'}, t') := \boldsymbol{G}(\boldsymbol{x}_t, t)\ \boldsymbol{G}^T(\boldsymbol{x}_{t'}, t')$. Note that it is positive semidefinite.

C.3 Averaging learned distribution over sample realizations

What is the 'typical' distribution learned by an ensemble of diffusion models which differ only in the samples each used during training? In this subsection, we show that the net effect of averaging over sample realizations is to contribute a noise term to the PF-ODE. The path-integral representation we obtain is of the class we discussed in the previous subsection.

Suppose a diffusion model is associated with a parameterized score approximator $\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}, t)$. The distribution learned by the diffusion model is then

$$q(\boldsymbol{x}_0|\boldsymbol{x}_T;\boldsymbol{\theta}) = \int \mathcal{D}[\boldsymbol{p}_t]\mathcal{D}[\boldsymbol{x}_t]\ \exp\left\{\int_0^T \boldsymbol{ip}_t \cdot [\dot{\boldsymbol{x}}_t + \beta_t \boldsymbol{x}_t + \boldsymbol{D}_t \hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)]\ dt\right\}\ , \tag{45}$$

where we have used the schematic form of the PF-ODE path-integral representation for clarity. (Moving to discrete time does not affect our arguments, but only makes notation more cumbersome.) Averaging over sample realizations is mathematically equivalent to computing the characteristic function of the score approximator. The sample-averaged q, $\mathbb{E}_{\boldsymbol{\theta}}[q(\boldsymbol{x}_0|\boldsymbol{x}_T;\boldsymbol{\theta})] = [q(\boldsymbol{x}_0|\boldsymbol{x}_T)]$, is

$$[q(\boldsymbol{x}_0|\boldsymbol{x}_T)] = \int \mathcal{D}[\boldsymbol{p}_t]\mathcal{D}[\boldsymbol{x}_t]\ \exp\left\{\int_0^T \boldsymbol{ip}_t \cdot [\dot{\boldsymbol{x}}_t + \beta_t \boldsymbol{x}_t]\ dt\right\} \mathbb{E}_{\boldsymbol{\theta}}\left[e^{\int_0^T \boldsymbol{ip}_t^T \boldsymbol{D}_t \hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)\ dt}\right]\ . \tag{46}$$

Assuming the score approximator ensemble is well-behaved, its characteristic function can be written as a cumulant expansion. Here, we have

$$\begin{aligned} &\log \mathbb{E}_{\boldsymbol{\theta}}\left[e^{\int_0^T \boldsymbol{ip}_t^T \boldsymbol{D}_t \hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)\ dt}\right] \\ &= \int_0^T \boldsymbol{ip}_t \boldsymbol{D}_t [\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)]\ dt - \frac{1}{2}\int_0^T \int_0^T \boldsymbol{p}_t^T \boldsymbol{D}_t \mathrm{Cov}_{\boldsymbol{\theta}}\left[\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t), \hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}_{t'}, t')\right] \boldsymbol{D}_{t'} \boldsymbol{p}_{t'} + \cdots \end{aligned} \tag{47}$$

where the dots indicate higher-order cumulants and $[\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)]$ indicates the ensemble-averaged score approximator. In this work, we neglect the higher-order terms. Often, they are suppressed by some factor (e.g., the number of model parameters divided by the number of samples).

We obtain dynamics of the class described in the previous subsection. Here, the $D \times D$ V-kernel is

$$V_{ij}(\boldsymbol{x}_t, t; \boldsymbol{x}_{t'}, t') := \sum_{a,b} D_{t,ia} \mathrm{Cov}_{\boldsymbol{\theta}}[\hat{s}_a(\boldsymbol{x}_t, t), \hat{s}_b(\boldsymbol{x}_{t'}, t')] D_{t',bj}\ , \tag{48}$$

or equivalently $\boldsymbol{V}(\boldsymbol{x}_t, t; \boldsymbol{x}_{t'}, t') := \boldsymbol{D}_t \mathrm{Cov}_{\boldsymbol{\theta}}\left[\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t), \hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}_{t'}, t')\right] \boldsymbol{D}_{t'}$.

D NAIVE SCORE ESTIMATORS GENERALIZE: DETAILS

In this appendix, we show that integrating the PF-ODE using naive score estimates yields a specific kind of generalization (Prop. 4.1). Suppose that we are integrating the PF-ODE from some initial point $\boldsymbol{x}_T$ using T first-order Euler updates (or some other integration scheme; the choice does not matter in the continuous-time limit), so that

$$\boldsymbol{x}_t = \boldsymbol{x}_{t+1} + \left(\beta_{t+1}\boldsymbol{x}_{t+1} + \boldsymbol{D}_{t+1}\boldsymbol{s}(\boldsymbol{x}_{t+1}, t+1)\right)\Delta t \ . \tag{49}$$

But suppose that we do not have direct access to the score function. Instead, assume that at each time step we draw R samples $\boldsymbol{x}_{0,t+1}^{(r)} \sim p(\boldsymbol{x}_0|\boldsymbol{x}_{t+1}, t+1)$, compute the naive score estimator

$$\hat{\boldsymbol{s}}(\boldsymbol{x}_{t+1}, t+1) = \frac{1}{R}\sum_r \boldsymbol{S}_{t+1}^{-1}(\alpha_{t+1}\boldsymbol{x}_{0,t+1}^{(r)} - \boldsymbol{x}_{t+1}) \ , \tag{50}$$

and use this quantity as the score function for that time step's update. We are interested in studying the extent to which this scheme produces a distribution different from $p_{data}(\boldsymbol{x}_0)$.

Using the result from Appendix C, the typical learned distribution $[q(\boldsymbol{x}_0|\boldsymbol{x}_T)]$ is characterized by the average and V-kernel of $\hat{\boldsymbol{s}}$. If R is somewhat larger than 1, by the central limit theorem $\hat{\boldsymbol{s}}$ is approximately Gaussian, so higher-order terms in the cumulant expansion (Eq. 47) can be neglected.

Since $\hat{\boldsymbol{s}}$ is just an average of (independent) proxy scores, this score estimator is unbiased, i.e., $[\hat{\boldsymbol{s}}] = \boldsymbol{s}$. The V-kernel is

$$\boldsymbol{V}(\boldsymbol{x}_t, t; \boldsymbol{x}_{t'}, t') := \boldsymbol{D}_t \text{Cov}_{\boldsymbol{\theta}}[\hat{\boldsymbol{s}}(\boldsymbol{x}_t, t), \hat{\boldsymbol{s}}(\boldsymbol{x}_{t'}, t')]\boldsymbol{D}_{t'} = \boldsymbol{D}_t \text{Cov}_{\boldsymbol{\theta}}[\hat{\boldsymbol{s}}(\boldsymbol{x}_t, t), \hat{\boldsymbol{s}}(\boldsymbol{x}_t, t)]\boldsymbol{D}_t \ \delta(t-t')$$

since samples generated at different time steps are independent of one another. Moreover,

$$\text{Cov}_{\boldsymbol{\theta}}[\hat{\boldsymbol{s}}(\boldsymbol{x}_t, t)] = \frac{1}{R}\text{Cov}_{\boldsymbol{\theta}}[\tilde{\boldsymbol{s}}(\boldsymbol{x}_t, t)] \tag{51}$$

since the estimator is a sum of independent and identically distributed proxy scores. Finally,

$$\begin{aligned} V_{ij}(\boldsymbol{x}_t, t; \boldsymbol{x}_{t'}, t') &= \frac{1}{R}\sum_{a,b} D_{t,ia}\text{Cov}_{\boldsymbol{\theta}}[\tilde{s}_a(\boldsymbol{x}_t, t), \tilde{s}_b(\boldsymbol{x}_t, t)]D_{t,bj} \ \delta(t-t') \\ &= \frac{1}{R}\sum_{a,b} D_{t,ia}\left[S_{t,ab}^{-1} + \partial_{ab}^2 \log p(\boldsymbol{x}_t|t)\right]D_{t,bj} \ \delta(t-t') \ . \end{aligned} \tag{52}$$

E Linear score estimator: details

In this appendix, we compute the sample-realization-averaged distribution learned by a linear score estimator (Prop. 5.1). Whether it generalizes or not depends strongly on whether the number of features F scales with the number of samples P used during training. First, we must compute the optimum of the DSM objective for a linear model. Then we will determine the average and V-kernel of the optimal linear score estimator.

E.1 Definition of linear score model

Consider a linear score estimator

$$\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t) = \boldsymbol{w}_0 + \boldsymbol{W}\boldsymbol{\phi}(\boldsymbol{x},t) \qquad\qquad \hat{s}_i(\boldsymbol{x},t) = w_{0i} + \sum_{j=1}^{F} W_{ij}\phi_j(\boldsymbol{x},t)\ , \tag{53}$$

where the feature maps $\boldsymbol{\phi} = (\phi_1, ..., \phi_F)^T$ are linearly independent, smooth functions from $\mathbb{R}^D \times [0,T]$ to $\mathbb{R}$ that are square-integrable with respect to the measure $\lambda_t p(\boldsymbol{x},t)$ for all t. The parameters to be estimated are $\boldsymbol{\theta} := \{\boldsymbol{w}_0, \boldsymbol{W}\}$, with $\boldsymbol{w}_0 \in \mathbb{R}^D$ and $\boldsymbol{W} \in \mathbb{R}^{D\times F}$.

E.2 Optimum of DSM objective for linear score model

For this estimator, the DSM objective reads

$$J_1(\boldsymbol{\theta}) = \int \frac{\lambda_t}{2}\, \|\boldsymbol{w}_0 + \boldsymbol{W}\boldsymbol{\phi}(\boldsymbol{x},t) - \tilde{\boldsymbol{s}}(\boldsymbol{x},t;\boldsymbol{x}_0)\|_2^2\; p(\boldsymbol{x}|\boldsymbol{x}_0,t)p_{data}(\boldsymbol{x}_0)p(t)\; d\boldsymbol{x}d\boldsymbol{x}_0dt\ . \tag{54}$$

Note,

$$\frac{\partial \hat{s}_i}{\partial w_{0a}} = \delta_{ia} \qquad\qquad \frac{\partial \hat{s}_i}{\partial W_{ab}} = \delta_{ia}\phi_b\ . \tag{55}$$

Using these to take the gradient of the DSM objective, we have

$$\begin{aligned} \frac{\partial J_1}{\partial w_{0a}} &= \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}_0,t}\left\{\lambda_t\left[w_{0a} + \sum_{j=1}^{F} W_{aj}\phi_j(\boldsymbol{x},t) - \tilde{s}_a(\boldsymbol{x},t;\boldsymbol{x}_0)\right]\right\} \\ \frac{\partial J_1}{\partial W_{ab}} &= \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}_0,t}\left\{\lambda_t\left[w_{0a} + \sum_{j=1}^{F} W_{aj}\phi_j(\boldsymbol{x},t) - \tilde{s}_a(\boldsymbol{x},t;\boldsymbol{x}_0)\right]\phi_b(\boldsymbol{x},t)\right\}\ . \end{aligned} \tag{56}$$

Setting these equal to zero, we have

$$\begin{aligned} \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}_0,t}\left\{\lambda_t\right\} w_{0a} + \sum_{j=1}^{F} W_{aj}\mathbb{E}_{\boldsymbol{x},\boldsymbol{x}_0,t}\left\{\lambda_t\phi_j(\boldsymbol{x},t)\right\} &= \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}_0,t}\left\{\lambda_t\tilde{s}_a(\boldsymbol{x},t;\boldsymbol{x}_0)\right\} \\ \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}_0,t}\left\{\lambda_t\phi_b(\boldsymbol{x},t)\right\} w_{0a} + \sum_{j=1}^{F} W_{aj}\mathbb{E}_{\boldsymbol{x},\boldsymbol{x}_0,t}\left\{\lambda_t\phi_j(\boldsymbol{x},t)\phi_b(\boldsymbol{x},t)\right\} &= \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}_0,t}\left\{\lambda_t\tilde{s}_a(\boldsymbol{x},t;\boldsymbol{x}_0)\phi_b(\boldsymbol{x},t)\right\}\ . \end{aligned} \tag{57}$$

The first row tells us that

$$w_{0a} = \frac{1}{\mathbb{E}_t[\lambda_t]}\mathbb{E}_{\boldsymbol{x},\boldsymbol{x}_0,t}\left\{\lambda_t\tilde{s}_a(\boldsymbol{x},t;\boldsymbol{x}_0)\right\} - \frac{1}{\mathbb{E}_t[\lambda_t]}\sum_{j=1}^{F} W_{aj}\mathbb{E}_{\boldsymbol{x},\boldsymbol{x}_0,t}\left\{\lambda_t\phi_j(\boldsymbol{x},t)\right\}\ , \tag{58}$$

or equivalently that the optimal bias term satisfies $\boldsymbol{w}_0^* = \langle\tilde{\boldsymbol{s}}\rangle - \boldsymbol{W}^*\langle\boldsymbol{\phi}\rangle$, where we have used $\langle\cdots\rangle$ to denote averages with respect to $\lambda_t p(\boldsymbol{x},\boldsymbol{x}_0,t)/\mathbb{E}[\lambda_t]$, and where we have defined the vectors

$$\begin{aligned} \langle\tilde{\boldsymbol{s}}\rangle &:= \frac{\mathbb{E}_{\boldsymbol{x},\boldsymbol{x}_0,t}[\lambda_t\tilde{\boldsymbol{s}}(\boldsymbol{x},t;\boldsymbol{x}_0)]}{\mathbb{E}_t[\lambda_t]} = \frac{1}{\mathbb{E}_t[\lambda_t]}\int \lambda_t\ \tilde{\boldsymbol{s}}(\boldsymbol{x},t;\boldsymbol{x}_0)\ p(\boldsymbol{x},\boldsymbol{x}_0,t)\ d\boldsymbol{x}d\boldsymbol{x}_0dt \\ \langle\boldsymbol{\phi}\rangle &:= \frac{\mathbb{E}_{\boldsymbol{x},t}[\lambda_t\boldsymbol{\phi}(\boldsymbol{x},t)]}{\mathbb{E}_t[\lambda_t]} = \frac{1}{\mathbb{E}_t[\lambda_t]}\int \lambda_t\ \boldsymbol{\phi}(\boldsymbol{x},t)\ p(\boldsymbol{x},t)\ d\boldsymbol{x}dt\ . \end{aligned} \tag{59}$$

Using the first row result, the second row can be written as

$$\langle \phi_b \rangle \left[\langle \tilde{s}_a \rangle - \sum_{j=1}^{F} W_{aj} \langle \phi_j \rangle \right] + \sum_{j=1}^{F} W_{aj} \frac{\mathbb{E}_{\boldsymbol{x},\boldsymbol{x}_0,t} \left\{ \lambda_t \phi_j(\boldsymbol{x},t) \phi_b(\boldsymbol{x},t) \right\}}{\mathbb{E}_t[\lambda_t]} = \frac{\mathbb{E}_{\boldsymbol{x},\boldsymbol{x}_0,t} \left\{ \lambda_t \tilde{s}_a(\boldsymbol{x},t;\boldsymbol{x}_0) \phi_b(\boldsymbol{x},t) \right\}}{\mathbb{E}_t[\lambda_t]}$$

and hence the second row can be written in terms of matrices

$$\begin{aligned} \boldsymbol{\Sigma}_{\boldsymbol{\phi}} &:= \frac{\mathbb{E}_{\boldsymbol{x},t} \left\{ \lambda_t \left[\boldsymbol{\phi}(\boldsymbol{x},t) - \langle \boldsymbol{\phi} \rangle \right] \left[\boldsymbol{\phi}(\boldsymbol{x},t) - \langle \boldsymbol{\phi} \rangle \right]^T \right\}}{\mathbb{E}_t[\lambda_t]} \\ &= \frac{1}{\mathbb{E}_t[\lambda_t]} \int \lambda_t \left[\boldsymbol{\phi}(\boldsymbol{x},t) - \langle \boldsymbol{\phi} \rangle \right] \left[\boldsymbol{\phi}(\boldsymbol{x},t) - \langle \boldsymbol{\phi} \rangle \right]^T p(\boldsymbol{x},t) \, d\boldsymbol{x} dt \\ \boldsymbol{J} &:= - \frac{\mathbb{E}_{\boldsymbol{x},\boldsymbol{x}_0,t} \left\{ \lambda_t \left[\boldsymbol{\phi}(\boldsymbol{x},t) - \langle \boldsymbol{\phi} \rangle \right] \left[\tilde{\boldsymbol{s}}(\boldsymbol{x},t;\boldsymbol{x}_0) - \langle \tilde{\boldsymbol{s}} \rangle \right]^T \right\}}{\mathbb{E}_t[\lambda_t]} \\ &= - \frac{1}{\mathbb{E}_t[\lambda_t]} \int \lambda_t \left[\boldsymbol{\phi}(\boldsymbol{x},t) - \langle \boldsymbol{\phi} \rangle \right] \left[\tilde{\boldsymbol{s}}(\boldsymbol{x},t;\boldsymbol{x}_0) - \langle \tilde{\boldsymbol{s}} \rangle \right]^T p(\boldsymbol{x},\boldsymbol{x}_0,t) \, d\boldsymbol{x} d\boldsymbol{x}_0 dt \, . \end{aligned} \tag{60}$$

In particular,

$$\boldsymbol{W}^* \boldsymbol{\Sigma}_{\boldsymbol{\phi}} = -\boldsymbol{J}^T \implies \boldsymbol{W}^* = -\boldsymbol{J}^T \boldsymbol{\Sigma}_{\boldsymbol{\phi}}^{-1} \, , \tag{61}$$

where we have assumed that $\boldsymbol{\Sigma}_{\boldsymbol{\phi}}$ is invertible. This ought to be true, since the feature maps are independent and $p(\boldsymbol{x}|t)$ is a smooth distribution supported on all of $\mathbb{R}^D$ (especially since we are technically only considering t as small as ϵ, the nonzero lower bound, for regularization purposes).

The optimal score is

$$\hat{\boldsymbol{s}}_*(\boldsymbol{x},t) = \boldsymbol{w}_0^* + \boldsymbol{W}^* \boldsymbol{\phi}(\boldsymbol{x},t) = \boldsymbol{J}^T \boldsymbol{\Sigma}_{\boldsymbol{\phi}}^{-1} \left[\, \langle \boldsymbol{\phi} \rangle - \boldsymbol{\phi}(\boldsymbol{x},t) \, \right] + \langle \tilde{\boldsymbol{s}} \rangle \, . \tag{62}$$

As a side comment, omitting the bias term just removes the mean corrections from the definitions of $\boldsymbol{J}$ and $\boldsymbol{\Sigma}_{\boldsymbol{\phi}}$, as well as the $\langle \boldsymbol{\phi} \rangle$ and $\langle \tilde{\boldsymbol{s}} \rangle$ offsets. Without it, the optimal score is $\hat{\boldsymbol{s}}_*(\boldsymbol{x},t) = \boldsymbol{W}^* \boldsymbol{\phi}(\boldsymbol{x},t) = -\boldsymbol{J}^T \boldsymbol{\Sigma}_{\boldsymbol{\phi}}^{-1} \boldsymbol{\phi}(\boldsymbol{x},t)$, where $\boldsymbol{J}$ and $\boldsymbol{\Sigma}_{\boldsymbol{\phi}}$ are instead defined to be

$$\begin{aligned} \boldsymbol{\Sigma}_{\boldsymbol{\phi}} &:= \frac{\mathbb{E}_{\boldsymbol{x},t} \left\{ \lambda_t \; \boldsymbol{\phi}(\boldsymbol{x},t) \boldsymbol{\phi}(\boldsymbol{x},t)^T \right\}}{\mathbb{E}_t[\lambda_t]} \\ \boldsymbol{J} &:= - \frac{\mathbb{E}_{\boldsymbol{x},\boldsymbol{x}_0,t} \left\{ \lambda_t \; \boldsymbol{\phi}(\boldsymbol{x},t) \tilde{\boldsymbol{s}}(\boldsymbol{x},t;\boldsymbol{x}_0)^T \right\}}{\mathbb{E}_t[\lambda_t]} \, . \end{aligned} \tag{63}$$

In the rest of this appendix, we will assume that the bias term is present.

E.3 Optimum of DSM objective given a finite number of samples

Assume we have access to $P \gg 1$ samples $\boldsymbol{x}^{(n)}, \boldsymbol{x}_0^{(n)}, t^{(n)} \sim p(\boldsymbol{x}, \boldsymbol{x}_0, t)$, and that we estimate the parameters of the linear score model using naive sample mean estimators

$$\begin{aligned} \bar{\lambda}_t &:= \frac{1}{P} \sum_n \lambda^{(n)} \\ \hat{\boldsymbol{b}} &:= \frac{1}{\bar{\lambda}_t} \frac{1}{P} \sum_n \lambda^{(n)} \tilde{\boldsymbol{s}}(\boldsymbol{x}^{(n)}, t^{(n)}; \boldsymbol{x}_0^{(n)}) \\ \hat{\boldsymbol{\mu}}_{\boldsymbol{\phi}} &:= \frac{1}{\bar{\lambda}_t} \frac{1}{P} \sum_n \lambda^{(n)} \boldsymbol{\phi}(\boldsymbol{x}^{(n)}, t^{(n)}) \\ \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\phi}} &:= \frac{1}{\bar{\lambda}_t} \frac{1}{P} \sum_n \lambda^{(n)} \left[\boldsymbol{\phi}(\boldsymbol{x}^{(n)}, t^{(n)}) - \hat{\boldsymbol{\mu}}_{\boldsymbol{\phi}} \right] \left[\boldsymbol{\phi}(\boldsymbol{x}^{(n)}, t^{(n)}) - \hat{\boldsymbol{\mu}}_{\boldsymbol{\phi}} \right]^T \\ \hat{\boldsymbol{J}} &:= -\frac{1}{\bar{\lambda}_t} \frac{1}{P} \sum_n \lambda^{(n)} \left[\boldsymbol{\phi}(\boldsymbol{x}^{(n)}, t^{(n)}) - \hat{\boldsymbol{\mu}}_{\boldsymbol{\phi}} \right] \left[\tilde{\boldsymbol{s}}(\boldsymbol{x}^{(n)}, t^{(n)}; \boldsymbol{x}_0^{(n)}) - \hat{\boldsymbol{b}} \right]^T \end{aligned} \tag{64}$$

where we have used $\lambda^{(n)}$ as a slightly less cumbersome shorthand for $\lambda_{t^{(n)}}$. We will not worry about using Bessel's correction in the covariance estimators, and we will see below that $\hat{\boldsymbol{s}}$ is actually unbiased for finite P even if the covariance estimators are not. Our learned score estimator is then

$$\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t) = \hat{\boldsymbol{J}}^T \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\phi}}^{-1} \left[\, \hat{\boldsymbol{\mu}}_{\boldsymbol{\phi}} - \boldsymbol{\phi}(\boldsymbol{x},t) \, \right] + \hat{\boldsymbol{b}} \, . \tag{65}$$

E.4 Linear score model estimator is unbiased

We are primarily interested in variance due to $\boldsymbol{x}_0$ (for reasons that will become clear), so we will consider an ensemble of systems for which the $\boldsymbol{x}^{(n)}$ and $t^{(n)}$ sample draws are the same, but the $\boldsymbol{x}_0^{(n)}$ draws are different. Our estimator depends linearly on $\tilde{\boldsymbol{s}}$, the quantity through which it depends on the $\boldsymbol{x}_0$ samples. In particular,

$$\begin{aligned}\hat{\boldsymbol{J}}^T \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\phi}}^{-1} [\hat{\boldsymbol{\mu}}_{\boldsymbol{\phi}} - \boldsymbol{\phi}(\boldsymbol{x},t)] &= \frac{1}{P}\sum_n \frac{\lambda^{(n)}}{\bar{\lambda}_t}\left[\tilde{\boldsymbol{s}}(\boldsymbol{x}^{(n)},t^{(n)};\boldsymbol{x}_0^{(n)}) - \hat{\boldsymbol{b}}\right]\left[\boldsymbol{\phi}(\boldsymbol{x}^{(n)},t^{(n)}) - \hat{\boldsymbol{\mu}}_{\boldsymbol{\phi}}\right]^T \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\phi}}^{-1}[\boldsymbol{\phi}(\boldsymbol{x},t) - \hat{\boldsymbol{\mu}}_{\boldsymbol{\phi}}] \\ &= \frac{1}{P}\sum_n \frac{\lambda^{(n)}}{\bar{\lambda}_t}\tilde{\boldsymbol{s}}(\boldsymbol{x}^{(n)},t^{(n)};\boldsymbol{x}_0^{(n)})\left[\boldsymbol{\phi}(\boldsymbol{x}^{(n)},t^{(n)}) - \hat{\boldsymbol{\mu}}_{\boldsymbol{\phi}}\right]^T \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\phi}}^{-1}[\boldsymbol{\phi}(\boldsymbol{x},t) - \hat{\boldsymbol{\mu}}_{\boldsymbol{\phi}}] \ ,\end{aligned}$$

so

$$\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t) = \frac{1}{P}\sum_n \frac{\lambda^{(n)}}{\bar{\lambda}_t} Q(\boldsymbol{x}^{(n)},t^{(n)};\boldsymbol{x},t)\ \tilde{\boldsymbol{s}}(\boldsymbol{x}^{(n)},t^{(n)};\boldsymbol{x}_0^{(n)}) \tag{66}$$

where we have defined the kernel function

$$Q(\boldsymbol{x},t;\boldsymbol{x}',t') := 1 + [\boldsymbol{\phi}(\boldsymbol{x},t) - \hat{\boldsymbol{\mu}}]^T \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\phi}}^{-1}[\ \boldsymbol{\phi}(\boldsymbol{x}',t') - \hat{\boldsymbol{\mu}}\ ] \ . \tag{67}$$

To see that this estimator is unbiased (when the model is sufficiently expressive), suppose the true score has the form of our linear estimator, i.e.,

$$\boldsymbol{s}(\boldsymbol{x},t) = \boldsymbol{w}_0^* + \boldsymbol{W}^*\boldsymbol{\phi}(\boldsymbol{x},t) = \boldsymbol{W}^*[\ \boldsymbol{\phi}(\boldsymbol{x},t) - \langle\boldsymbol{\phi}\rangle\ ] \ , \tag{68}$$

where we have used the fact that $\mathbb{E}_{\boldsymbol{x}}[\boldsymbol{s}] = \langle\boldsymbol{s}\rangle = \boldsymbol{0}$. Next, note that

$$\frac{1}{P}\sum_n \frac{\lambda^{(n)}}{\bar{\lambda}_t} Q(\boldsymbol{x}^{(n)},t^{(n)};\boldsymbol{x},t) = 1 \ . \tag{69}$$

Averaging our estimator over $\boldsymbol{x}_0$ sample draws yields

$$\begin{aligned}\mathbb{E}[\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x},t)] &= \frac{1}{P}\sum_n \frac{\lambda^{(n)}}{\bar{\lambda}_t} Q(\boldsymbol{x}^{(n)},t^{(n)};\boldsymbol{x},t)\ \boldsymbol{W}^*[\boldsymbol{\phi}(\boldsymbol{x}^{(n)},t^{(n)}) - \hat{\boldsymbol{\mu}} + \hat{\boldsymbol{\mu}} - \langle\boldsymbol{\phi}\rangle] \\ &= \frac{1}{P}\sum_n \frac{\lambda^{(n)}}{\bar{\lambda}_t} Q(\boldsymbol{x}^{(n)},t^{(n)};\boldsymbol{x},t)\ \boldsymbol{W}^*[\boldsymbol{\phi}(\boldsymbol{x}^{(n)},t^{(n)}) - \hat{\boldsymbol{\mu}}] + \boldsymbol{W}^*(\hat{\boldsymbol{\mu}} - \langle\boldsymbol{\phi}\rangle) \\ &= \frac{1}{P}\sum_n \frac{\lambda^{(n)}}{\bar{\lambda}_t}\ \boldsymbol{W}^*[\boldsymbol{\phi}(\boldsymbol{x}^{(n)},t^{(n)}) - \hat{\boldsymbol{\mu}}]\left[\boldsymbol{\phi}(\boldsymbol{x}^{(n)},t^{(n)}) - \hat{\boldsymbol{\mu}}\right]^T \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\phi}(\boldsymbol{x},t) - \hat{\boldsymbol{\mu}}) + \boldsymbol{W}^*(\hat{\boldsymbol{\mu}} - \langle\boldsymbol{\phi}\rangle) \\ &= \boldsymbol{W}^*(\boldsymbol{\phi}(\boldsymbol{x},t) - \hat{\boldsymbol{\mu}}) + \boldsymbol{W}^*(\hat{\boldsymbol{\mu}} - \langle\boldsymbol{\phi}\rangle) \\ &= \boldsymbol{W}^*(\boldsymbol{\phi}(\boldsymbol{x},t) - \langle\boldsymbol{\phi}\rangle) \ ,\end{aligned}$$

i.e., it is unbiased. What is worth emphasizing is that this is *exactly* true, and does not require taking any kind of large P limit. In other words, as long as P is large enough that $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\phi}}$ is invertible, one recovers the true weights $\boldsymbol{w}_0^*$ and $\boldsymbol{W}^*$, independent of the $\boldsymbol{x}$ and t sample draws. This is why variance due to $\boldsymbol{x}_0$ sample draws matters, and variance due to the other draws does not, at least for this linear model.

E.5 Computing the V-kernel of the linear score model

Computing the V-kernel amounts to computing the covariance of the score model with respect to $\boldsymbol{x}_0$ sample realizations. In the previous section, we computed the mean of our estimator; the covariance calculation will be fairly similar. Note that

$$\begin{aligned}\mathrm{Cov}[\hat{\boldsymbol{s}}(\boldsymbol{z}),\hat{\boldsymbol{s}}(\boldsymbol{z}')] &= \frac{1}{P^2}\sum_{n,m}\frac{\lambda^{(n)}}{\bar{\lambda}_t}\frac{\lambda^{(m)}}{\bar{\lambda}_t} Q(\boldsymbol{z}^{(n)};\boldsymbol{z})Q(\boldsymbol{z}^{(m)};\boldsymbol{z}')\ \mathrm{Cov}[\tilde{\boldsymbol{s}}(\boldsymbol{z}^{(n)};\boldsymbol{x}_0^{(n)}),\tilde{\boldsymbol{s}}(\boldsymbol{z}^{(m)};\boldsymbol{x}_0^{(m)})] \\ &= \frac{1}{P^2}\sum_n \left(\frac{\lambda^{(n)}}{\bar{\lambda}_t}\right)^2 Q(\boldsymbol{z}^{(n)};\boldsymbol{z})Q(\boldsymbol{z}^{(n)};\boldsymbol{z}')\ \mathrm{Cov}[\tilde{\boldsymbol{s}}(\boldsymbol{z}^{(n)};\boldsymbol{x}_0^{(n)})] \ ,\end{aligned}$$

where we have used $\boldsymbol{z}$ as shorthand for $\{\boldsymbol{x}, t\}$, and the fact that $\boldsymbol{x}_0$ sample draws are independent of one another. Now we will invoke the central limit theorem. Using $\boldsymbol{C}(\boldsymbol{z}) := \text{Cov}[\tilde{\boldsymbol{s}}(\boldsymbol{z}; \boldsymbol{x}_0)]$ as shorthand, when P is very large, to leading order in $1/P$ we have

$$\begin{aligned}\text{Cov}[\hat{\boldsymbol{s}}(\boldsymbol{z}), \hat{\boldsymbol{s}}(\boldsymbol{z}')] &\approx \frac{1}{P} \int \left(\frac{\lambda_t}{\bar{\lambda}_t} \right)^2 Q(\boldsymbol{z}''; \boldsymbol{z}) Q(\boldsymbol{z}''; \boldsymbol{z}') \, \boldsymbol{C}(\boldsymbol{z}'') \, p(\boldsymbol{z}'') \, d\boldsymbol{z}'' \\ &\approx \frac{1}{P} \int \left(\frac{\lambda_t}{\mathbb{E}[\lambda_t]} \right)^2 Q(\boldsymbol{z}''; \boldsymbol{z}) Q(\boldsymbol{z}''; \boldsymbol{z}') \, \boldsymbol{C}(\boldsymbol{z}'') \, p(\boldsymbol{z}'') \, d\boldsymbol{z}''\end{aligned} \tag{70}$$

where we replace the estimates $\hat{\boldsymbol{\mu}}$ and $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\phi}}$ that appear in the kernel function with the true quantities, i.e., we redefine Q to be

$$Q(\boldsymbol{x}'', t''; \boldsymbol{x}, t) := 1 + \left[\boldsymbol{\phi}(\boldsymbol{x}'', t'') - \langle \boldsymbol{\phi} \rangle \right]^T \boldsymbol{\Sigma}_{\boldsymbol{\phi}}^{-1} \left[\boldsymbol{\phi}(\boldsymbol{x}, t) - \langle \boldsymbol{\phi} \rangle \right] \ . \tag{71}$$

If the number of features F does *not* scale with the number of samples P, then we are done: in the $P \to \infty$ limit, the score estimator covariance, and hence the V-kernel, approach zero. Alternatively, if the number of features F *does* scale with the number of samples P, a nontrivial result is possible.

The second term of Q, a quadratic form involving the model's feature maps, is the only place in Eq. 70 one can get nontrivial scaling with F. Hence, if we define

$$\tilde{\boldsymbol{C}}_{ij} := \int \frac{\lambda_t^2}{\mathbb{E}[\lambda_t]^2} \left[\boldsymbol{\phi}(\boldsymbol{z}'') - \langle \boldsymbol{\phi} \rangle \right] \left[\boldsymbol{\phi}(\boldsymbol{z}'') - \langle \boldsymbol{\phi} \rangle \right]^T \ C_{ij}(\boldsymbol{z}'') \, p(\boldsymbol{z}'') \, d\boldsymbol{z}'' \ , \tag{72}$$

and the limit

$$\lim_{P \to \infty} \left[\boldsymbol{\phi}(\boldsymbol{z}) - \langle \boldsymbol{\phi} \rangle \right]^T \boldsymbol{\Sigma}_{\boldsymbol{\phi}}^{-1} \tilde{\boldsymbol{C}}_{ij} \boldsymbol{\Sigma}_{\boldsymbol{\phi}}^{-1} \left[\boldsymbol{\phi}(\boldsymbol{z}') - \langle \boldsymbol{\phi} \rangle \right] \tag{73}$$

exists and is finite, then the asymptotic V-kernel is

$$V_{ij}(\boldsymbol{z}; \boldsymbol{z}') = \lim_{P \to \infty} \frac{1}{P} \sum_{a,b} D_{t,ia} [\boldsymbol{\phi}(\boldsymbol{z}) - \langle \boldsymbol{\phi} \rangle]^T \boldsymbol{\Sigma}_{\boldsymbol{\phi}}^{-1} \tilde{\boldsymbol{C}}_{ab} \boldsymbol{\Sigma}_{\boldsymbol{\phi}}^{-1} [\boldsymbol{\phi}(\boldsymbol{z}') - \langle \boldsymbol{\phi} \rangle] \ D_{t',bj} \ . \tag{74}$$

Note also that, in the large P limit, the V-kernel *also* does not depend on the $\boldsymbol{x}$ and t sample draws.

F Neural network score estimator in NTK regime: details

In this appendix, we prove Prop. 5.2, which means computing the V-kernel of a fully-connected, infinite-width neural network in the 'lazy' learning (Chizat et al., 2019) regime. Although we focus on an extremely specific type of network here, note that our argument can be straightforwardly adapted to compute the V-kernel of other architectures with NTK limits, like convolutional neural networks (Arora et al., 2019).

F.1 Definition of neural network model

Consider a neural network score function approximator $\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{x}, t)$ trained on the DSM objective (Eq. 4). As elsewhere, we may use $\boldsymbol{z}$ as shorthand for $\{\boldsymbol{x}, t\}$. For concreteness, assume that the network is fully-connected, has $L \geq 1$ layers and N_* trainable parameters, and that each hidden layer has N neurons and an identical pointwise nonlinearity G:

$$\begin{aligned} a_i^{(0)}(\boldsymbol{z}) &:= \psi_i(\boldsymbol{z}) \\ a_i^{(\ell+1)}(\boldsymbol{z}) &:= G\left(\frac{1}{\sqrt{N}} \sum_j W_{ij}^{(\ell+1)} a_j^{(\ell)}(\boldsymbol{z}) \right) \quad \ell = 0, ..., L-2 \\ a_i^{(L)}(\boldsymbol{z}) &:= \frac{1}{\sqrt{N}} \sum_j W_{ij}^{(L)} a_j^{(L-1)}(\boldsymbol{z}) \qquad\qquad \hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{z}) := \boldsymbol{a}^{(L)}(\boldsymbol{z}) \ . \end{aligned} \tag{75}$$

The (non-trainable) initial feature maps $\boldsymbol{\psi} := (\psi_1, ..., \psi_{N_0})^T$ account for various preconditioning-related choices. For example, in practice, diffusion models receive time/noise as input only through some time/noise embedding (Ho et al., 2020; Song et al., 2021; Karras et al., 2022).

Although characterizing the gradient descent dynamics of $\hat{\boldsymbol{s}}$ may be difficult in general, if the initial network weights are sampled i.i.d. from a standard normal (i.e., $W_{ij}^{(\ell)} \sim \mathcal{N}(0, 1)$ for all i, j, and ℓ), as N is taken to infinity the network output becomes independent of the precise values of the initial weights. Moreover, the network's output throughout training can be written in terms of a kernel function—the so-called NTK—defined by

$$K^{cc'}(\boldsymbol{z}, \boldsymbol{z}') := \sum_i \mathbb{E}_{\boldsymbol{\theta}} \left\{ \frac{\partial \hat{s}_c(\boldsymbol{z})}{\partial \theta_i} \frac{\partial \hat{s}_{c'}(\boldsymbol{z}')}{\partial \theta_i} \right\} \tag{76}$$

where c and c' index different network outputs. In the infinite-width ($N \to \infty$) limit, $K^{cc'}(\boldsymbol{z}, \boldsymbol{z}') = \delta_{cc'} K(\boldsymbol{z}, \boldsymbol{z}')$, i.e., the off-diagonal kernels are identically zero and all kernels along the diagonal are the same (Shan & Bordelon, 2022).

F.2 Learned score after full-batch gradient descent

Computing the learned score. For simplicity, we assume that our neural network model is trained via full-batch gradient descent on P samples from $p(\boldsymbol{x}, \boldsymbol{x}_0, t)$. Although this assumption does not reflect standard practice (Song et al., 2021; Karras et al., 2022), it makes our computation substantially easier. If we let the dimensionless parameter τ denote training time, the output evolves via

$$\frac{d}{d\tau} \hat{\boldsymbol{s}}(\boldsymbol{x}', t') = \mathbb{E}_{\boldsymbol{x}, t, \boldsymbol{x}_0} \left\{ \frac{\lambda_t}{\mathbb{E}[\lambda_t]} \frac{\partial \hat{\boldsymbol{s}}(\boldsymbol{x}', t')}{\partial \boldsymbol{\theta}} \frac{\partial \hat{\boldsymbol{s}}(\boldsymbol{x}, t)^T}{\partial \boldsymbol{\theta}} \left[\tilde{\boldsymbol{s}}(\boldsymbol{x}, t; \boldsymbol{x}_0) - \hat{\boldsymbol{s}}(\boldsymbol{x}, t) \right] \right\} \ . \tag{77}$$

In the infinite-width limit, we can replace the outer product that appears with the NTK:

$$\frac{d}{d\tau} \hat{\boldsymbol{s}}(\boldsymbol{x}', t') = \mathbb{E}_{\boldsymbol{x}, t, \boldsymbol{x}_0} \left\{ \frac{\lambda_t}{\mathbb{E}[\lambda_t]} K(\boldsymbol{x}', t'; \boldsymbol{x}, t) \left[\tilde{\boldsymbol{s}}(\boldsymbol{x}, t; \boldsymbol{x}_0) - \hat{\boldsymbol{s}}(\boldsymbol{x}, t) \right] \right\} \ . \tag{78}$$

Define the Gram matrix $\boldsymbol{K} \in \mathbb{R}^{P \times P}$, the time-weighting matrix $\boldsymbol{\Lambda}_T \in \mathbb{R}^{P \times P}$, the target matrix $\tilde{\boldsymbol{S}} \in \mathbb{R}^{P \times D}$, and the output matrix $\hat{\boldsymbol{S}} \in \mathbb{R}^{P \times D}$ via

$$\begin{aligned} K_{ab} &:= K(\boldsymbol{x}^{(a)}, t^{(a)}; \boldsymbol{x}^{(b)}, t^{(b)}) \\ \Lambda_{T,ab} &:= \delta_{ab} \lambda_{t^{(a)}} / \mathbb{E}[\lambda_t] \\ \tilde{S}_{ai} &:= \tilde{s}_i(\boldsymbol{x}^{(a)}, t^{(a)}; \boldsymbol{x}_0^{(a)}) \\ \hat{S}_{ai} &:= \hat{s}_i(\boldsymbol{x}^{(a)}, t^{(a)}) \ . \end{aligned} \tag{79}$$

Eq. 78 implies that

$$\frac{d}{d\tau}\hat{\boldsymbol{S}} = \frac{1}{P}\boldsymbol{K}\boldsymbol{\Lambda}_T\left(\tilde{\boldsymbol{S}} - \hat{\boldsymbol{S}}\right) \ . \tag{80}$$

Hence, after training, the network's output on the set of samples is given by

$$\hat{\boldsymbol{S}} = e^{-\boldsymbol{K}\boldsymbol{\Lambda}_T\tau/P}\hat{\boldsymbol{S}}_0 + (\boldsymbol{I} - e^{-\boldsymbol{K}\boldsymbol{\Lambda}_T\tau/P})\tilde{\boldsymbol{S}} \tag{81}$$

where τ is the total training 'time' and $\hat{\boldsymbol{S}}_0$ is the $P \times D$ matrix containing the network's initial output on the samples. Let $\boldsymbol{k}(\boldsymbol{x}, t)$ denote the P-dimensional vector whose i-th component is $K(\boldsymbol{x}^{(i)}, t^{(i)}; \boldsymbol{x}, t)$. The network's output given other inputs evolves according to the ODE

$$\frac{d}{d\tau}\hat{\boldsymbol{s}}(\boldsymbol{x}, t)^T = \frac{1}{P}\boldsymbol{k}(\boldsymbol{x}, t)^T\boldsymbol{\Lambda}_T\left(\tilde{\boldsymbol{S}} - \hat{\boldsymbol{S}}\right) \ , \tag{82}$$

whose solution is

$$\hat{\boldsymbol{s}}(\boldsymbol{x}, t)^T = \hat{\boldsymbol{s}}_0(\boldsymbol{x}, t)^T + \boldsymbol{k}(\boldsymbol{x}, t)^T\boldsymbol{K}^{-1}(\boldsymbol{I} - e^{-\boldsymbol{K}\boldsymbol{\Lambda}_T\tau/P})(\tilde{\boldsymbol{S}} - \hat{\boldsymbol{S}}_0) \tag{83}$$

where $\hat{\boldsymbol{s}}_0(\boldsymbol{x}, t)$ is the network's initial output given a $\{\boldsymbol{x}, t\}$ input. If the Gram matrix $\boldsymbol{K}$ is rank-deficient, we must use its Moore-Penrose pseudoinverse. Alternatively, one can avoid this issue by including a weight regularization term in the objective.

Expressing the learned score in terms of eigenfunctions. We will find it useful to consider a Mercer decomposition of K with respect to the measure $\lambda_t p(\boldsymbol{x}, t)/\mathbb{E}[\lambda_t]$, so that K can be written

$$K(\boldsymbol{x}, t; \boldsymbol{x}', t') = \sum_k \lambda_k \phi_k(\boldsymbol{x}, t)\phi_k(\boldsymbol{x}', t') \tag{84}$$

where the features are orthonormal and complete, i.e.,

$$\begin{aligned}\int \frac{\lambda_t}{\mathbb{E}[\lambda_t]}\phi_k(\boldsymbol{x}, t)\phi_{k'}(\boldsymbol{x}, t)\ p(\boldsymbol{x}, t)\ d\boldsymbol{x}dt = \delta_{k,k'} \\ \sum_k \frac{\lambda_t}{\mathbb{E}[\lambda_t]}p(\boldsymbol{x}, t)\ \phi_k(\boldsymbol{x}, t)\phi_k(\boldsymbol{x}', t') = \delta(\boldsymbol{x} - \boldsymbol{x}')\delta(t - t') \ .\end{aligned} \tag{85}$$

If we assume $\boldsymbol{K}$ has rank F not necessarily equal to P, we can write Eq. 83 in terms of the eigenfunctions associated with the Mercer decomposition by defining the $P \times F$ matrix $\boldsymbol{\Phi}$ with

$$\Phi_{ak} := \phi_k(\boldsymbol{x}^{(a)}, t^{(a)}) \tag{86}$$

and noting that $\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Lambda}\boldsymbol{\Phi}^T$, where $\boldsymbol{\Lambda}$ is the $F \times F$ diagonal matrix of associated eigenvalues. It is useful to observe that

$$\delta_{kk'} = \mathbb{E}_{\boldsymbol{x},t}\left\{\frac{\lambda_t}{\mathbb{E}[\lambda_t]}\phi_k(\boldsymbol{x}, t)\phi_{k'}(\boldsymbol{x}, t)\right\} = \frac{1}{P}\sum_n \frac{\lambda^{(n)}}{\mathbb{E}[\lambda_t]}\phi_k(\boldsymbol{x}^{(n)}, t^{(n)})\phi_{k'}(\boldsymbol{x}^{(n)}, t^{(n)}) + \mathcal{O}(1/\sqrt{P}) \ ,$$

which implies $\boldsymbol{I}_F = \frac{\boldsymbol{\Phi}^T\boldsymbol{\Lambda}_T\boldsymbol{\Phi}}{P}$ to leading order. Similarly, the completeness relation becomes

$$\boldsymbol{I}_P \approx \frac{\boldsymbol{\Phi}\boldsymbol{\Phi}^T\boldsymbol{\Lambda}_T}{P} = \frac{\boldsymbol{\Lambda}_T\boldsymbol{\Phi}\boldsymbol{\Phi}^T}{P} \tag{87}$$

to leading order. Using these identities, we can rewrite Eq. 83 as

$$\begin{aligned}\hat{\boldsymbol{s}}(\boldsymbol{x}, t)^T &= \hat{\boldsymbol{s}}_0(\boldsymbol{x}, t)^T + \left[\boldsymbol{\phi}(\boldsymbol{x}, t)^T\boldsymbol{\Lambda}\boldsymbol{\Phi}^T\right]\left[\frac{\boldsymbol{\Lambda}_T\boldsymbol{\Phi}\boldsymbol{\Lambda}^{-1}\boldsymbol{\Phi}^T\boldsymbol{\Lambda}_T}{P^2}\right]\left[\frac{\boldsymbol{\Phi}}{P}(\boldsymbol{I} - e^{-\boldsymbol{\Lambda}\tau})\boldsymbol{\Phi}^T\boldsymbol{\Lambda}_T\right](\tilde{\boldsymbol{S}} - \hat{\boldsymbol{S}}_0) \\ &= \hat{\boldsymbol{s}}_0(\boldsymbol{x}, t)^T + \frac{1}{P}\boldsymbol{\phi}(\boldsymbol{x}, t)^T(\boldsymbol{I} - e^{-\boldsymbol{\Lambda}\tau})\boldsymbol{\Phi}^T\boldsymbol{\Lambda}_T(\tilde{\boldsymbol{S}} - \hat{\boldsymbol{S}}_0) \ .\end{aligned} \tag{88}$$

Equivalently,

$$\hat{\boldsymbol{s}}(\boldsymbol{x}, t) = \hat{\boldsymbol{s}}_0(\boldsymbol{x}, t) + \frac{1}{P}(\tilde{\boldsymbol{S}} - \hat{\boldsymbol{S}}_0)^T\boldsymbol{\Lambda}_T\boldsymbol{\Phi}(\boldsymbol{I} - e^{-\boldsymbol{\Lambda}\tau})\boldsymbol{\phi}(\boldsymbol{x}, t) \ . \tag{89}$$

Let $\boldsymbol{S}$ denote the $P \times D$ matrix whose entries are the true score evaluated on the set of input samples $\{(\boldsymbol{x}^{(a)}, t^{(a)})\}$. When averaged over $\boldsymbol{x}_0$ sample realizations, our estimator is

$$\mathbb{E}[\hat{\boldsymbol{s}}(\boldsymbol{x}, t)] = \hat{\boldsymbol{s}}_0(\boldsymbol{x}, t) + \frac{1}{P}(\boldsymbol{S} - \hat{\boldsymbol{S}}_0)^T \boldsymbol{\Lambda}_T \boldsymbol{\Phi}(\boldsymbol{I} - e^{-\boldsymbol{\Lambda}\tau})\boldsymbol{\phi}(\boldsymbol{x}, t) \tag{90}$$

which implies

$$\hat{\boldsymbol{s}}(\boldsymbol{x}, t) - \mathbb{E}[\hat{\boldsymbol{s}}(\boldsymbol{x}, t)] = \frac{1}{P}(\tilde{\boldsymbol{S}} - \boldsymbol{S})^T \boldsymbol{\Lambda}_T \boldsymbol{\Phi}(\boldsymbol{I} - e^{-\boldsymbol{\Lambda}\tau})\boldsymbol{\phi}(\boldsymbol{x}, t) \ . \tag{91}$$

To make things slightly easier, define the kernel

$$Q(\boldsymbol{x}, t; \boldsymbol{x}', t') := \sum_{k=1}^{F} \phi_k(\boldsymbol{x}, t)(1 - e^{-\lambda_k \tau})\phi_k(\boldsymbol{x}', t') \ . \tag{92}$$

In terms of this kernel, we can write

$$\hat{\boldsymbol{s}}(\boldsymbol{x}, t) - \mathbb{E}[\hat{\boldsymbol{s}}(\boldsymbol{x}, t)] = \frac{1}{P}\sum_{n} \frac{\lambda^{(n)}}{\mathbb{E}[\lambda_t]} \left[\tilde{\boldsymbol{s}}(\boldsymbol{x}^{(n)}, t^{(n)}; \boldsymbol{x}_0^{(n)}) - \boldsymbol{s}(\boldsymbol{x}^{(n)}, t^{(n)})\right] Q(\boldsymbol{x}^{(n)}, t^{(n)}; \boldsymbol{x}, t) \ . \tag{93}$$

We will use this result in the next subsection to compute the V-kernel of this model.

F.3 Computing the V-kernel of the NTK model

The covariance of the learned score estimator with respect to $\boldsymbol{x}_0$ sample realizations is

$$\begin{aligned}
\mathrm{Cov}[\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{z}), \hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\boldsymbol{z}')] &= \frac{1}{P^2}\sum_{n,m} \frac{\lambda^{(n)}}{\mathbb{E}[\lambda_t]} \frac{\lambda^{(m)}}{\mathbb{E}[\lambda_t]} \mathrm{Cov}\left[\tilde{\boldsymbol{s}}(\boldsymbol{z}^{(n)}; \boldsymbol{x}_0^{(n)}), \tilde{\boldsymbol{s}}(\boldsymbol{z}^{(m)}; \boldsymbol{x}_0^{(m)})\right] Q(\boldsymbol{z}^{(n)}; \boldsymbol{z}) Q(\boldsymbol{z}^{(m)}; \boldsymbol{z}') \\
&= \frac{1}{P^2}\sum_{n} \left(\frac{\lambda^{(n)}}{\mathbb{E}[\lambda_t]}\right)^2 \mathrm{Cov}\left[\tilde{\boldsymbol{s}}(\boldsymbol{z}^{(n)}; \boldsymbol{x}_0^{(n)})\right] Q(\boldsymbol{z}^{(n)}; \boldsymbol{z}) Q(\boldsymbol{z}^{(n)}; \boldsymbol{z}') \\
&= \frac{1}{P}\int \frac{\lambda_{t''}^2}{\mathbb{E}[\lambda_t]^2} \boldsymbol{C}(\boldsymbol{z}'') Q(\boldsymbol{z}''; \boldsymbol{z}) Q(\boldsymbol{z}''; \boldsymbol{z}') \, p(\boldsymbol{z}'') \, d\boldsymbol{z}'' \ ,
\end{aligned}$$

when P is large, where we exploited the independence of the samples in the first step, and the central limit theorem in the second. As elsewhere, we have used $\boldsymbol{C}(\boldsymbol{z}) := \mathrm{Cov}\left[\tilde{\boldsymbol{s}}(\boldsymbol{z}; \boldsymbol{x}_0)\right]$ as shorthand.

We can rewrite this in a form similar to our result for linear models (c.f. Prop. 5.1). Note,

$$\begin{aligned}
\mathrm{Cov}[\hat{s}_i(\boldsymbol{z}), \hat{s}_j(\boldsymbol{z}')] = &\ \frac{1}{P}\boldsymbol{\phi}(\boldsymbol{z})^T(\boldsymbol{I}_F - e^{-\boldsymbol{\Lambda}\tau})\tilde{\boldsymbol{C}}_{ij}(\boldsymbol{I}_F - e^{-\boldsymbol{\Lambda}\tau})\boldsymbol{\phi}(\boldsymbol{z}') \\
\tilde{\boldsymbol{C}}_{ij} := &\int \frac{\lambda_{t''}^2}{\mathbb{E}[\lambda_t]^2} \, \boldsymbol{\phi}(\boldsymbol{z}'')\boldsymbol{\phi}(\boldsymbol{z}'')^T C_{ij}(\boldsymbol{z}'') \, p(\boldsymbol{z}'') \, d\boldsymbol{z}'' \ .
\end{aligned} \tag{94}$$

Hence, the V-kernel is

$$V_{ij}(\boldsymbol{z}; \boldsymbol{z}') = \lim_{P \to \infty} \frac{1}{P}\sum_{a,b} D_{t,ia}\boldsymbol{\phi}(\boldsymbol{z})^T(\boldsymbol{I}_F - e^{-\boldsymbol{\Lambda}\tau})\tilde{\boldsymbol{C}}_{ab}(\boldsymbol{I}_F - e^{-\boldsymbol{\Lambda}\tau})\boldsymbol{\phi}(\boldsymbol{z}')D_{t',bj} \ . \tag{95}$$

provided that the limit exists and is finite.

The infinite training time limit is of particular interest, since in this limit we expect the model to interpolate all of its (noisy) samples. In this limit, we have

$$\begin{aligned}
\mathrm{Cov}[\hat{s}_i(\boldsymbol{z}), \hat{s}_j(\boldsymbol{z}')] = &\ \frac{1}{P}\int \frac{\lambda_{t''}^2}{\mathbb{E}[\lambda_t]^2} \, \boldsymbol{\phi}(\boldsymbol{z})^T\boldsymbol{\phi}(\boldsymbol{z}'')\boldsymbol{\phi}(\boldsymbol{z}'')^T\boldsymbol{\phi}(\boldsymbol{z}')C_{ij}(\boldsymbol{z}'') \, p(\boldsymbol{z}'') \, d\boldsymbol{z}'' \\
= &\ \frac{1}{P}\int \frac{\lambda_{t''}}{\mathbb{E}[\lambda_t]} \, \boldsymbol{\phi}(\boldsymbol{z})^T\boldsymbol{\phi}(\boldsymbol{z}'') \left[\frac{\lambda_{t''}}{\mathbb{E}[\lambda_t]}\boldsymbol{\phi}(\boldsymbol{z}'')^T\boldsymbol{\phi}(\boldsymbol{z}')p(\boldsymbol{z}'')\right] C_{ij}(\boldsymbol{z}'') \ \ d\boldsymbol{z}'' \\
= &\ \frac{1}{P}\int \frac{\lambda_{t''}}{\mathbb{E}[\lambda_t]} \, \boldsymbol{\phi}(\boldsymbol{z})^T\boldsymbol{\phi}(\boldsymbol{z}'')\delta(\boldsymbol{z}'' - \boldsymbol{z}')C_{ij}(\boldsymbol{z}'') \ \ d\boldsymbol{z}'' \\
= &\ \frac{1}{P}\frac{\lambda_{t'}}{\mathbb{E}[\lambda_t]} \, \boldsymbol{\phi}(\boldsymbol{z})^T\boldsymbol{\phi}(\boldsymbol{z}')C_{ij}(\boldsymbol{z}')
\end{aligned} \tag{96}$$

where we have exploited the completeness relation. Now we encounter a subtle technical point. Since $F \neq P$, in the $F, P \to \infty$ limit the quantity

$$d(\boldsymbol{z}, \boldsymbol{z}') := \frac{1}{P} \frac{\lambda_{t'}}{\mathbb{E}[\lambda_t]} \boldsymbol{\Phi}(\boldsymbol{z})^T \boldsymbol{\Phi}(\boldsymbol{z}') \tag{97}$$

is not quite equal to the Dirac delta function, but is instead proportional to it. We need to work out the constant of proportionality. To do this, observe that

$$\sum_n d(\boldsymbol{z}^{(n)}, \boldsymbol{z}^{(n)}) = \frac{1}{P} \sum_n \frac{\lambda^{(n)}}{\mathbb{E}[\lambda_t]} \boldsymbol{\Phi}(\boldsymbol{z}^{(n)})^T \boldsymbol{\Phi}(\boldsymbol{z}^{(n)}) \to \sum_{k=1}^{F} \int \frac{\lambda_t}{\mathbb{E}[\lambda_t]} \, \phi_k(\boldsymbol{z}) \phi_k(\boldsymbol{z}) \, p(\boldsymbol{z}) \, d\boldsymbol{z} = F \, .$$

On the other hand, for the Dirac delta function, we would have

$$\sum_n \delta(0) = \frac{P}{\Delta \boldsymbol{z}} \, , \tag{98}$$

where $\Delta \boldsymbol{z}$ is some small bin size. This implies

$$d(\boldsymbol{z}, \boldsymbol{z}') = \frac{F \Delta \boldsymbol{z}}{P} \delta(\boldsymbol{z} - \boldsymbol{z}') \, . \tag{99}$$

If we define $\kappa := (F \Delta \boldsymbol{z})/P$, and assume κ remains constant as both parameters approach infinity, we finally obtain

$$V_{ij}(\boldsymbol{z}; \boldsymbol{z}') = \kappa \sum_{a,b} D_{t,ia} \boldsymbol{C}_{ab} D_{t,bj} \; \delta(\boldsymbol{z} - \boldsymbol{z}') \, . \tag{100}$$