Learning Fractional White Noises in Neural Stochastic Differential Equations

Anh Tong
KAIST
anhtong@kaist.ac.kr

Thanh Nguyen-Tang
Johns Hopkins University
nguyent@cs.jhu.edu

Toan Tran
VinAI Research, Vietnam
v.toantm3@vinai.io

Jaesik Choi
KAIST, INEEJI
jaesik.choi@kaist.ac.kr

Abstract

Differential equations play important roles in modeling complex physical systems. Recent advances present interesting research directions by combining differential equations with neural networks. By including noise, stochastic differential equations (SDEs) allows us to model data with uncertainty and measure imprecision. There are many variants of noises known to exist in many real-world data. For example, previously white noises are idealized and induced by Brownian motions. Nevertheless, there is a lack of machine learning models that can handle such noises. In this paper, we introduce a generalized fractional white noise to existing models and propose an efficient approximation of noise sample paths based on classical integration methods and sparse Gaussian processes. Our experimental results demonstrate that the proposed model can capture noise characteristics such as continuity from various time series data, therefore improving model fittings over existing models. We examine how we can apply our approach to score-based generative models, showing that there exists a case of our generalized noise resulting in a better image generation measure.

1 Introduction

Differential equations center around modeling relations between functions and their rate of change, playing important roles in many dynamical systems in physics, biology and finance. Incorporating such a tool in machine learning not only helps us understand more about machine learning models [42, 66, 44], but also expands the range of applications [31, 12]. Recent work on neural ordinary differential equations (neural ODEs) opens a new direction of connecting differential equations and neural networks [11]. Extensions to stochastic differential equations (SDEs) is proposed for handling noise in data or model uncertainty [83, 84, 54, 46].

Noises in data are often encountered in many real-world cases, and can directly affect model performance. The importance of understanding the characteristics of noise in data has been emphasized in a number of studies including: heteroscedastic noise modeling in time series analysis [82, 57], volatility modeling in financial research and market simulation [40, 14]; the significance of colored noise in neuroscience [20, 7, 4]. Rather than using traditional Brownian noises [5] in neural SDEs, providing a capacity to model more generalized type of noises not only allows this framework to extend to such applications but also retains the expressiveness from the representation powers of neural networks. By learning noises together with the drift functions and diffusion functions of SDEs, the models become more flexible as the drift functions and diffusion functions, in turn, will be adjusted according to the learned noises. Although mathematical research on analyzing general white

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

noise is well-developed [38, 33], it is still challenging to implement practically for machine learning models like [54, 46].

In this paper, we introduce a novel class of white noises defined under a stochastic integral form [38, 33] to existing neural SDE models [83, 84, 54, 46] which only consider noises generated by Wiener process. The most prominent examples of such stochastic processes include fractional Brownian motions (fBm) and multifractional Brownian motions (mBm) which are generalizations of Brownian motions or Wiener processes. The importance of such processes is emphasized in many applications from financial modeling [13, 28], to motion tracking in physical and biophysical systems [29, 35, 41, 17]. Existing techniques on sampling these stochastic processes [18, 65] prohibit models to scale up as they require to keep track of sample paths during SDE solves [54]. To overcome this, we propose an approximation under numerical SDE solvers considering increments at a time. That is, we first establish a discretized version of stochastic integrals to formulate distribution over noise increments. The key approach is that we borrow the idea in sparse Gaussian process literature [74, 37] to decouple the dependency between increments. Therefore, we can reparameterize the increments to obtain an equivalent SDE which is easy to solve. To show the convergence of our approximations to SDE solutions, we study the approximated covariance functions of underlying stochastic processes to establish error bounds using rough path theory [56, 26] and existing sparse Gaussian process bounds [8].

The contributions of this paper are threefold. First, we introduce a new type of neural SDEs with general white noise. Second, we propose an approximation of that white noise, which is inspired by approximation methods in Gaussian process research to obtain equivalent neural SDEs. Finally, we demonstrate empirically that our approach can capture noise characteristics and provide insight of how white noise affect learning score-based generative models as well as observe a setting that improves baseline models.

2 Related work

Differential equations and neural networks [11] presents neural ODEs using numerical method in solving differential equations to learn parameters in neural networks. There are a number of follow-up works aiming to improvements in different aspects including numerical solver and/or training process [45, 43, 24, 90, 61], implicit termination [10], theoretical understanding [22, 60], and extensions to graph neural networks [9]. Moreover, it has seen some applications in normalizing flows [89, 31, 39] as well as in control [21, 59]. Modeling time series data is one of successful practices of such models [12, 91]. Not close but related, work by [77] demonstrates excellent results on image generation tasks. Like SDEs are a stochastic extension of differential equations, neural SDEs are natural extensions of neural ODEs when incorporating uncertainty into models. However, the formulation is not always straightforward. [83, 84] established theoretical aspects emphasizing high expressiveness of a class of neural SDEs while [54, 46] present how to train neural SDEs efficiently. Many advanced tools in stochastic analysis like rough paths [56] are incorporated into differential equations, and therefore empowered machine learning models [49, 48, 64, 63].

SDEs and Gaussian process There are equivalences and connections between Gaussian processes and linear time-invariant SDEs that [80] discusses to a large extent. [73] presents a faster learning Gaussian process using filtering and smoothing methods [79]. Models of this research direction often focus on spatial-temporal data and establish state-of-the-art results. [36, 88] learn differential equations with Gaussian processes instead of neural networks. [19] considers a similar noise to ours as filtered white noises, finding an analytic form of corresponding kernels in the same manner as [73]. Unlike our approach focusing on a generic procedure, this work studies a specific class of kernels. Gaussian processes can be a tool to model posteriors over paths [1] or to match with densities from Fokker-Planck-Kolmogorov equations [75]. [55] devises a variational inference to approximate posterior inspired by work in sparse GP latent variables.

3 Background

3.1 White Noise

Brownian motion has a long history dated back from the observation of pollen in water in [5], then was studied formally by Einstein [23]. In this paper, we focus on a generalization of Brownian

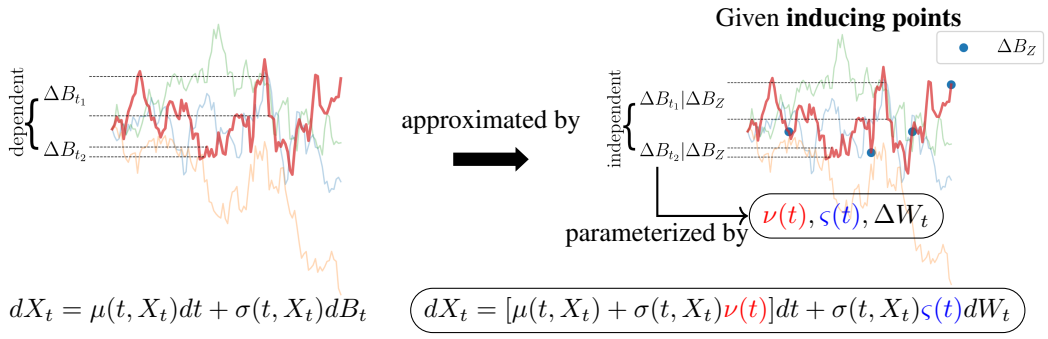


Figure 1: Illustration of sample paths of B_t in neural SDEs. **Left**: Neural SDE with $B_t = \int_0^t f(s,t)dW_s$. Any increments of B_t are correlated. **Right**: Our approximation when a set of inducing points for increments is given, the increments now are pairwise independent and are rewritten as a mean part $\nu(t)$, a variance part $\varsigma(t)$ and an increment of a new Wiener process, ΔW_t. Finally, our approach solves an alternative SDE.

motions under Riemann-Liouville fractional integral defined by:

$$B_t = \frac{1}{\Gamma(h(t) + 1/2)} \int_0^t (t-s)^{h(t)-1/2} dW_s, \tag{1}$$

where $h(t) : \mathbb{R} \to (0, 1)$ is the *Hurst exponent*, W_s is a Brownian motion and $\Gamma(\cdot)$ is the Gamma function. When $h(t)$ is a constant H, this stochastic process resembles a fractional Brownian motion (fBm). However, as noted in [58], it places a great emphasis on the origin ($t = 0$) and no longer possesses power-law spectral density like in fBm. As we are interested in the case that $h(t)$ is a function of t, we emphasize more on the local variation of Hurst exponent than such properties. Our formulation in (1) can be understood as a variant of multifractional Brownian motions (mBm) [69]. One of the goals in this paper is to learn $h(t; \theta)$ using a function approximator, for example, a neural network with parameter θ.

Interpretation of Hurst exponent The stochastic process in (1) asymptotically shares some properties with fBm and mBm [65] including *long-range dependencies* and *self-similarity*. We refer our further analysis related to (1) as fBm or mBm. Here, long-range dependencies are defined for a stochastic process if the sum of autocovariance function is unbounded (see Appendix A.4). Time series analysis has a similar notion for fractional ARIMA models [78]. Deep learning approaches also try to incorporate such long-range dependency to their models [32].

With the Hurst exponent $h(t)$, B_t exhibits a long-range dependence when $h(t) > 1/2$. As $h(t) = 1/2$, the stochastic process falls back to a Brownian motion which no longer has long-range dependency. When $h(t) < 1/2$, the stochastic process is "rough", anti-persistent, having irregularity. This parameter also expresses Hölder continuity of stochastic processes [26]. If $h(t)$ changes over time, this means all of such characteristics happen locally.

Brownian motion, B_t, or Wiener process has a rich history in modeling physical system [67]. Black-Scholes models [40], long-standing models in studying financial markets are based on Brownian motions. However, recent work shows fBm is more suitable due to the irregularity of volatility [14]. Correlated noise [7] also can be found in fMRI data. Applications of fractional Brownian motions can be found in [87] [15].

3.2 Background of Neural SDEs

Consider a dynamical system $\{X_t\}$ whose dynamic evolution is governed by the following stochastic differential equation

$$dX_t = \mu(t, X_t)dt + \sigma(t, X_t)dW_t, \tag{2}$$

where $\mu : [0, T] \times \mathbb{R}^n \to \mathbb{R}^n$ is the drift and $\sigma : [0, T] \times \mathbb{R}^n \to \mathbb{R}^{n \times m}$ is the diffusion. Both of those functions are assumed to be well-behaved (i.e., smooth and linear growth) and are parameterized by neural networks. Readers can refer to [80] for a gentle introduction of SDE in machine learning or to [72] for advanced materials. [83, 84, 54, 46] are excellent starting points to be familiar with neural SDEs.

Back-propagation using optimize-then-discretize strategy This strategy is adopted in neural ODEs, allowing to perform back-propagation in time [11]. Suppose we are interested in optimizing a loss function $\mathcal{L}$ at the terminal state X_T. This approach considers a variable $\partial\mathcal{L}/\partial X_t$. The dynamic of this variable can be modeled by a differential equation leading to computing gradient by using chain rules.

In neural SDEs, such a strategy requires a more careful treatment as [54] addresses the randomness in $\partial\mathcal{L}/\partial X_t$. Also, forward and backward SDEs need to share the same sample paths of Wiener process.

Model learning There are two main parameter estimation methods for Neural SDEs. The first one is latent SDE [54] that follows a variational approach approximating posterior distributions by a neural SDE which shares the same diffusion with priors. In this framework, the evidence lower bounds are derived by applying the Girsanov theorem. Recently, [47] proposes SDE-GAN inspired by generative neural networks [30]. In this paper, we mainly use the former in our experiments. However, it is worth noting that there is no restriction to training with SDE-GAN as long as chosen models satisfy our assumption.

3.3 Sparse Gaussian processes

We also give a brief introduction of sparse Gaussian processes which are known as scalable methods for Gaussian processes. A Gaussian process (GP) [71] is defined by a set of random variables, any finite number of which have a joint Gaussian distribution. Formally, a GP is completely specified by a mean function $m(\cdot)$ which usually is set to be zero for prior and covariance function $k(\cdot,\cdot)$. We then can denote a GP by $f(x) \sim \mathcal{GP}(0, k(x, x'))$. Learning or sampling from a Gaussian process usually takes $\mathcal{O}(N^3)$ with the number of data points N. Sparse Gaussian process methods [74, 37] introduce a set of pseudo points (inducing points) u at locations Z which jointly follow a Gaussian distribution with data points. The density function of $f(x)$, $p(f(x))$ is then approximated by $p(f(x)|u)p(u)$. Here the mean and the variance of $p(f(x)|u)$ can be computed as

$$m(x; u, Z) = \alpha(x)^\top u, \quad v^2(x; Z) = k(x, x) - \alpha(x)^\top k(Z, Z)\alpha(x), \tag{3}$$

where $\alpha(x) = k(Z, Z)^{-1}k(Z, x)$. There is a growing interest in this approach because of its scalability compared to traditional GPs [6, 3, 86, 37]. In this paper, we focus on the sampling procedure where we first sample $u \sim \mathcal{N}(0, k(Z, Z))$, then sample $f(x) \sim p(f(x)|u)$.

4 Learning White Noises in Neural SDEs

This section presents our main model. We first approximate stochastic integrals using Stieltjes approximations and then use sparse Gaussian processes to leverage dependencies between random variables. We analyze the convergence to SDEs solutions for this approximation. Finally, we form an equivalent SDE to our model (see Figure 1).

4.1 Neural SDEs with B_t

Consider a stochastic differential equation

$$dX_t = \mu(t, X_t)dt + \sigma(t, X_t)dB_t, \quad t \in [0, T], \tag{4}$$

with $\mu : [0, T] \times \mathbb{R}^n \to \mathbb{R}^n$ and $\sigma : [0, T] \times \mathbb{R}^n \to \mathbb{R}^{n\times m}$. Compared to (2), the noise coming from B_t is an m-dimensional stochastic process, $B_t = \int_0^t f(t, s)dW_s$ from (1) as we denote $f(t, s) = \frac{1}{\Gamma(h(t)+1/2)}(t - s)^{h(t)-1/2}$.

We are interested in learning both the parameters of $\mu(t, X_t)$, $\sigma(t, X_t)$ and that of $f(t, s)$. Solving SDEs in (4) is challenging since B_t is constructed from an integral. Direct sampling methods of B_t (i.e., require computing Cholesky decomposition) are costly. Other approaches [65] use an approximation of stochastic integral as a linear combination of random variables. However, the number of these random variables prevents optimize-then-discretize methods to scale.

4.2 Integral approximation

We use Stieltjes integration to approximate the integral in (1) by discretizing the interval $[0, T]$ with $0 = s_1 < s_2 < \cdots < s_N = T$, i.e.,

$$B_t = \int_0^t f(t,s)dW_s = \int_0^T \mathbb{1}\{s \in [0,t]\} f(t,s)dW_s \approx \sum_{i=1}^{N-1} \mathbb{1}\{s_i \in [0,t]\} f(t,s_i)(W_{s_{i+1}} - W_{s_i})$$

$$= \sum_{i=1}^{N-1} \underbrace{\mathbb{1}\{s_i \in [0,t]\} f(t,s_i)\sqrt{\Delta s_i}}_{g_i(t)} \xi_i = \sum_{i=1}^{N-1} g_i(t)\xi_i, \tag{5}$$

where ξ_i are independently distributed standard normal random variables coming from $W_{s_{i+1}} - W_{s_i} \sim \mathcal{N}(0, \Delta s_i)$ with $\Delta s_i = s_{i+1} - s_i$, and $\mathbb{1}\{s \in [0,t]\}$ is the indicator function. Note that this is a reasonable approximation because Itô's integrals, in general, converge in L^2 when N goes to infinity. Moreover, even if W_t has infinite variation, the probability of a sample path leading to the divergence of approximation infinitely often is zero [72]. The discretization is also inevitable whenever it is required to sample B_t.

In several numerical approaches, for example, Euler-Maruyama [51], solving stochastic differential equations involves approximating increments, dB_t, by

$$\Delta B_t := B_{t+\Delta t} - B_t \approx \sum_{i=1}^{N-1} (g_i(t+\Delta t) - g_i(t))\xi_i = \sum_{i=1}^{N-1} \Delta g_i(t)\xi_i, \tag{6}$$

where $\Delta g_i(t) = g_i(t+\Delta t) - g_i(t)$. Comparing to increments ΔW_t when solving (2), simulating ΔW_t is relatively simple as they are independent between each time step. On the other hand, there exist dependencies between ΔB_t as

$$\mathbb{E}[\Delta B_t \Delta B_{t'}] \approx \sum_{i=1}^{N-1}\sum_{j=1}^{N-1} g_i(t)g_j(t')\mathbb{E}[\xi_i\xi_j] = \sum_{i=1}^{N-1} g_i(t)g_i(t') \neq 0. \tag{7}$$

Here, $\xi_i, \xi_j \sim \mathcal{N}(0,1)$, and $\mathbb{E}[\xi_i\xi_j] = 0$ when $i \neq j$. Our aim is to represent (6) in a way that is simple to sample from but preserves the dependencies between ΔB_t.

4.3 Sampling increments with sparse Gaussian process

From the observation in (7), we devise a sampling method inspired by sparse Gaussian processes. In particular, we consider a Gaussian process[1] $B'_t = \frac{\Delta B_t}{\Delta t}$ for a fixed Δt. In fact, Δt is later used as the time step of SDE solvers. Note that directly modeling ΔB_t in (6) is numerically unstable when computing (7) with covariance matrices having small values.

Indeed, B'_t is a Gaussian process because at a given time t, B'_t is Gaussian with a zero mean. To compute the covariance function $k(t,s)$, we simply compute $k(t,s) = \mathbb{E}[B'_t B'_s] - \mathbb{E}[B'_t]\mathbb{E}[B'_s] = \mathbb{E}[B'_t B'_s]$ and follow the same step with (7)

$$k(t,s) = \mathbb{E}[B'_t B'_s] = \frac{1}{(\Delta t)^2}\sum_{i=1}^{N-1} g_i(t)g_i(s). \tag{8}$$

To this end, we denote $B'_t \sim \mathcal{GP}(0, k(t,s))$. Sampling from a Gaussian process based on a full covariance matrix is not scalable. To overcome this difficulty, we propose a sampling scheme based on sparse-inducing GP [74]. The main idea of this method is to introduce a bottleneck as a summation of the joint distribution of B'_t.

Define inducing-point locations as $Z = \{z_1, \ldots, z_M\}$ which spread across $[0, T]$. The number of inducing points, M, is much smaller than the number of discretization mesh of numerical solvers. The inducing points $u = (B'_{z_1}, \ldots, B'_{z_M})$ are distributed as a multivariate Gaussian distribution, i.e., $u \sim \mathcal{N}(0, k(Z,Z))$. Following the description of sparse GPs in §3.3, $p(B'_t)$ is approximated by $p(B'_t|u)p(u)$.

The sampling procedure of B'_t consists of two steps:

$$\text{Step 1}: \quad u = L_Z\varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, I_M), \tag{9}$$

$$\text{Step 2}: \quad B'_t = m(t;u,Z) + v(t;Z)\xi_t, \qquad \xi_t \sim \mathcal{N}(0,1). \tag{10}$$

Here L_Z is the Cholesky decomposition of $k(Z,Z)$, I_M is an $M \times M$ identity matrix. $m(\cdot)$ and $v(\cdot)$ are the conditional mean and variance taken from (3).

[1] This can be viewed as the "derivative" of B_t w.r.t. time. However, B_t is known as non-differentiable.

The above is a complete sampling procedure for $p(B'_t|u)p(u)$. Note that in our approximation, B'_t is independent *conditioned* on that u are given. Therefore, we obtain the increment ΔB_t as

$$\Delta B_t = B'_t \Delta t = m(t;u)\Delta t + v(t;u)\Delta t \xi_t = m(t;u,Z)\Delta t + v(t;Z)\sqrt{\Delta t}\Delta W_t. \quad (11)$$

Here, we replace $\sqrt{\Delta t}\xi_t$ with increments of a (new) Brownian motion W_t since ξ_t are independent for $t \in [0,T]$.

Remark One may consider an alternative approach where we use sparse approximation directly for B_t with given kernel function $k(t,s) \propto |t|^{2h_{t,s}} + |s|^{h_{t,s}} - |t-s|^{2h_{t,s}}$ (see (16)), and can obtain a similar form of increments. However, this kernel function is not differentiable w.r.t the parameters of Hurst function. Our approach is to devise a "so-called" differentiation of B_t under discretization. Note that this notion differs from the definition of the derivative of B_t in [52] in the sense of Schwartz space. However, such a representation is not useful for a practical implementation.

Our approach shares a similarity with [49] on controlled differential equations driven by a signal, i.e., X_t. The signal X_t is approximated by linear interpolations which allows to compute the derivative of X_t. We intentionally model the "derivative" of B_t by B'_t first, and then reconstruct the signal B_t from B'_t.

4.4 Forming equivalent SDEs

To this end, we show that the increments ΔB_t can be represented with increments of a Wiener process, ΔW_t in (11). Here, the equivalence is under the assumption that the numerical procedure involves only ΔB_t, i.e., Euler-Maruyama methods. Therefore, the original SDE in (4) is approximated by

$$dX_t = (\mu(t,X_t) + \sigma(t,X_t)\nu(t;u,Z))dt + \sigma(t,X_t)\varsigma(t;Z)dW_t, \quad u \sim \mathcal{N}(0,k(Z,Z)). \quad (12)$$

This is obtained by conversions, $dt \approx \Delta t$ and $dW_t \approx \Delta W_t$ as we expand the diffusion term

$$\begin{aligned}\sigma(t,X_t)\Delta B_t &= \sigma(t,X_t)(m(t;u,Z)\Delta t + v(t;Z)\sqrt{\Delta t}\Delta W_t) \\ &= \sigma(t,X_t)\nu(t;u,Z)\Delta t + \sigma(t,X_t)\varsigma(t;Z)\Delta W_t \\ &\approx \sigma(t,X_t)\nu(t;u,Z)dt + \sigma(t,X_t)\varsigma(t;Z)dW_t.\end{aligned}$$

where $\nu(t;u,Z) = m(t;u,Z)$, and $\varsigma(t;Z) = v(t;Z)\sqrt{\Delta t}$. The function $\varsigma(t;Z)$ induced from the approximation of B'_t is now absorbed into the diffusion function while $\nu(t;u,Z)$ together with the diffusion function now is added to the drift function. Note that $\nu(t;u,Z)$ and $\varsigma(t;Z)$ is conditioned on the inducing point u. Therefore, at the initial time $t=0$, we need to precompute u first.

Although (12) and (4) seem similar in terms of the expressiveness of drift functions and diffusion functions, the drift function of (12) contains a randomness stemming from the sampling u in (9) while that of (4) is deterministic. This can be understood that the dependency between ΔB_t is translated into the drift functions. We can elucidate the difference via Algorithm 1 and Algorithm 2 in Appendix.

Note that $\nu(t;u,Z)$ and $\varsigma(t;Z)$ are associated with two types of discretizations: that of approximating Stieltjes integral and that of SDE numerical solver.

One of the key advantages of this new SDE is that it can inherit all theory and practices of well-established study of stochastic analysis for Brownian motion or Wiener process.The preliminary results of Girsanov's theorem, i.e. [34] cannot used in latent SDE [54]. Itô's rules for fBm and mBm can be found in [52] but hard to use in practice. On the other hand, our approach can work for any B_t defined in (1).

Another benefit of our formulation is that the sampling procedure is simple, requiring only one ξ_t at a time and avoids a multiplication of $\mathcal{O}(N)$ memory like [65]. This becomes significant as the optimize-then-discretize method requires constructing the same sample paths of B_t in forward SDEs and backward SDEs.

Computational analysis The evaluation of diffusion functions in (12) requires computing an aggregate sum over discretization that scales linearly with the number of discretization points. It may cause some overhead when computing Itô correction term during solving backward SDE [54]. Therefore, Stratonovich SDEs are more efficient than Itô SDEs. One limitation of integral approximation is that for a long interval $[0,T]$, one may need more discretized partitions to achieve desired approximation accuracy.

On the computational cost of sparse GPs, we only need to perform the Cholesky decomposition at the initial time of every SDE integration. This takes $\mathcal{O}(M^3)$ with M the number of inducing points. At a time t during solving SDEs, with the pre-computed L_Z, sampling B'_t has $\mathcal{O}(N+M^2)$ time complexity as we perform matrix multiplications in (3).

Limitation Currently, our model only considers fixed step sizes in SDE numerical solvers. Most theoretical studies in stochastic analysis work for $H > 1/4$ ($H > 1/3$ in this study) to make sense of stochastic integrals or SDE solutions [70]. The Euler-Maruyama update in our approach may not be sufficiently good when the Hurst exponent is less than $1/2$ as it is pointed out in [16]. Higher-order solvers can be more accurate, requiring handing double integrals, i.e., $\int\int dB_t dB_s$ which we wish to address in future work. Although this paper presents an initial theoretical justification of convergence to solutions, the precise order of convergence is not clear yet. Therefore, for practical use, we recommend avoiding using Hurst exponents with small values in our method.

4.5 Convergence analysis

The error of our approximation is composed of three main factors. The first is of SDE solvers. Due to the assumption of fixed step sizes, only a few solvers, i.e., Euler-Maruyama, are applicable with $1/2$ strong convergence. Secondly, the discretization in Stieltjes approximations attributing to the total error leads to the convergence in L^2. The convergence of such approximations are known. The approximation of noise from sparse GPs is unconventional in solving SDEs because the approximation is in distributional sense rather than point-wise or path-wise. We will show the convergence using rough path theory [56, 26] and sparse GPs bound from [8] for constant Hurst $H \in (1/3, 1/2)$.

We follow the notation in [39, 56, 26]. Let us denote the canonical lift of B as $\mathbf{B} = (B, \mathbb{B})$. Let $\widehat{B}$ be the path corresponding to the sampling procedure in (9) and (10). Let $\mathbf{X}$ be the solution of $d\mathbf{X}_t = f(t, X_t)d\mathbf{B}_t$ and $\widehat{X}$ be the solution of $d\widehat{X} = f(t, \widehat{X}_t)d\widehat{B}$ where $f : [0, T] \times \mathbb{R}^n \to \mathbb{R}^n$ is a bounded function that is four-times continuously differentiable. We assume that the initial state follows a distribution p_0 which is is continuous and strictly positive in $\mathbb{R}^n$. Let us denote the probability density distribution of the solution X at time t by $p_t(x)$; and that of the approximated solution $\widehat{X}$ by $\widehat{p}_t(x)$.

Theorem 1 (Convergence of solutions given sparse GP approximation)**.** *With the Hurst exponent of B, $H \in (1/3, 1/2)$, assume that $\mathbb{E}[(\widehat{B}_t - \widehat{B}_{t+\tau})^2] \leq L|\tau|^{1/\varrho}$ for any $t \in [0, T]$, and $\varrho \in (1/2H, 3/2]$. Then, for every $\alpha \in (1/3, 1/2\varrho)$, there exists a constant C such that*

$$\mathrm{KL}(\widehat{p}_t \| p_t) \leq C \sup_{t \in [0,T]} \mathrm{KL}(\widehat{B}_t \| B_t)^\theta. \tag{13}$$

Here, $\theta \in (0, 1/2 - \varrho\alpha)$ and $\mathrm{KL}(\cdot\|\cdot)$ is the Kullback-Leibler divergence.

This theorem says that if the distribution of $\widehat{B}$ converges to that of B in terms of the KL divergence, their corresponding solutions converge in the same sense. Note that we make an assumption that $\mathbb{E}[(\widehat{B}_t - \widehat{B}_{t+\tau})^2] \leq L|\tau|^{1/\varrho}, \varrho > 1/2H$ which means that $\widehat{B}$ is less "rough" than B. This is a reasonable assumption since $\widehat{B}$ suffers information loss from sparse GP approximations.

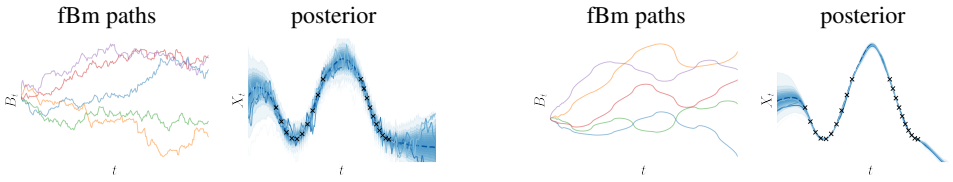
Our proof is done by (1) assessing the p-variation of $\widehat{B}$ to bound the distance in rough path space between the canonical lifts of B and $\widehat{B}$ via [26, Corollary 10.6]; and (2) using the local Lipschitz property of Itô-Lyons maps [26, Theorem 8.5] as we follow the technique in [39]. The complete proof is provided in Appendix B along with necessary backgrounds and discussion.

According to [8], we can quantify the convergence of $\widehat{B}$ to B as its KL divergence can be bounded, $\mathrm{KL}(\widehat{B}\|B) \leq C \sum_{i=M+1}^{\infty} \lambda_i$. Here λ_i are eigenvalues in eigenvalue-eigenfunction pairs $(\lambda_i, \phi_i(\cdot))$ of Mercer's theorem [62], $k(x, x') = \sum_{i=1}^{\infty} \lambda_i \phi_i(x)\phi_i(x')$. Recall M is the number of inducing points. Therefore, the convergence of $\mathrm{KL}(\widehat{B}\|B)$ depends on the decaying rate of eigenvalues and the number of inducing points. We empirically show the fast decaying rate of eigenvalues for finite samples which also can be found in Appendix B.

4.6 Parameterizing white noise

The previous section concerns how to learn white noises in neural SDEs without providing concrete forms of white noise. This section will show how it affects SDEs in general and discuss the parameterization form of the Hurst exponent.

When the Hurst exponent $h(t)$ is constant, say $h(t) = H$. There is a qualitative difference between settings of H. Figure 2 shows how the Hurst parameter affects the model fit trained by Latent SDE. With a smaller value ($H = 0.3$), sample paths are rough, producing more irregularity. When $H = 0.8$, the model has smoother sample paths, and is more conservative as predictions depend on the past. This parameter has a great influence on the uncertainty quantification of the models.



(a) $H = 0.3$ (b) $H = 0.8$

Figure 2: Latent SDE with fBm for two settings: $H = 0.3$ and $H = 0.8$.

For the case that $h(t)$ is function of t, any function approximators can work in general to model $h(t; \theta)$. In our implementation, we use small neural networks. With the new representation in (12), we can train the parameters of $h(t; \theta)$ together with the parameters of drift and diffusion. A more general approach is that $f(t, s)$ can accept either neural networks or kernel-based functions when we do not consider normalization. However, their interpretation is not clear as that of mBm.

5 Experimental Results

This section will first justify how close the samples generated by our method are to true ones. We then investigate the ability to learn the Hurst function for synthetic data and test our model for real-world data. Finally, we examine the role of Hurst exponents in score-based generative models. All experiments are implemented using PyTorch [68] with the library `torchsde` [54, 47]. Our source code is available at this repository.

How to estimate Hurst from data? We follow [27] to empirically estimate Hurst exponent from data. The key technique is based on the correlation of increment of B_t and estimates a ratio over sliding windows which is an estimation of Hurst exponent.

5.1 Quality of approximate samples

We would like to investigate if our approximation of B'_t can produce the sample path of B_t that is close enough to those of the true process. The true Hurst function is set as $h(t) = 0.35 + 0.5t$.The samples from [65, 25] are used as a baseline. This experiment runs 1000 samples. The estimations of the Hurst exponent for two sampling techniques are in Figure 3. The Hurst estimation corresponding to our samples is close to the true Hurst function. It tends to overestimate the long-range dependency ($h(t) > 1/2$) and underestimate the irregularity ($h(t) < 1/2$) in the samples.

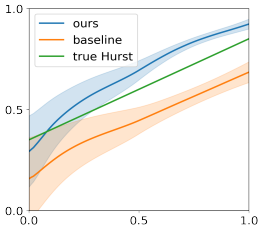


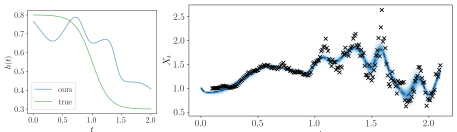
Figure 3: Our estimation vs true Hurst function.

5.2 Recovering Hurst function

Next, we test our model with a synthetic data. Here the data is generated from a SDE with a known solution:

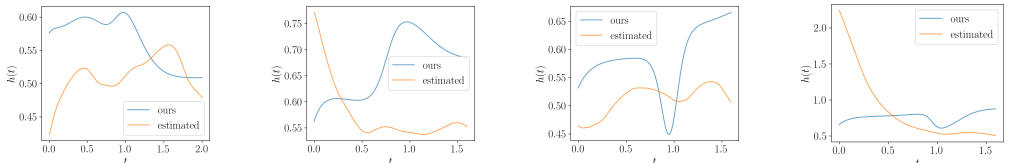
$$dX_t = \alpha X_t dt + \beta dB_t, \quad X_0 = x_0, \quad t \in [0, 2],$$

where B_t is a mBm given a fixed $h(t) = 0.3 + 0.5\texttt{sigmoid}(7(1 - t))$. According to Itô integrals for mBm cases [52], the exact solution is $X_t = x_0 \exp(\alpha t - \frac{1}{2}\beta^2 t^{2h(t)} + \beta B_t)$. We generate 100 data points for each sample path and obtain up to 5 sample paths. After training our model, we observe that the obtained Hurst function can capture the dynamic similar to the true Hurst. As we increase the number of sample paths (from 1 to 5), our learned Hurst tends to approach the true Hurst. Figure 4 shows an example of our learned Hurst function given 5 sample paths. We also provide the results for different Hurst functions and an ablation study in Appendix C.3.



(a) (b)

Figure 4: Synthetic data. (a): Comparison between found Hurst vs. true Hurst. (b): The posterior plot of our model which can describe correctly uncertainty regions when the noise is more irregular.



(a) AMZN (b) AAPL (c) GOOGL (c) MSFT

Figure 5: Found Hurst functions in financial data.

5.3 Financial data

In this experiment, we compare the Hurst function found by our model with the time-dependent Hurst estimation computed by [27].

Data set We collect two months (from September 26th, 2021 to October 23rd, 2021) of hourly data from four stock indices Amazon (AMZN), Apple (APPL), Google (GOOGL), Microsoft (MSFT). The data is retrieved from [85]. It is then standardized and split into the first 80% time points to train and the remaining 20% to test.

Obtaining Hurst function Figure 5 compares our found Hurst functions and estimated Hurst computed from [27]. Our obtained Hurst functions have bigger values than that of [27]. This resembles the observation in the previous experiment. A common observation among these stocks is that Hurst functions made some changes (bump up or drop) around the end of October. This agrees with many analyses, for example, Amazon missed revenue expectations for the third quarter or Apple revealed new products. The posterior plots for our model in each stock index are placed in Appendix C.4.

Quantitative comparison We compare our model to two main baselines: Latent SDE with standard Brownian motion [54] and Latent ODE [12]. We report three main comparison criteria: training negative log-likelihood (training NLL), test negative log-likelihood (test NLL) and test root mean square error (test RMSE). Overall, our model can fit training data well as our model outperforms others in terms of training NLLs (see Appendix C.4). This is because our model can capture correctly noise part while Latent SDE relies on less flexible Brownian noise. Still, it is not clear to identify which model performs better in predictive tasks. However, our model produces more consistent results with less variance.

5.4 Effect of Hurst exponent in score-based generative models

This experiment explores a different aspect by studying the effect of Hurst exponent in training score-based generative models. Recent work [77] on generative models based on score functions and SDEs presents impressive results on par with generative adversarial networks (GAN). We consider the case where the perturbation in data is under a new SDE, $dX_t = -\frac{1}{2}\beta(t)X_t dt + \sqrt{\beta(t)}dB_t$ where $B_t = \int_0^t f(t,s)dW_s$. We derive the equivalent SDE (discussed more in Appendix C.5):

$$dX_t = \left[-\frac{1}{2}\beta(t)\varsigma^2(t;Z)X_t + \sqrt{\beta(t)}\nu(t;u,Z)\right]dt + \sqrt{\beta(t)}\varsigma(t;Z)dW_t, \quad u \sim \mathcal{N}(0, k(Z,Z)). \tag{14}$$

Varying Hurst exponents We investigate how different types of noise affect learning and sampling in this model. We consider four scenarios of noise: (1) more anti-persistent as $H = 0.3$; (2) long-range dependent with $H = 0.8$; (3) linearly increasing $h(t)$; (4) linearly decreasing $h(t)$. We only consider the task of generating MNIST images [53]. Figure 6 shows example of generated images. We observed that $H > 1/2$ is not favorable, and requires SDEs with long terminal time T. To quantify the quality of generated images, we compute the negative log-likelihood (NLL) of a set of out-of-sample images like in [76].

Discussion Although we do not present a way to find optimal $h(t)$, we show that there is a setting in which the NLL appears to be better (see Figure 7 a). That is, the NLL of decreasing Hurst setting is $\mathbf{3.79}$ (bits/dim) compared to 4.97 (bits/dim) of Brownian motion ($H = 0.5$). There exists work [50] showing that signal-to-noise ratio (SNR) is the key to establishing an optimal strategy how to inject noises. There is another work studying the best strategy for reverse variance [2]. In contrast to many diffusion models, our approach has an additional term $\sqrt{\beta(t)}\nu(t;u)$ which is unrelated to X_t but is included in the drift of the backward SDEs when generating data (see Figure 7 b). Moreover, our approach can suggest a way to control the sample path of B_t in the way that we condition on inducing

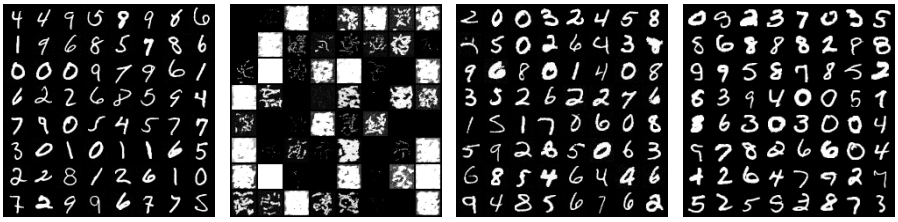
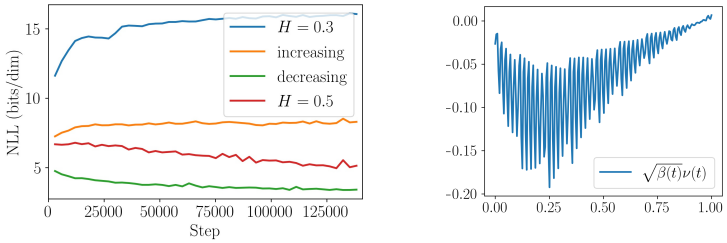


Figure 6: Generated images from different configuration of Hurst.



(a) (b)

Figure 7: (a): Negative log likelihoods on a small batch of test set. (b): Additional drift $\sqrt{\beta(t)}\nu(t; u, Z)$.

points u since B'_t is sampled from the conditional distribution $p(B'_t|u)$. For example, we can follow the variational inference of [37] to find the posterior $q(u)$ for a designed objective, then sample B'_t given $q(u)$ instead of $p(u)$.

6 Conclusion

In conclusion, this work presents a new class of neural SDEs with a generalization of white noise. We propose an efficient approximate sampling approach for the noises so that neural SDEs can be converted into convenient forms. The techniques used in this paper are relatively simple, and related to Gaussian process (GP) research. Our experiments have verified the ability of our models to learn noises from synthetic data to financial time-series as well as showcase an image generation task. As mentioned earlier, our current work can be improved with higher-order solvers where the double integrals require to model precisely. Prior work [52, 70] is a good starting point for such an approach. Another promising direction is to extend neural controlled differential equations with driving signals which are general GPs instead of Brownian motions. The closure property of GPs under differentiation will be useful to develop this model.

Acknowledgements

This work was conducted by Center for Applied Research in Artificial Intelligence (CARAI) grant funded by DAPA and ADD (UD190031RD) and was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2022-0-00984, Artificial Intelligence, Explainability, Personalization, Plug and Play, Universal Explanation Platform), (No. 2019-0-00075, Artificial Intelligence Graduate School Program (KAIST)), (No. 2022-0-00184, Development and Study of AI Technologies to Inexpensively Conform to Evolving Policy on Ethics), and KAIST-NAVER Hypercreative AI Center. In addition, this research was supported, in part, by the DARPA GARD award HR00112020004, NSF CAREER award IIS-1943251, and NSF BIGDATA award IIS-1838139 (to TNT). We would like to thank anonymous reviewers for insightful feedback.

References

[1] Cedric Archambeau, Dan Cornford, Manfred Opper, and John Shawe-Taylor. Gaussian process approximations of stochastic differential equations. In *Gaussian Processes in Practice*, Proceedings of Machine Learning Research, 2007.

[2] Fan Bao, Chongxuan Li, Jun Zhu, and Bo Zhang. Analytic-DPM: an analytic estimate of the optimal reverse variance in diffusion probabilistic models. In *ICLR*, 2022.

[3] Matthias Bauer, Mark van der Wilk, and Carl Edward Rasmussen. Understanding probabilistic sparse Gaussian process approximations. In *NeurIPS*, 2016.

[4] Vikranth R. Bejjanki, Rava Azeredo da Silveira, Jonathan D. Cohen, and Nicholas B. Turk-Browne. Noise correlations in the human brain and their impact on pattern classification. *PLOS Computational Biology*, 2017.

[5] Robert Brown. A brief account of microscopical observations made in the months of June, July and August 1827, on the particles contained in the pollen of plants; and on the general existence of active molecules in organic and inorganic bodies. *The Philosophical Magazine*, 1828.

[6] Thang D. Bui, Josiah Yan, and Richard E. Turner. A unifying framework for Gaussian process pseudo-point approximations using power expectation propagation. *J. Mach. Learn. Res.*, 18, 2017.

[7] E. T. Bullmore, Chris Long, John Suckling, Jalal M. Fadili, Gemma Calvert, Fernando Zelaya, T.Adrian Carpenter, and Mick Brammer. Colored noise and computational inference in neurophysiological (fMRI) time series analysis: Resampling methods in time and wavelet domains. *Human Brain Mapping*, 2001.

[8] David Burt, Carl Edward Rasmussen, and Mark Van Der Wilk. Rates of convergence for sparse variational Gaussian process regression. In *ICML*, 2019.

[9] Ben Chamberlain, James Rowbottom, Maria I Gorinova, Michael Bronstein, Stefan Webb, and Emanuele Rossi. Grand: Graph neural diffusion. In *ICML*, 2021.

[10] Ricky T. Q. Chen, Brandon Amos, and Maximilian Nickel. Learning neural event functions for ordinary differential equations. *ICLR*, 2021.

[11] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. *NeurIPS*, 2018.

[12] Ricky T. Q. Chen, Yulia Rubanova, and David Duvenaud. Latent ODEs for irregularly-sampled time series. In *NeurIPS*, 2019.

[13] Fabienne Comte and Eric Renault. Long memory in continuous-time stochastic volatility models. *Mathematical Finance*, 1998.

[14] Sylvain Corlay, Joachim Lebovits, and Jacques Lévy Véhel. Multifractional Stochastic volatility models. *Mathematical Finance*, 2014.

[15] Donny Danudirdjo and Akira Hirose. Synthesis of two-dimensional fractional Brownian motion via circulant embedding. In *ICIP*, 2011.

[16] A. M. Davie. Differential equations driven by rough paths: an approach via discrete approximation. *Applied Mathematics Research eXpress*, 2008.

[17] Carmine Di Rienzo, Vincenzo Piazza, Enrico Gratton, Fabio Beltram, and Francesco Cardarelli. Probing short-range protein Brownian motion in the cytoplasm of living cells. *Nature Communications*, 2014.

[18] Ton Dieker. *Simulation of fractional Brownian motion*. PhD thesis, 2004.

[19] Matthew Dowling, Piotr Sokół, and Il Memming Park. Hida-matérn kernel, 2021.

[20] J. Andrew Doyle and Alan C. Evans. What colour is neural noise?, 2018.

[21] Jianzhun Du, Joseph Futoma, and Finale Doshi-Velez. Model-based reinforcement learning for semi-markov decision processes with neural ODEs, 2020.

[22] Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural ODEs, 2019.

[23] A. Einstein. Über die von der molekularkinetischen theorie der wärme geforderte bewegung von in ruhenden flüssigkeiten suspendierten teilchen. *Annalen der Physik*, 1905.

[24] Chris Finlay, Jörn-Henrik Jacobsen, Levon Nurbekyan, and Adam M Oberman. How to train your neural ODE: the world of jacobian and kinetic regularization, 2020.

[25] Christopher Flynn. stochastic. https://github.com/crflynn/stochastic, 2020.

[26] Peter K Friz and Martin Hairer. *A Course on Rough Paths With an Introduction to Regularity Structures Second Edition*. Springer, 2014.

[27] Matthieu Garcin. Estimation of time-dependent hurst exponents with variational smoothing and application to forecasting foreign exchange rates. *Physica A: Statistical Mechanics and its Applications*, 2016.

[28] Jim Gatheral, Thibault Jaisson, and Mathieu Rosenbaum. Volatility is rough. *Quantitative Finance*, 2018.

[29] Ido Golding and Edward C. Cox. Physical nature of bacterial cytoplasm. *Phys. Rev. Lett.*, 2006.

[30] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014.

[31] Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. FFJORD: Free-form continuous dynamics for scalable reversible generative models. *ICLR*, 2019.

[32] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces, 2021.

[33] M Hairer. Introduction to malliavin calculus, 2021.

[34] Fabian Harang, Torstein Nilssen, and Frank Proske. Girsanov theorem for multifractional Brownian processes, 2018.

[35] Y. He, S. Burov, R. Metzler, and E. Barkai. Random time-scale invariant diffusion and transport coefficients. *Phys. Rev. Lett.*, 2008.

[36] Markus Heinonen, Cagatay Yildiz, Henrik Mannerström, Jukka Intosalmi, and Harri Lähdesmäki. Learning unknown ODE models with Gaussian processes. In *ICML*, 2018.

[37] James Hensman, Nicolò Fusi, and Neil D. Lawrence. Gaussian processes for big data. In *UAI*, 2013.

[38] T Hida and Si Si. *Lectures on White Noise Functionals*. World Scientific, 2008.

[39] Liam Hodgkinson, Chris van der Heide, Fred Roosta, and Michael W. Mahoney. Stochastic continuous normalizing flows: training SDEs as ODEs. In *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, 2021.

[40] John C. Hull. *Options, futures, and other derivatives*. Pearson Prentice Hall, 2006.

[41] Felix Höfling and Thomas Franosch. Anomalous transport in the crowded world of biological cells. *Reports on Progress in Physics*, 2013.

[42] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *CoRR*, abs/1806.07572, 2018.

[43] Jacob Kelly, Jesse Bettencourt, Matthew James Johnson, and David Duvenaud. Learning differential equations that are easy to solve. In *NeurIPS*, 2020.

[44] Patrick Kidger. On neural differential equations. *CoRR*, 2022.

[45] Patrick Kidger, Ricky T. Q. Chen, and Terry Lyons. "Hey, that's not an ODE": Faster ODE Adjoints via Seminorms. *ICML*, 2021.

[46] Patrick Kidger, James Foster, Xuechen Li, and Terry Lyons. Efficient and accurate gradients for neural SDEs. In *NeurIPS*, 2021.

[47] Patrick Kidger, James Foster, Xuechen Li, Harald Oberhauser, and Terry Lyons. Neural SDEs as Infinite-Dimensional GANs. *ICML*, 2021.

[48] Patrick Kidger and Terry Lyons. Signatory: differentiable computations of the signature and logsignature transforms, on both CPU and GPU. In *ICLR*, 2021.

[49] Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural Controlled Differential Equations for Irregular Time Series. *NeurIPS*, 2020.

[50] Diederik P Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. On density estimation with diffusion models. In *NeurIPS*, 2021.

[51] Peter E. Kloeden and Eckhard Platen. *Numerical Solution of Stochastic Differential Equations*. Springer, 1992.

[52] Joachim Lebovits and Jacques Lévy Véhel. White noise-based stochastic calculus with respect to multifractional Brownian motion. *Stochastics: An International Journal of Probability and Stochastic Processes*, 2014.

[53] Yann LeCun, Corinna Cortes, and CJ Burges. MNIST handwritten digit database. *ATT Labs [Online]. Available:http://yann.lecun.com/exdb/mnist*, 2010.

[54] Xuechen Li, Ting-Kam Leonard Wong, Ricky T. Q. Chen, and David Duvenaud. Scalable gradients for stochastic differential equations. In *AISTATS*, 2020.

[55] Andreas Look, Chen Qiu, Maja Rudolph, Jan Peters, and Melih Kandemir. Deterministic inference of neural stochastic differential equations. *CoRR*, abs/2006.08973, 2020.

[56] Terry J. Lyons. Differential equations driven by rough signals. *Revista Matemática Iberoamericana*, 1998.

[57] Miguel Lázaro-Gredilla and Michalis K. Titsias. Variational heteroscedastic Gaussian process regression. In *ICML*, 2011.

[58] Benoit B. Mandelbrot and John W. Van Ness. Fractional Brownian motions, fractional noises and applications. *SIAM Review*, 1968.

[59] Stefano Massaroli, Michael Poli, Federico Califano, Jinkyoo Park, Atsushi Yamashita, and Hajime Asama. Optimal energy shaping via neural approximators, 2021.

[60] Stefano Massaroli, Michael Poli, Jinkyoo Park, Atsushi Yamashita, and Hajime Asama. Dissecting neural ODEs, 2021.

[61] Stefano Massaroli, Michael Poli, Sho Sonoda, Taji Suzuki, Jinkyoo Park, Atsushi Yamashita, and Hajime Asama. Differentiable multiple shooting layers, 2021.

[62] James Mercer. Functions of positive and negative type, and their connection the theory of integral equations. *Phil. Trans. R. Soc. Lond. A*, 1909.

[63] James Morrill, Adeline Fermanian, Patrick Kidger, and Terry Lyons. A generalised signature method for multivariate time series feature extraction, 2021.

[64] James Morrill, Cristopher Salvi, Patrick Kidger, James Foster, and Terry Lyons. Neural rough differential equations for long time series. *ICML*, 2021.

[65] S. V. Muniandy and S. C. Lim. Modeling of locally self-similar processes using multifractional Brownian motion of riemann-liouville type. *Phys. Rev. E*, 2001.

[66] Avik Pal, Yingbo Ma, Viral Shah, and Christopher Rackauckas. Opening the blackbox: Accelerating neural differential equations by regularizing internal solver heuristics, 2021.

[67] Giorgio Parisi. Brownian motion. *Nature*, 2005.

[68] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *NeurIPS*, 2019.

[69] Romain-François Peltier and Jacques Lévy Véhel. Multifractional Brownian Motion : Definition and Preliminary Results. Research report, INRIA, 1995.

[70] Zhongmin Qian and Xingcheng Xu. Itô integrals for fractional Brownian motion and applications to option pricing, 2018.

[71] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[72] D. Revuz and M. Yor. *Continuous Martingales and Brownian Motion*. Springer Berlin Heidelberg, 2004.

[73] Simo Sarkka, Arno Solin, and Jouni Hartikainen. Spatiotemporal learning via infinite-dimensional bayesian filtering and smoothing: A look at Gaussian process regression through Kalman filtering. *IEEE Signal Processing Magazine*, 2013.

[74] Edward Snelson and Zoubin Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *NeurIPS*, 2006.

[75] Arno Solin, Ella Tamir, and Prakhar Verma. Scalable inference in sdes by direct matching of the Fokker–Planck–Kolmogorov equation. In *NeurIPS*, 2021.

[76] Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. In *NeurIPS*, 2021.

[77] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021.

[78] Fallaw Sowell. Modeling long-run behavior with the fractional arima model. *Journal of Monetary Economics*, 1992.

[79] Simo Särkkä. *Bayesian Filtering and Smoothing*. Institute of Mathematical Statistics Textbooks. Cambridge University Press, 2013.

[80] Simo Särkkä and Arno Solin. *Applied Stochastic Differential Equations*. Institute of Mathematical Statistics Textbooks. Cambridge University Press, 2019.

[81] M. Talagrand. Transportation cost for Gaussian and other product measures. *Geometric and functional analysis*, 1996.

[82] Ruey S. Tsay. Conditional heteroscedastic time series models. *Journal of the American Statistical Association*, 1987.

[83] Belinda Tzen and Maxim Raginsky. Neural stochastic differential equations: Deep latent Gaussian models in the diffusion limit. *CoRR*, 2019.

[84] Belinda Tzen and Maxim Raginsky. Theoretical guarantees for sampling and inference in generative models with latent diffusions. In *COLT*, 2019.

[85] Alpha Vantage. Alpha vantage, 2022.

[86] Christopher K. I. Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In *NeurIPS*, 2001.

[87] W. Willinger, M.S. Taqqu, R. Sherman, and D.V. Wilson. Self-similarity through high-variability: statistical analysis of ethernet lan traffic at the source level. *IEEE/ACM Transactions on Networking*, 1997.

[88] Cagatay Yildiz, Markus Heinonen, Jukka Intosalmi, Henrik Mannerström, and Harri Lähdesmäki. Learning stochastic differential equations with Gaussian processes without gradient matching, 2018.

[89] Linfeng Zhang, Weinan E, and Lei Wang. Monge-Ampère flow for generative modeling, 2018.

[90] Juntang Zhuang, Nicha C Dvornek, Sekhar Tatikonda, and James s Duncan. MALI: A memory efficient and reverse accurate integrator for neural ODEs. In *ICLR*, 2021.

[91] Çağatay Yıldız, Markus Heinonen, and Harri Lähdesmäki. ODE^{2}VAE: Deep generative second order odes with Bayesian neural networks. In *NeurIPS*, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See the last paragraph of Section 4.4
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] Mentioned in the main text and more detailed in Appendix B
   (b) Did you include complete proofs of all theoretical results? [Yes] Complete proof can be found in Appendix B
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] MNIST dataset for image generation tasks.
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]