Weierstraß-Institut
für Angewandte Analysis und Stochastik

Leibniz-Institut im Forschungsverbund Berlin e. V.

Preprint ISSN 2198-5855

Functional SDE approximation inspired by a deep operator network architecture

Martin Eigel, Charles Miranda

submitted: December 21, 2023

Weierstrass Institute
Mohrenstr. 39
10117 Berlin
Germany
E-Mail: martin.eigel@wias-berlin.de
charles.miranda@wias-berlin.de

No. 3079
Berlin 2023

2020 *Mathematics Subject Classification.* 65C30, 60H10, 91G60, 60H35, 68T07.

Key words and phrases. SDE, operator network, neural network approximation, Wiener chaos, polynomial approximation, functional representation.

ME acknowledge partial funding from the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) in the the priority programme SPP 2298 “Theoretical Foundations of Deep Learning”. ME & CM acknowledge funding by the ANR-DFG project “COFNET: Compositional functions networks - adaptive learning for high-dimensional approximation and uncertainty quantification”.

Edited by
Weierstraß-Institut für Angewandte Analysis und Stochastik (WIAS)
Leibniz-Institut im Forschungsverbund Berlin e. V.
Mohrenstraße 39
10117 Berlin
Germany

Fax: +49 30 20372-303
E-Mail: preprint@wias-berlin.de
World Wide Web: http://www.wias-berlin.de/

Functional SDE approximation inspired by a deep operator network architecture

Martin Eigel, Charles Miranda

Abstract

We present a novel approach to solve Stochastic Differential Equations (SDEs) with Deep Neural Networks by a Deep Operator Network (DeepONet) architecture. The notion of DeepONets relies on operator learning in terms of a reduced basis. We make use of a polynomial chaos expansion (PCE) of stochastic processes and call the corresponding architecture SDEONet. The PCE has been used extensively in the area of uncertainty quantification with parametric partial differential equations. This however is not the case with SDE, where classical sampling methods dominate and functional approaches are seen rarely. A main challenge with truncated PCEs occurs due to the drastic growth of the number of components with respect to the maximum polynomial degree and the number of basis elements. The proposed SDEONet architecture aims to alleviate the issue of exponential complexity by learning a sparse truncation of the Wiener chaos expansion. A complete convergence analysis is presented, making use of recent Neural Network approximation results. Numerical experiments illustrate the promising performance of the suggested approach in 1D and higher dimensions.

1 Introduction

Stochastic differential equations (SDE) can be seen as a form of generalisation of ordinary differential equations (ODE) by the introduction of or more stochastic processes into the formulation. The trajectories are hence also stochastic processes and the theoretical analysis requires in particular a notion of integrating with respect to white noise, which is given by the Ito integral. They are heavily used in physical models such as molecular dynamics, in financial mathematics and are also very prominent in recent developments in deep learning, in particular in the highly active field of generative models [41]. A standard way to obtain numerical approximations of the solution is by using the Euler-Maruyama scheme, an adaptation of the explicit Euler scheme for ODEs. The convergence rate depends directly on the regularity of the SDE, and hence is quite limited by the typical lack thereof.

In this work, we consider a functional representation of SDEs in terms of polynomial chaos expansions (PCE), which use tensorisations of polynomials that are orthogonal with respect to a certain probability measure as a basis. Any stochastic process with finite variance can be written in this way, as stated in the famous theorem of Cameron and Martin [5, 20]. The PCE has been very popular in the analysis and numerical treatment of parametric PDEs in the field of Uncertainty Quantification [36, 7, 12]. For SDEs, despite advantages of functional approximations such as direct access to statistical quantities of interest, these are an exception [17] and stochastic methods are clearly dominant. A central reason for this is that functional approximations of irregular processes usually require a fine discretisation and a large number of expansion terms, i.e. high polynomial degrees, to achieve an adequate accuracy. Due to the exponential growth of complexity ("curse of dimensionality"), this cannot be handled practically without modern compression tools. Moreover, the analysis of such elaborate schemes is rather difficult.

DOI 10.20347/WIAS.PREPRINT.3079 Berlin 2023

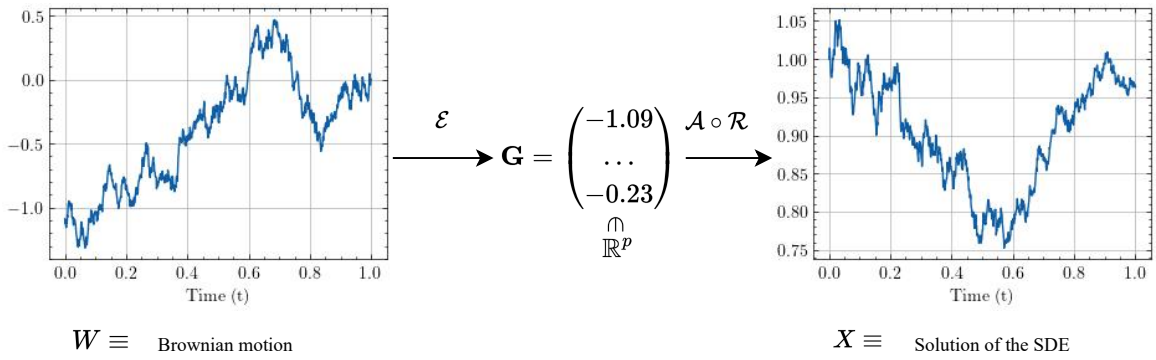


Figure 1: Sketch of the SDEONet action working on input G (processed Brownian motion, left), approximation $\mathcal{A}$ and reconstruction $\mathcal{R}$ of the SDE solution (right).

Our developed approach relies on a deep neural network (NN) architecture called Deep Operator Network (DeepONet) [26], which aims at learning (approximations of the actions of) operators in infinite dimensional function spaces. DeepONets feature a specific structure consisting of a trunk and a branch network that enable the complete theoretical analysis of these NN representations [24]. While DeepONets and related approaches such as Fourier Neural Operators have been used in the context of (partial) differential equations [15, 23], an application to SDEs has not been presented to our knowledge. We use the notion of reduced basis exploration provided by a DeepONet with the PCE of an SDE as analysed in [18], leading to a compressed functional SDE representation where the evolution of the coefficients follows the trajectories of an appropriate ODE. The principle of the devised architecture is sketched in Figure 1. There, a Brownian motion W is encoded by $\mathcal{E}$ with respect to an assumed basis in time, leading to input G of approximation $\mathcal{A}$ and reconstruction $\mathcal{R}$. The result of the network operation is the solution X of the SDE.

As our model problem, we consider the continuous stochastic process $(X_t)_{t\in[0,T]}$ that satisfies the SDE,

$$\mathrm{d}X_t = \mu(t, X_t)\,\mathrm{d}t + \sigma(t, X_t)\,\mathrm{d}W_t, \text{ with } X_0 = x_0, \tag{1.1}$$

where $(W_t)_{t\in[0,T]}$ is a Brownian motion defined on a filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\in[0,T]}, \mathbb{P})$.

Our main achievements are:

- Development of a DeepONet inspired architecture for the functional (Wiener chaos) representation of SDE solutions.
- Convergence and complexity analysis of this architecture in terms of the discretisation parameters. To achieve this, recent results on NN approximations of polynomials and Hölder continuous functions are used [37, 32, 34].

We present our main result Theorem 4.1 qualitatitely for the devised SDEONet architecture in the following.

Theorem 1.1 (Main result)**.** *Let $p, m = 2^k \in \mathbb{N}$, $\mathcal{G}$ be a SDE solution operator given by Definition 3.8*

and $\varepsilon \in \left(0, \left[\frac{4}{e^2 T}\right]^{1/3}\right)$. *Then, there exists a SDEONet* $\mathcal{N}^{p,m}$ *given by Definition 3.10 that satisfies*

$$\hat{E} \lesssim \left(\frac{1}{(p+1)!} + 2^{-k}\right)^{1/2} + \varepsilon \sqrt{\sum_{j=1}^{p} \mathbb{E}[\widetilde{\Psi_j}^2]} + \sqrt{\varepsilon}\left(1 - \frac{1}{(p+1)!}\frac{T\mathfrak{c}(K,T)^{2(p+1)}}{2(p+1)+1}(1+\mathfrak{c}(K,T))^{p+1}\right)^{1/2},$$

with $\mathfrak{c}(K,T)$ *a constant that depends only on* K*, linked to the regularity of the drift* μ *and the diffusion* σ*, and* T*. The SDEONet is composed of an* approximator $\mathcal{A}$ *that satisfies*

$$\begin{aligned}\operatorname{size}(\mathcal{A}) &\leq C_1 p \max_{j\in\{1,\dots,p\}} |k_j^*|^3 \log(1+|k_j^*|)|k_j^*|_0^2 \log(p\varepsilon^{-1}),\\ \operatorname{depth}(\mathcal{A}) &\leq C_1 \max_{j\in\{1,\dots,p\}} |k_j^*| \log(1+|k_j^*|)^2 |k_j^*|_0 \log(1+|k_j^*|_0)\log(p\varepsilon^{-1}),\end{aligned}$$

with a positive constant $C_1 > 0$ *independent of* p, m, ε *and of a* trunk net τ *that satisfies*

$$\operatorname{depth}(\tau) \leq (2 + \lceil \log_2(n+1)\rceil)(12+n) \qquad \textit{and} \qquad \operatorname{size}(\tau) \leq C_2 p \left(\frac{\varepsilon}{\sqrt{T}}\right)^{-\frac{1}{n+1}},$$

with a constant $C_2 > 0$ *that depends only on the regularity of the* x_j*.*

With this work, we aim to provide a new perspective on SDE (functional) solution representations with NNs that can be fully analysed mathematically. It should be noted that there have been different attempts to represent and analyse SDE solutions with NNs. For instance, in [2] the Kolmogorov backward SDE was learned with an SDE to show complexity bounds of an equivalent PDEs. In particular, an Euler discretization scheme (where layers correspond to the steps of the time discretisation) and an architecture based on the Picard iteration were presented. Inspired by this work, an ResNet-based architecture for the Langevin SDE was presented in [11], the results of which can be used for interacting particle transport in the context of Bayesian inverse problems [14, 10].

An alternative compression technique can be identified in low-rank tensor formats [16, 29]. In particular, the hierarchical formats such as the well-known tensor trains (TT) [33] can reduce exponential complexity to a polynomial one if low-rank approximability is satisfied. In [1] it was demonstrated that a TT based Longstaff-Schwarz algorithm competes with (and even surpasses) equivalent NN constructions [3].

The structure of the paper is as follows: Preliminaries on the considered SDEs and their approximation in terms of Wiener chaos expansions are reviewed in Section 2. We recall a central convergence result and the correspondence of expansion coefficients with ODE trajectories. Section 3 is concerned with the introduction of NN and in particular DON. Moreover, our new SDEONet architecture is described. The convergence analysis of this architecture is carried out in Section 4, combining results from Wiener chaos and NN approximation of polynomials and functions with Hölder regularity. The practical performance of SDEONets is illustrated in Section 5 on the basis of benchmark problems in one and more dimensions. The paper ends with a summary of the achieved results and an outlook into future work.

2 SDEs and Wiener chaos

In this section, we review polynomial chaos representations of stochastic processes in terms of Hermite polynomials, as presented in [19]. We introduce the specific setting we use for the construction of the SDEONet architecture in Section 3. First, we recall the SDE setting and requirements for well-posedness and regularity of the solution.

2.1 Definitions and notation

Given a probability space $(\Omega, \mathcal{F}, \mathbb{P})$ and a $\mathbb{R}^d$-valued Brownian motion W, we consider

- $\{(\mathcal{F}_t); t \in [0,T]\}$, the filtration generated by the Brownian motion W,
- $L^p(\mathcal{F}) := L^p(\Omega, \mathcal{F}, \mathbb{P}), p \in \mathbb{N}^*$, the space of all $\mathcal{F}$-measurable random variables (r.v.) $X : \Omega \to \mathbb{R}^d$ satisfying $\|X\|_p^p := \mathbb{E}[\|X\|_{\ell_p}^p] < \infty$,
- $C^{p,q}(U \times V, W), p, q \in \mathbb{N} \cup \{\infty\}$, the space of functions $f : U \times V \to W$ that are p continuously differentiable in the first component and q continuously differentiable in the second component,
- $C^\beta(K) := \{f \in C^n(K) : \|f\|_{C^\beta} \leq \infty\}$, with $K \subset \mathbb{R}^d$ is compact and $\beta = (n, \xi) \in \mathbb{N} \times (0,1]$, the space of β-Hölder continuous functions, where

$$\|f\|_{C^\beta} := \max\left\{\max_{|\alpha|\leq n} \|\partial^\alpha f\|_\infty, \max_{|\alpha|=n} \mathrm{Lip}_\xi(\partial^\alpha f)\right\} \quad \text{and} \quad \mathrm{Lip}_\xi(f) := \sup_{x\neq y\in K} \frac{|f(x)-f(y)|}{|x-y|^\xi}.$$

To ensure the uniqueness of the solution of (1.1), we henceforth require the following assumptions to be satisfied.

Assumption 2.1. *Let $T > 0$ and let*

$$\mu : [0,T] \times \mathbb{R}^d \to \mathbb{R}^d, \qquad \sigma : [0,T] \times \mathbb{R}^d \to \mathbb{R}^{d\times m}$$

be $\mathcal{F}$-measurable functions satisfying

1 *linear growth*

$$\|\mu(t,x)\|_2 + \|\sigma(t,x)\|_{\mathsf{Fro}} \leq K(1+\|x\|_2), \quad \forall x \in \mathbb{R}^d, t \in [0,T] \tag{2.1}$$

for some constant $K > 0$,

2 *uniform Lipschitz continuity with a Lipschitz constant, $K \geq 0$*

$$\|\mu(t,x)-\mu(t,y)\|_2 + \|\sigma(t,x)-\sigma(t,y)\|_{\mathsf{Fro}} \leq K\|x-y\|_2, \quad \forall x,y \in \mathbb{R}^d, t \in [0,T]. \tag{2.2}$$

Let x_0 be a random variable that is independent of $\mathcal{F}$ with finite second-moment $\mathbb{E}[\|x_0\|_2^2] \leq \infty$.

Under these assumptions, [31, Theorem 5.2.1] asserts that the SDE (1.1) has a unique t-continuous solution X_t.

Definition 2.2 (Multiple stochastic integrals [28, 30]). *Let $f \in L^2([0,T]^n)$ be a symmetric function. The n-multiple stochastic integral $I_n(f)$ is defined as the stochastic integral over the cone $\mathcal{C} := \{0 < t_1 < t_2 < \cdots < t_n \leq T\}$, i.e.,*

$$I_n(f) := \int_0^T \int_0^{t_n} \cdots \int_0^{t_2} f(t_1, \ldots, t_n) \, \mathrm{d}W_{t_1} \ldots \mathrm{d}W_{t_n}$$

It satisfies the following properties

- I_n *is linear,*
- $I_n(f) = I_n(\tilde{f})$, *where* $\tilde{f}$ *is the symmetrization of* f, *which is*

$$\tilde{f}(t_1, \ldots, t_n) := \frac{1}{n!} \sum_{\sigma \in S_n} f(t_{\sigma(1)}, \ldots, t_{\sigma(n)}),$$

- $\mathbb{E}[I_n(f) I_m(g)] = \delta_n^m n! \langle \tilde{f}, \tilde{g} \rangle_{L^2([0,T]^n)}$.

2.2 Wiener chaos expansion of SDEs

In this section we recall the notion of the Wiener chaos expansion and convergence results required later. For more information, we refer to [30, 20].

Normalised Hermite polynomials are defined through the identities

$$H_0(x) := 1 \tag{2.3}$$

$$H_n(x) := \frac{(-1)^n}{\sqrt{n!}} \exp\left(\frac{x^2}{2}\right) \frac{\mathrm{d}^n}{\mathrm{d}x^n} \left(\exp\left(-\frac{x^2}{2}\right)\right), \quad n \geq 1. \tag{2.4}$$

The n-th Wiener chaos $\mathcal{H}_n$ is the closed linear subspace of $L^2(\Omega, \mathcal{F}, \mathbb{P})$ generated by the family of random variables $\left\{ H_n\left(\int_0^T h_s \, \mathrm{d}W_s\right) : \|h\|_{L^2([0,T])} = 1 \right\}$. The vector spaces $\mathcal{H}_n, n \geq 0$, are orthogonal, giving rise to the Wiener chaos expansion [30, Theorem 1.1.1]

$$L^2(\Omega, \mathcal{F}, \mathbb{P}) = \bigoplus_{n=0}^{\infty} \mathcal{H}_n. \tag{2.5}$$

Therefore, any random variable $Y \in L^2(\Omega, \mathcal{F}, \mathbb{P})$ admits an orthogonal decomposition

$$Y = y_0 + \sum_{k=1}^{\infty} \sum_{|n|=k} y_k^n \prod_{i=1}^{\infty} H_{n_i}\left(\int_0^T e_i(s) \, \mathrm{d}W_s\right), \tag{2.6}$$

where $n = (n_i)_{i \geq 1}$ is a sequence of positive integers determining the polynomial degree, $|n| = \sum_{i \geq 1} n_i$, and $(e_i)_{i \geq 1}$ is an orthonormal basis of $L^2([0,T])$. The coefficients are given by projection,

$$y_0 = \mathbb{E}[Y],$$
$$y_k^n = \mathbb{E}\left[Y \prod_{i=1}^{\infty} H_{n_i}\left(\int_0^T e_i(s) \, \mathrm{d}W_s\right) \right].$$

We now consider the one-dimensional continuous stochastic process, $(X_t)_{t \in [0,T]}$ satisfying (1.1). Using (2.6) and Malliavin calculus, [19] derived the following propagator system.

Theorem 2.3 (Propagator system [19, Theorem 2]). *Given Assumption 2.1, let $(X_t)_{t\in[0,T]}$ satisfy (1.1) and assume that $(X_t)_t \in L^2([0,T]\times\Omega)$. Then, X_t exhibits the chaos expansion*

$$X_t = \sum_{k\geq 0}\sum_{|\alpha|=k} x_\alpha(t) \underbrace{\prod_{i=1}^{\infty} H_{\alpha_i}\overbrace{\left(\int_0^T e_i(s)\,\mathrm{d}W_s\right)}^{=:G_i}}_{=:\Psi_\alpha} \tag{2.7}$$

and the coefficients $x_\alpha(t)$ satisfy the system of ordinary differential equations

$$\frac{\mathrm{d}x_\alpha}{\mathrm{d}t}(t) = \mu(t,X_t)_\alpha + \sum_{j=1}^{\infty}\sqrt{\alpha_j}e_j(t)\sigma(t,X_t)_{\alpha^-(j)}, \tag{2.8}$$

$$x_\alpha(0) = 1_{\alpha=0}x_0, \tag{2.9}$$

where $\mu(t,X_t)_\alpha$ (resp. $\sigma(t,X_t)_\alpha$) denotes the α-coefficients of the Wiener chaos expansion associated with the random variable $\mu(t,X_t)$ (resp. $\sigma(t,X_t)$), and $\alpha^-(j) = (\alpha_1,\dots,\alpha_{j-1},\alpha_j - 1,\alpha_{j+1},\dots)$.

In order to approximate the process $(X_t)_t$, one considers the truncation,

$$X_t^{p,k} := \sum_{j=0}^{p}\sum_{|\alpha|=j} x_\alpha(t)\prod_{i=1}^{k} H_{\alpha_i}\left(\int_0^T e_i(s)\,\mathrm{d}W_s\right), \tag{2.10}$$

which uses orthogonal projections Ψ_α only with respect to the first k basis elements $(e_i)_{1\leq i\leq k}$ and only up to the p-th order Wiener chaos. In [18], the authors derive an upper bound on the L^2-error $\mathbb{E}[(X_t^{p,k}-X_t)^2]$.

Theorem 2.4 (L^2-error of the Wiener chaos truncation). *Given Assumption 2.1 and let $(X_t)_{t\in[0,T]}$ satisfy (1.1). Moreover, assume that $\mu,\sigma\in C^{1,\infty}([0,T]\times\mathbb{R})$ such that*

$$\left|\frac{\partial^{\ell+m}\mu}{\partial t^\ell\partial x^m}(t,x) - \frac{\partial^{\ell+m}\mu}{\partial t^\ell\partial x^m}(t,y)\right| + \left|\frac{\partial^{\ell+m}\sigma}{\partial t^\ell\partial x^m}(t,x) - \frac{\partial^{\ell+m}\sigma}{\partial t^\ell\partial x^m}(t,y)\right| \leq K|x-y|, \quad K>0,$$

for $t\in[0,T]$, $x,y\in\mathbb{R}$, any $\ell\in\{0,1\}$, and $m\geq 0$. Then it holds that

$$\mathbb{E}[(X_t^{p,k}-X_t)^2] \leq C(1+x_0^2)\left(\frac{1}{(p+1)!} + \sum_{\ell=k+1}^{\infty}\left(E_\ell(t)^2 + \int_0^t E_\ell^2(\tau)\,\mathrm{d}\tau\right)\right), \tag{2.11}$$

where $C=C(t,K)$ is a positive constant and the function $E_\ell(t)$ is defined by

$$E_\ell(t) := \int_0^t e_\ell(s)\,\mathrm{d}s. \tag{2.12}$$

In the following, we use the Haar basis $\{e_0, e_{2^{n-1}+j} : 1\leq j\leq 2^{n-1}, n\geq 1\}$ defined by

$$e_0(t) := \frac{\mathbf{1}_{[0,T]}(t)}{\sqrt{T}},$$

$$e_{2^{n-1}+j}(t) := \sqrt{\frac{2^{n-1}}{T}}\left(\mathbf{1}_{[T\frac{2j-2}{2^n},T\frac{2j-1}{2^n}]} - \mathbf{1}_{[T\frac{2j-1}{2^n},T\frac{2j}{2^n}]}\right).$$

For this basis, the integrals G_i in (2.7) can be computed explicitly, which is used in our analysis. In fact,

$$\int_0^T e_0(t)\,\mathrm{d}W_t = \frac{1}{\sqrt{T}} W_T,$$
$$\int_0^T e_{2^{n-1}+j}(t)\,\mathrm{d}W_t = \frac{2^{\frac{n-1}{2}}}{\sqrt{T}} \left(\left(W_{T\frac{2j-1}{2^n}} - W_{T\frac{2j-2}{2^n}} \right) - \left(W_{T\frac{2j}{2^n}} - W_{T\frac{2j-1}{2^n}} \right) \right).$$

The first basis functions are plotted in Figure 2.

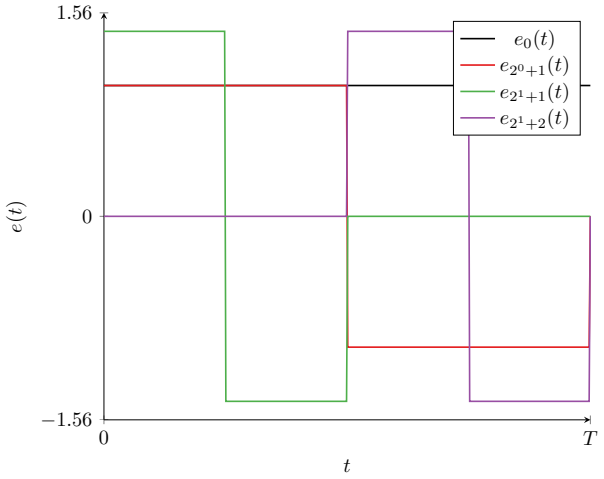


Figure 2: First Haar basis functions for $n \leq 2$.

3 Deep operator networks

In this section, we formally introduce neural networks and review results on deep operator networks [24]. These form the foundation of our architecture for SDEs that is presented in Section 3.3.

3.1 Neural Networks

A general neural network definition is as follows.

Definition 3.1 (Neural network and σ-realisation)**.** *Let $d, s, L \in \mathbb{N} \setminus \{0\}$. A* neural network Φ *with input dimension d, output dimension s and L layers is a sequence of matrix-vector tuples,*

$$\Phi = ((W^1, b^1), \ldots, (W^L, b^L)),$$

where $W^\ell \in \mathbb{R}^{n_\ell \times n_{\ell-1}}$ and $b^\ell \in \mathbb{R}^{n_\ell}$, with $n_0 = d$, $n_L = s$ and $n_1, \ldots, n_{L-1} \in \mathbb{N}$.

We denote by $\mathcal{N}_{d,L,s}$ the space of neural network Φ with input dimension d, output dimension s and L layers, and also $\mathcal{N}_{d,s} := \bigcup_{L\geq 1} \mathcal{N}_{d,L,s}$. If $K \subset \mathbb{R}^d$ and $\sigma : \mathbb{R} \to \mathbb{R}$ is an arbitrary activation function, the associated realisation of Φ with activation function σ over K is defined as the map $R_\sigma\Phi : K \to \mathbb{R}^s$ such that

$$R_\sigma\Phi(x) = A_L \circ \sigma \circ A_{L-1} \circ \cdots \circ \sigma \circ A_1(x),$$

where $A_\ell(x) = W^\ell \cdot x + b^\ell$ *is an affine transformation.*

We also introduce the following nomenclature for a neural network $\Phi \in \mathcal{N}_{d,s}$*,*

$$\operatorname{depth}(\Phi) := L, \qquad \operatorname{size}(\Phi) := \|\Phi\|_0 := \sum_{\ell=1}^{L} (n_{\ell-1}+1) n_\ell.$$

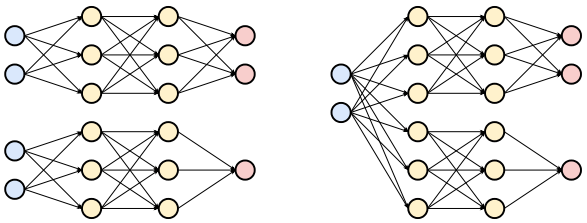


Figure 3: **Left:** Two neural networks. **Right:** Parallelisation with shared inputs of both networks according to Proposition 3.2.

The parallelisation of neural networks is a common operation that we use in the subsequent analysis.

Proposition 3.2 (Parallelisation [34, Definition 2.7])**.** *Let* $\Phi^1 = ((W_1^1, b_1^1), \dots, (W_L^1, b_L^1)) \in \mathcal{N}_{d,L,s_1}$ *and* $\Phi^2 = ((W_1^2, b_1^2), \dots, (W_L^2, b_L^2)) \in \mathcal{N}_{d,L,s_2}$ *be two neural networks with* d*-dimensional input. Then* $P(\Phi^1, \Phi^2) = ((\tilde{A}_1, \tilde{b}_1), \dots, (\tilde{A}_L, \tilde{b}_L))$*, where*

$$\tilde{A}_1 := \begin{pmatrix} A_1^1 \\ A_1^2 \end{pmatrix}, \quad \tilde{b}_1 := \begin{pmatrix} b_1^1 \\ b_1^2 \end{pmatrix}, \quad \tilde{A}_\ell := \begin{pmatrix} A_\ell^1 & 0 \\ 0 & A_\ell^2 \end{pmatrix}, \quad \tilde{b}_\ell := \begin{pmatrix} b_\ell^1 \\ b_\ell^2 \end{pmatrix} \quad \text{for } 1 < \ell \leq L$$

is a neural network with d*-dimensional input and* L *layers, called the* parallelisation *of* Φ^1 *and* Φ^2*. Moreover,* $P(\Phi^1, \Phi^2)$ *satisfies*

$$\operatorname{size}(P(\Phi^1, \Phi^2)) = \operatorname{size}(\Phi^1) + \operatorname{size}(\Phi^2) \quad \text{and} \quad R_\sigma P(\Phi^1, \Phi^2) = (R_\sigma \Phi^1, R_\sigma \Phi^2).$$

We recall approximation results for continuously differentiable and Hölder continuous functions that are used in the later analysis. The neural network complexity required for approximating multivariate Hermite polynomials up to a certain accuracy has been examined in [37].

Theorem 3.3 (Deep ReLU neural networks approximation of multivariate Hermite polynomials [37, Theorem 3.7])**.** *Let* $\Lambda \subset \{\alpha \in \mathbb{N}^\infty : |\alpha| < \infty\}$ *be* finite*. For every* $\varepsilon \in (0, e^{-1})$ *there exists a neural network* Φ_ε *such that*

$$\max_{\alpha \in \Lambda} \|H_\alpha - \tilde{H}_{\varepsilon,\alpha}\|_{L^2_\mu(\mathbb{R}^{|\operatorname{supp}(\Lambda)|})} \leq \varepsilon,$$

where $H_{\varepsilon,\alpha} = R_\sigma \Phi_\varepsilon : \mathbb{R}^{|\operatorname{supp}(\Lambda)|} \to \mathbb{R}^{|\Lambda|}$*,* $\operatorname{supp}(\Lambda) := \{j \in \operatorname{supp}(\alpha) : \alpha \in \Lambda\}$ *and* μ *is the multivariate Gaussian measure. Moreover, there exists a positive constant* C *(independent of* $m(\Lambda) := \max_{\alpha\in\Lambda} |\alpha|$*,* $d(\Lambda) := \max_{\alpha\in\Lambda} |\alpha|_0$ *and of* ε*) such that*

$$\begin{aligned} \operatorname{size}(\Phi_\varepsilon) &\leq C|\Lambda| m(\Lambda)^3 \log(1 + m(\Lambda)) d(\Lambda)^2 \log(\varepsilon^{-1}), \\ \operatorname{depth}(\Phi_\varepsilon) &\leq C m(\Lambda) \log(1 + m(\Lambda))^2 d(\Lambda) \log(1 + d(\Lambda)) \log(\varepsilon^{-1}). \end{aligned}$$

Remark 3.4. The result above gives complexity results for approximating multivariate Hermite polynomials using Deep ReLU neural networks. Instead, if we use the Rectified Power Unit (RePU) activation function defined by $\operatorname{RePU}^p : x \mapsto \max(x, 0)^p$, with $p \geq 2$, then there exists a Deep Neural network

with this activation function which represents *exactly* a multivariate polynomial. From [32, Proposition 2.14] we have that there exists a Deep RePU Neural network ϕ which represents exactly any polynomial $p \in \mathbb{P}_\Lambda$, and such that

$$\begin{aligned} \operatorname{size}(\phi) &\leq C|\Lambda|, \\ \operatorname{depth}(\phi) &\leq C \log(|\Lambda|), \end{aligned}$$

with a constant $C > 0$ depending only on the power p.

Theorem 3.5 (Approximation of β-Hölder continuous function [34, Theorem 3.1])**.** *Let $d \in \mathbb{N}$, $B, p > 0$ and $\beta = (n, \xi) \in \mathbb{N} \times (0, 1]$. Then, there exists a constant $c = c(d, n, \xi, B) > 0$ such that for any function $f \in C^\beta([-1/2, 1/2]^d)$ with $\|f\|_{C^\beta} \leq B$ and any $\varepsilon \in (0, 1/2)$ there is a neural network Φ_ε^f such that*

$$\begin{aligned} \|R_\sigma(\Phi_\varepsilon^f) - f\|_{L^p([-1/2,1/2]^d)} &< \varepsilon, \\ \|R_\sigma(\Phi_\varepsilon^f)\|_\infty &\leq \lceil B \rceil, \end{aligned}$$

and

$$\begin{aligned} \operatorname{depth}(\Phi_\varepsilon^f) &\leq (2 + \lceil \log_2(n + \xi) \rceil) \left(11 + \frac{n+\xi}{d}\right), \\ \operatorname{size}(\Phi_\varepsilon^f) &\leq c\varepsilon^{-\frac{d}{n+\xi}}. \end{aligned}$$

3.2 Operator Neural Networks

The basis for our NN construction are recent results on operator networks. Operators are mappings between infinite dimensional function spaces. Prominent examples are the solution operators for ODEs and PDEs, which map function space inputs to the solution of the differential equation in another function space [40, 26, 25, 23]. Typical inputs are parameters describing coefficients or initial and boundary data, as in particular is common in Uncertainty Quantification. The differential equation setting also provides a mathematical framework for statistical inverse problems, where the object of interest is the inverse operator that maps some observables to the underlying model data that is to be inferred [39, 13]. In recent years, machine learning-based operator approximation has attracted growing interest due to the possibly high cost of classical operator approximation techniques, particularly those related to high-dimensional parametric and nonlinear PDEs [27, 9]. Opposite to simulation methods, operator learning infers operators from solution data and a well-known approach is given by the DeepONet architecture [26]. DeepONet can query any coordinate in the (parameter) domain to obtain the value of the output function. However, for training and testing, the input function must be evaluated at a set of predetermined locations (often called “snapshots” in reduced basis methods), which requires a fixed observation grid for all observations.

Remark 3.6. For illustration, consider the 1D dynamical system defined on domain $[0, 1]$ by

$$\frac{\mathrm{d}u}{\mathrm{d}x}(x) = f(u(x), \phi(x), x), \qquad u(0) = 0.$$

The *operator* $\mathcal{G}$ that maps the perturbation ϕ to the solution u satisfies

$$(\mathcal{G}\phi)(x) = \int_0^x f((\mathcal{G}\phi)(y), \phi(y), y)\,\mathrm{d}y.$$

In the linear case $f(u(x), \phi(x), x) = \phi(x)$, the considered operator to be learned is the *antiderivative operator*

$$\mathcal{G} : \phi \mapsto \left(u : x \mapsto \int_0^x \phi(y)\,\mathrm{d}y \right).$$

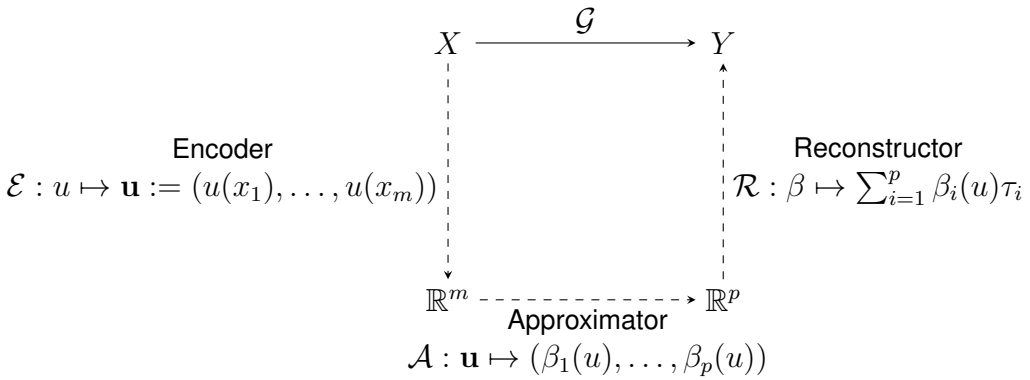


Figure 4: Structure of the deep operator network.

We henceforth assume that $D \subset \mathbb{R}^d$ and $U \subset \mathbb{R}^n$ are compact domains (e.g. with Lipschitz boundary). The architecture of a deep operator network is depicted in Figure 4, where the operator $\mathcal{G}$ is to be represented approximately. For this, the encoder $\mathcal{E}$ results in a finite dimensional input representation that is mapped via the approximator $\mathcal{A}$ to the finite dimensional out. Eventually, this is transferred to the image of $\mathcal{G}$ by the reconstruction $\mathcal{R}$. The following definition makes this rigorous.

Definition 3.7 (Deep operator network (DeepONet) [24, Definitions 2.1 & 2.4])**.** *Assume separable Banach spaces X, Y with continuous embeddings $\iota : X \hookrightarrow L^2(D)$ and $\bar{\iota} : Y \hookrightarrow L^2(U)$. Let $\mu \in \mathcal{P}_2(X)$ be a Borel probability measure on X with finite second moments such that there exists $A \subset X$ with $\mu(A) = 1$ and A consists of continuous functions. Moreover, let $\mathcal{G} : X \to Y$ be a Borel measurable mapping such that $\mathcal{G} \in L^2_\mu$. For the construction of the DeepONet architecture (see Figure 4), three* operators *are used:*

- ***Encoder**: Given a set of* sensor *points $\{x_j\}_{j=1}^m \subset X$, define the linear mapping*

$$\mathcal{E} : \begin{cases} C(D) & \longrightarrow \mathbb{R}^m \\ u & \longmapsto (u(x_1), \dots, u(x_m)) \end{cases} \tag{3.1}$$

  as the encoder *mapping.*

- ***Approximator**: Given sensor points $\{x_j\}_{j=1}^m$, the* approximator *is a deep neural network $\mathcal{A} \in \mathcal{N}_{m,p}$.*

- *Given the encoder and approximator, we define the **branch net***

$$\beta : \begin{cases} C(D) & \longrightarrow \mathbb{R}^p \\ u & \longmapsto R_\sigma \mathcal{A} \circ \mathcal{E}(u) \end{cases}. \tag{3.2}$$

  It represents the coefficients in the basis expansion.

- *Denote a **trunk net** by $\tau \in \mathcal{N}_{n,p}$ as a deep neural network representation of basis functions based on the encoder data.*

- ***Reconstructor***. *The τ-induced* reconstructor *is given by*

$$\mathcal{R} : \begin{cases} \mathbb{R}^p & \longrightarrow C(U) \\ \{\alpha_k\}_{k=1}^p & \longmapsto \sum_{k=1}^p \alpha_k (R_\sigma \tau)_k \end{cases} . \tag{3.3}$$

*A **DeepONet** $\mathcal{N}$ approximates the nonlinear operator $\mathcal{G}$. It is a mapping $\mathcal{N} : C(D) \to L^2(U)$ of the form $\mathcal{N} = \mathcal{R} \circ R_\sigma \mathcal{A} \circ \mathcal{E}$, where $\mathcal{E} : (X, \|\cdot\|_X) \to (\mathbb{R}^m, \|\cdot\|_{\ell^2})$ denotes the encoder given by (3.1), $R_\sigma \mathcal{A} : (\mathbb{R}^m, \|\cdot\|_{\ell^2}) \to (\mathbb{R}^p, \|\cdot\|_{\ell^2})$ denotes the approximation network, and $\mathcal{R} : (\mathbb{R}^p, \|\cdot\|_{\ell^2}) \to (L^2(U), \|\cdot\|_{L^2(U)})$ denotes the reconstruction of the form (3.3), induced by the trunk net τ.*

In [24], the authors study the approximation of $\mathcal{G}$ by $\mathcal{N}$. For the analysis, they consider the following error, measured in L^2_μ, where μ is associated with X,

$$\begin{aligned} \hat{E}^2 &:= \|\mathcal{G} - \mathcal{N}\|_{L^2_\mu} \\ &:= \int_X \int_U |\mathcal{G}(u)(y) - \mathcal{N}(u)(y)|^2 \, \mathrm{d}y \, \mathrm{d}\mu(u). \end{aligned} \tag{3.4}$$

3.3 SDEONet architecture

We now describe the construction of the SDEONet architecture, which is inspired by the DeepONet presented above and combined with the chaos representation of Section 2.2. For this, recall the polynomial chaos expansion (2.7) for a stochastic process written as

$$X_t(\omega) = \sum_{k \geq 0} \sum_{|\alpha|=k} x_\alpha(t) \Psi_\alpha(\omega), \tag{3.5}$$

where $\omega \in \Omega$. One can hence define a nonlinear operator $\mathcal{G}$ such that $X_t(\omega) = \mathcal{G}(\{W_s(\omega)\}_{s\in[0,T]})(t)$, i.e., it maps the Brownian motion W to the continuous stochastic process $(X_t)_{t\in[0,T]}$ satisfying (1.1). One can intuitively approximate such an operator $\mathcal{G}$ with a DeepONet $\mathcal{N}$.

Definition 3.8 (SDE solution operator)**.** *A (nonlinear) operator $\mathcal{G} : L^2([0,T] \times \Omega) \to L^2([0,T] \times \Omega)$ is said to be a if $\mathcal{G}(W)$ is a continuous stochastic process satisfying (1.1), where $W = (W_t)_{t\in[0,T]}$ is a Brownian motion. Given Assumption 2.1, it is the strong solution of the SDE with respect to W.*

Example 3.9 (Operator $\mathcal{G}$ of linear SDE)**.** *Consider the following* linear SDE

$$\mathrm{d}X_t = (a(t)X_t + b(t)) \, \mathrm{d}t + h(t) \, \mathrm{d}W_t,$$

where a, b and h are bounded functions on $[0,T]$. Then, for $\omega \in \Omega$ and $t \in [0,T]$, we have

$$\begin{aligned} \mathcal{G}(\omega)(t) = \exp\left(\int_0^t a(s) \, \mathrm{d}s\right) \Bigg(X_0 + \int_0^t \exp\left(-\int_0^s a(r) \, \mathrm{d}r\right) b(s) \, \mathrm{d}s \\ + \int_0^t \exp\left(-\int_0^s a(r) \, \mathrm{d}r\right) h(s) \, \mathrm{d}W_s(\omega) \Bigg) . \end{aligned}$$

The ingredients of the SDEONet architecture are defined next.

Definition 3.10 (SDEONet)**.** *Let $W = (W_t)_{t\in[0,T]}$ be a Brownian motion and $p, m = 2^k \in \mathbb{N}$ polynomial chaos discretisation parameters. We construct the SDEONet (see Figure 5) as a composition of the following operators:*

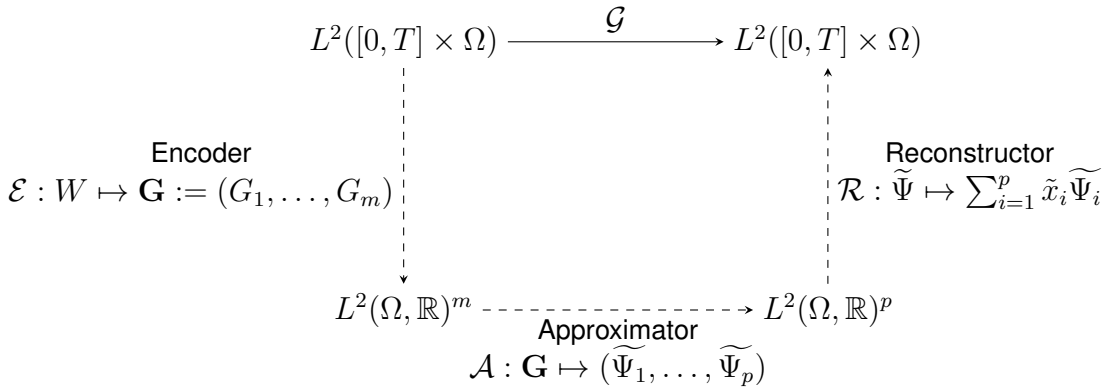


Figure 5: SDEONet components mapping a Brownian motion to the respective trajectory. Note that W (Brownian motion), G_i (integrals of $(e_i)_i$) and $\widetilde{\Psi}_i$ (basis approximation) are random variables defined on Ω. The coefficients $\tilde{x}_i$ are functions of time t.

- ***Encoder**: The mapping*

$$\mathcal{E}^{p,m} : L^2([0,T]\times\Omega) \to L^2(\Omega,\mathbb{R})^m \tag{3.6}$$

maps the Brownian motion W to $(G_i)_{i=0}^{m-1}$, with

$$G_i = \int_0^T e_i(t)\,\mathrm{d}W_t,$$

for $i = 2^{n-1}+j$, $1 \le j \le 2^{n-1}$ and $1 \le n \le k$.

- ***Approximator**: Given the values $\{G_i\}_{i=0}^{m-1}$, we denote an* approximator *$\mathcal{A} \in \mathcal{N}_{m,p}$ as a deep neural network such that its σ-realisation $R_\sigma \mathcal{A} = (\widetilde{\Psi}_j)_{j=1}^p : L^2(\Omega)^m \to L^2(\Omega,\mathbb{R})^p$ approximates the* chaos polynomials *$\Psi_{k_j^*}$.*

- ***Branch net**. Given the* encoder *and* approximator*, the* branch net *is defined as $\beta := \mathcal{A}\circ\mathcal{E}$.*

- ***Trunk net**: We denote a* trunk net *$\tau^p \in \mathcal{N}_{1,p}$ as a deep neural network such that its σ-realisation $R_\sigma \tau^p = (\widetilde{x}_j)_{j=1}^p : [0,T] \to \mathbb{R}^p$ approximates the coefficient functions $x_{k_j^*}$ in Theorem 2.3.*

- ***Reconstructor**. The τ-induced* reconstructor *is given by*

$$\mathcal{R}_\tau^p : \begin{cases} L^2(\Omega,\mathbb{R})^p & \longrightarrow L^2([0,T]\times\Omega) \\ (\widetilde{\Psi}_j)_{j=1}^p & \longmapsto \sum_{j=1}^p \widetilde{x}_j \widetilde{\Psi}_j \end{cases} \tag{3.7}$$

the mapping that approximates $(X_t^{m,p^*})_{t\in[0,T]}$.

*A **SDEONet** $\mathcal{N}^{m,p}$ approximates the nonlinear operator $\mathcal{G}$ in Definition 3.8. It is defined as mapping $\mathcal{N}^{m,p} : L^2([0,T]\times\Omega) \to L^2([0,T]\times\Omega)$ of the form $\mathcal{N}^{m,p} = \mathcal{R}_{\tau^p}\circ\mathcal{A}\circ\mathcal{E}$.*

4 Convergence analysis

In this section, a complete error analysis is carried out for the SDEONet architecture described in the last section. Similar to the analysis of DeepONets in [24], the overall error is split into several

components that are examined successively, namely truncation, approximation, and reconstruction errors. We state our main convergence result as a combination of the subsequent estimates in the following Theorem 4.1.

Let $\mathcal{G} : L^2([0,T]\times\Omega) \to L^2([0,T]\times\Omega)$ be a SDE solution operator according to Definition 3.8 and $\mathcal{N}^{m,p}$ be a SDEONet as of Definition 3.10 with $m, p \in \mathbb{N}$. We consider the error measured in the $L^2([0,T]\times\Omega)$-norm defined by

$$\begin{aligned}\hat{E} &:= \left(\int_0^T \mathbb{E}[|\mathcal{G}(W)(t) - \mathcal{N}(W)(t)|^2]\,\mathrm{d}t\right)^{1/2} \\ &= \left(\int_0^T \mathbb{E}[|X_t - \widetilde{X}_t^{m,p}|^2]\,\mathrm{d}t\right)^{1/2}.\end{aligned} \tag{4.1}$$

Theorem 4.1 (Neural network approximation of a strong solution of a SDE)**.** *Let $p, m = 2^k \in \mathbb{N}$, $\mathcal{G}$ be a SDE solution operator given by Definition 3.8 and $\varepsilon \in \left(0, \left[\frac{4}{e^2 T}\right]^{1/3}\right)$. Then, there exists a SDEONet $\mathcal{N}^{p,m}$ given by Definition 3.10 that satisfies*

$$\begin{aligned}\hat{E} \le &\min_{(q,\ell)\in J_p}\left((1+x_0^2)\left(\int_0^T C_3(t,K)\left(\frac{1}{(q+1)!} + \frac{2T(1+t)}{\ell}\right)\mathrm{d}t\right)\right)^{1/2} \\ &+ \sqrt{\varepsilon}\min_{(q,\ell)\in J_p}\left((1+x_0^2)\left(C_4(K,T) - \frac{1}{(q+1)!}\frac{T(C_5T)^{2(q+1)}}{2(q+1)+1}\left(1+\frac{1}{C_5T}\right)^{q+1}\right)\right)^{1/2} \\ &+ \varepsilon\sqrt{\sum_{j=1}^{p}\mathbb{E}[\widetilde{\Psi}_j^2]},\end{aligned}$$

with $C_3(t,K)$ defined in Theorem 2.4, $C_4(K,T) := \int_0^T e^{BteBt}\,\mathrm{d}t$ and $C_5 = C_5(K,T)$ are the constants in Lemma 4.9.

The SDEONet is composed of an approximator $\mathcal{A}$ *that satisfies*

$$\begin{aligned}\mathrm{size}(\mathcal{A}) &\le C_1 p \max_{j\in\{1,\dots,p\}} |k_j^*|^3 \log(1+|k_j^*|)|k_j^*|_0^2 \log(p\varepsilon^{-1}), \\ \mathrm{depth}(\mathcal{A}) &\le C_1 \max_{j\in\{1,\dots,p\}} |k_j^*| \log(1+|k_j^*|)^2 |k_j^*|_0 \log(1+|k_j^*|_0)\log(p\varepsilon^{-1}),\end{aligned}$$

with a positive constant $C_1 > 0$ independent of p, m, ε and of a trunk net τ *that satisfies*

$$\begin{aligned}\mathrm{depth}(\tau) &\le (2 + \lceil\log_2(n+1)\rceil)(12+n), \\ \mathrm{size}(\tau) &\le C_2 p\left(\frac{\varepsilon}{\sqrt{T}}\right)^{-\frac{1}{n+1}},\end{aligned}$$

with a constant $C_2 > 0$ that depends only on the regularity of the x_j.

Proof. By Lemma 4.3, the result comes from the combination of Lemma 4.4, Lemma 4.9 and Lemma 4.12. □

4.1 Auxiliary results

To prepare the convergence analysis in the following sections, we first introduce a decomposition of the error. This requires the finite multi-index set

$$\mathcal{I}_{p,k} := \left\{ \alpha \in \mathbb{N}^k : |\alpha| := \sum_{i=1}^{k} \alpha_i \leq p \right\} \tag{4.2}$$

and the *best p-terms* given by

$$k^* := \operatorname*{arg\,min}_{k=(k_1,\dots,k_p)\in\mathcal{I}^p} \int_0^T \mathbb{E}[|X_t - \sum_j x_{k_j}(t)\Psi_{k_j}(W)|^2]\,\mathrm{d}t. \tag{4.3}$$

Additionally, we define the following set of tuples

$$J_p := \left\{ (m,n) \in \mathbb{N}^2 : \binom{m+n}{n} \leq p \right\}.$$

The truncation error with respect to the basis $(e_i)_i$ in $L^2([0,T])$ can be bounded by the next result.

Proposition 4.2. *Let $n \in \mathbb{N}$. Then, for all $t \in [0,T]$,*

$$E_{2^{n-1}+j}(t) = \begin{cases} \frac{2^{\frac{n-1}{2}}}{\sqrt{T}} \left(t - T\frac{2j-2}{2^n}\right) & T\frac{2j-2}{2^n} \leq t \leq T\frac{2j-1}{2^n} \\ \frac{2^{\frac{n-1}{2}}}{\sqrt{T}} \left(T\frac{2j}{2^n} - t\right) & T\frac{2j-1}{2^n} \leq t \leq T\frac{2j}{2^n} \\ 0 & \text{else} \end{cases}, \tag{4.4}$$

and

$$\sum_{\ell=n+1}^{\infty} \sum_{j=1}^{2^n} \left(E^2_{2^{\ell-1}+j}(t) + \int_0^t E^2_{2^{\ell-1}+j}(\tau)\,\mathrm{d}\tau \right) \leq 2T(1+t)2^{-n}. \tag{4.5}$$

with E defined as in (2.12).

Proof. The first expression (4.4) follows from the definition of $e_{2^{n-1}+j}$. Concerning the second expression (4.5), first, note that $\max_{t\in[0,T]} E^2_{2^{n-1}+j}(t) = T2^{-(n+1)}$ and that $E_{2^{n-1}+j}(t) \neq 0$ only for $t \in \left[T\frac{2j-2}{2^n}, T\frac{2j}{2^n}\right]$. Then,

$$\begin{aligned} \sum_{\ell=n+1}^{\infty} \sum_{j=1}^{2^n} \left(E^2_{2^{\ell-1}+j}(t) + \int_0^t E^2_{2^{\ell-1}+j}(\tau)\,\mathrm{d}\tau \right) &\leq \sum_{\ell=n+1}^{\infty} \left(T2^{-(\ell+1)} + tT2^{-(\ell+1)} \right) \\ &= 2T(1+t)2^{-n}, \end{aligned}$$

since it is a geometric series. □

Lemma 4.3 (Decomposition of the error)**.** *Let $p, m \in \mathbb{N}$, $\mathcal{G}$ be a SDE solution operator according to Definition 3.8 and $\mathcal{N}^{p,m}$ a* SDEONet*. Then, the error $\hat{E}$ (4.1) can be decomposed as*

$$\hat{E} \leq \hat{E}_{\mathit{Trunc}} + \hat{E}_{\mathit{Approx}} + \hat{E}_{\mathit{Recon}}, \tag{4.6}$$

with

$$\hat{E}_{\text{Trunc}} := \left(\int_0^T \mathbb{E}[|X_t - \sum_{j=1}^{p} x_{k_j^*}(t)\Psi_{k_j^*}|^2] \, \mathrm{d}t \right)^{1/2}, \tag{4.7a}$$

$$\hat{E}_{\text{Approx}} := \left(\int_0^T \mathbb{E}[|\sum_{j=1}^{p} x_{k_j^*}(t)\Psi_{k_j^*} - \sum_{j=1}^{p} x_{k_j^*}(t)\widetilde{\Psi}_j|^2] \, \mathrm{d}t \right)^{1/2}, \tag{4.7b}$$

$$\hat{E}_{\text{Recon}} := \left(\int_0^T \mathbb{E}[|\sum_{j=1}^{p} x_{k_j^*}(t)\widetilde{\Psi}_j - \sum_{j=1}^{p} \widetilde{x}_j(t)\widetilde{\Psi}_j|^2] \, \mathrm{d}t \right)^{1/2}. \tag{4.7c}$$

Proof. Note that for any continuous process $(Y_t)_{t\in[0,T]}$ one has

$$\|Y\|^2_{L^2([0,T]\times\Omega)} := \frac{1}{T}\int_0^T \int_\Omega Y_t(\omega)^2 \, \mathrm{d}\mathbb{P}_{Y_t}(\omega) \, \mathrm{d}t,$$

which is a norm e.g. by using Minkowski's inequality. Applying the triangle inequality then gives the result. □

4.2 Truncation error

We first estimate the truncation error, which results from only using a finite number of basis elements and terms in the Wiener chaos expansion.

Lemma 4.4 (Upper bound of truncation error)**.** *Given Assumption 2.1 and let μ, σ satisfy the assumptions of Theorem 2.4. Then, the* truncation error (4.7a) *satisfies*

$$\hat{E}_{\text{Trunc}} \leq \min_{m,n\in J_p} \left((1+x_0^2) \left(\int_0^T C(t,K) \left(\frac{1}{(m+1)!} + \frac{2T(1+t)}{n} \right) \mathrm{d}t \right) \right)^{1/2}, \tag{4.8}$$

with $C(t,K)$ defined in Theorem 2.4.

Proof. Let $m, n \in \mathbb{N}$. Let $k = \lfloor \log_2(n) \rfloor$. Applying Theorem 2.4 leads to

$$\mathbb{E}[(X_t - X_t^{m,n})^2] \leq C(t,K)(1+x_0^2) \left(\frac{1}{(m+1)!} + \sum_{\ell=k+1}^{\infty} \sum_{j=1}^{2^k} \left(E_{j,\ell}(t)^2 + \int_0^t E_{j,\ell}^2(\tau) \, \mathrm{d}\tau \right) \right).$$

Integrating from $t = 0$ to T yields

$$\begin{aligned} \int_0^T \mathbb{E}[|X_t - X_t^{m,n}|^2] \, \mathrm{d}t \leq \int_0^T & C(t,K)(1+x_0^2) \\ & \times \left(\frac{1}{(m+1)!} + \sum_{\ell=k+1}^{\infty} \sum_{j=1}^{2^k} \left(E_{j,\ell}(t)^2 + \int_0^t E_{j,\ell}^2(\tau) \, \mathrm{d}\tau \right) \right) \mathrm{d}t. \end{aligned}$$

Then, using Proposition 4.2, we have

$$\begin{aligned} & \int_0^T C(t,K)(1+x_0^2) \left(\frac{1}{(m+1)!} + \sum_{\ell=k+1}^{\infty} \sum_{j=1}^{2^k} \left(E_{j,\ell}(t)^2 + \int_0^t E_{j,\ell}^2(\tau) \, \mathrm{d}\tau \right) \right) \mathrm{d}t \\ \leq & (1+x_0^2) \int_0^T C(t,K) \left(\frac{1}{(m+1)!} + \frac{2T(1+t)}{n} \right) \mathrm{d}t. \end{aligned}$$

The results follows by definition of k^*. □

The above lemma shows that the truncation error decays factorially fast in the number of polynomial chaos terms and linearly in the number of basis elements.

4.3 Approximation error

The second term in (4.6) is the approximation error that comes from the approximation of the polynomial chaos. This term is more involved and requires the use of Malliavin calculus, as introduced above. The approach is to explicitly introduce the L^2 error of the polynomial chaos and show that a neural network can indeed approximate the Hermite polynomials.

For the purposes of upper bounding the approximation error, we recall the following results.

Theorem 4.5 ([21, Theorem 2.9, page 289])**.** *Suppose that Assumption 2.1 is fulfilled. Then, there exists a continuous, adapted process X which is a strong solution of (1.1) relative to W, with initial condition x_0. Moreover, this process is square-integrable: for every $T > 0$, there exists a constant $C := C(K, T)$ such that*

$$\mathbb{E}[\|X_t\|_2^2] \leq C(1 + \mathbb{E}[\|x_0\|_2^2]) \exp(Ct)$$

for $0 \leq t \leq T$.

Proposition 4.6 ([18, Proposition 4.1])**.** *Under conditions of Theorem 2.4, we obtain the estimate*

$$\mathbb{E}[(D^n_{s_1,\dots,s_n} X_t)^2] \leq C^n (1 + x_0^2) \exp(Cnt),$$

where C is the same as in Theorem 4.5.

Theorem 4.7 ([38])**.** *Let $F \in L^2(\Omega)$. Suppose that F is* infinitely Malliavin derivable *and that for every $k \geq 0$, the k-th* Malliavin derivative $D^k F$ of F is square-integrable*. Then the symmetric functions f_n in the chaos decomposition*

$$F = \sum_{n=0}^{\infty} I_n(f_n)$$

can be computed by

$$f_n = \frac{1}{n!} \mathbb{E}[D^n F]$$

These results enable to bound the coefficient functions.

Lemma 4.8. *Let Assumption 2.1 be satisfied, and let μ, σ satisfy the assumptions of Theorem 2.4. Consider the Wiener chaos expansion (2.6) of X_t*

$$X_t = \sum_{n=0}^{\infty} \sum_{|\alpha|=n} x_\alpha(t) \Psi_\alpha.$$

Then, for $m \in \mathbb{N}$, we have

$$\sum_{\ell=0}^{m} \sum_{|\alpha|=\ell} x_\alpha(t)^2 \leq (1 + x_0^2) \left(e^{Cte^{Ct}} - \frac{(Cte^{Ct})^{m+1}}{(m+1)!} \right),$$

where $t \in [0, T]$ and $C = C(K, T)$ is a constant that depends only on T and the regularity of μ and σ.

Proof. Note that

$$X_t = \sum_{n=0}^{\infty} \sum_{|\alpha|=n} x_\alpha(t)\Psi_\alpha = \sum_{n=0}^{\infty} I_n(\xi_n(\mathbf{t}^n;t))$$

with $\mathbf{t}^n := (t_1,\dots,t_n)$. Since X_t is infinitely Malliavin derivative, by Theorem 4.7 the symmetric kernel functions are given by $\xi_n(\cdot;t) : \mathbf{t}^n \mapsto \frac{1}{n!}\mathbb{E}[D^n_{t_1,\dots,t_n}X_t]$, with n-th order *Malliavin derivative* of X_t denoted by D^nX_t. I_n is the multiple stochastic integral of order n introduced in Definition 2.2. It follows that

$$\begin{aligned}
\sum_{\ell=0}^{m}\sum_{|\alpha|=\ell} x_\alpha(t)^2 &= \sum_{\ell=0}^{m} \mathbb{E}[(I_\ell(\xi_\ell(\mathbf{t}^\ell;t)))^2] \\
&= \sum_{\ell=0}^{m} \ell!\langle \xi_\ell(\mathbf{t}^\ell;t), \xi_\ell(\mathbf{t}^\ell;t)\rangle_{L^2([0,T]^\ell)} \\
&= \sum_{\ell=0}^{m} \int^{(\ell);t} \mathbb{E}[(D^\ell_{t_1,\dots,t_\ell}X_t)^2]\,\mathrm{d}\mathbf{t}^\ell \\
&\le (1+x_0^2)\sum_{\ell=0}^{m} C^\ell e^{C\ell t} \int^{(\ell);t} \mathrm{d}\mathbf{t}^\ell \\
&= (1+x_0^2)\sum_{\ell=0}^{m} \frac{(Cte^{Ct})^\ell}{\ell!},
\end{aligned}$$

with $C = C(K,T)$ and $\int^{(\ell);t} f(\cdot)\,\mathrm{d}\mathbf{t}^\ell := \int_0^t \int_0^{t_\ell} \cdots \int_0^{t_2} f(\cdot)\,\mathrm{d}t_1 \dots \mathrm{d}t_\ell$. The inequality is derived by using Proposition 4.6 and Theorem 4.5. Now, note that $\sum_{\ell=0}^{m} \frac{(Cte^{Ct})^\ell}{\ell!} \le e^{Cte^{Ct}} - \frac{(Cte^{Ct})^{m+1}}{(m+1)!}$ using the Taylor-Lagrange formula. With this, we obtain

$$\sum_{\ell=0}^{m}\sum_{|\alpha|=\ell} x_\alpha(t)^2 \le \left((1+x_0^2)(e^{Cte^{Ct}} - \frac{(Cte^{Ct})^{m+1}}{(m+1)!}\right). \qquad \square$$

We can now bound the approximation error.

Lemma 4.9 (Upper bound of approximation error)**.** *Let Assumption 2.1 be satisfied and let μ, σ satisfy the assumptions of Theorem 2.4. Then, the* approximation error (4.7b) *satisfies*

$$\begin{aligned}
\hat{E}^2_{\text{Approx}} \le \min_{(m,n)\in J_p} \mathbb{E}\left[\sum_{j=1}^{p} |\Psi_{k_j^*} - \widetilde{\Psi}_j|^2)\right] \\
\times (1+x_0^2)\left(A(K,T) - \frac{1}{(m+1)!}\frac{T(CT)^{2(m+1)}}{2(m+1)+1}\left(1+\frac{1}{CT}\right)^{m+1}\right),
\end{aligned}$$

with constants $A(K,T) := \int_0^T e^{Cte^{Ct}}\,\mathrm{d}t$ and $C = C(K,T)$.

Proof. By the Cauchy-Schwarz inequality, it follows,

$$\begin{aligned}
\left|\sum_{j=1}^{p} x_{k_j^*}(t)\Psi_{k_j^*} - \sum_{j=1}^{p} x_{k_j^*}(t)\widetilde{\Psi}_j\right|^2 &= \left|\sum_{j=1}^{p} x_{k_j^*}(t)(\Psi_{k_j^*} - \widetilde{\Psi}_j)\right|^2 \\
&\le \left(\sum_{j=1}^{p} x_{k_j^*}(t)^2\right)\left(\sum_{j=1}^{p} |\Psi_{k_j^*} - \widetilde{\Psi}_j|^2\right).
\end{aligned}$$

Taking the expectation and integrating with respect to t leads to

$$\int_0^T \mathbb{E}\left[\left|\sum_{j=1}^p x_{k_j^*}(t)\Psi_{k_j^*} - \sum_{j=1}^p x_{k_j^*}(t)\widetilde{\Psi_j}\right|^2\right] \mathrm{d}t \leq \int_0^T \left(\sum_{j=1}^p x_{k_j^*}(t)^2\right) \mathbb{E}\left[\sum_{j=1}^p |\Psi_{k_j^*} - \widetilde{\Psi_j}|^2\right] \mathrm{d}t.$$

The next step is to find an upper bound of $\int_0^T \left(\sum_{j=1}^p x_{k_j^*}(t)^2\right) \mathrm{d}t$. By Lemma 4.8 with $(m,n) \in J_p$ we have

$$\sum_{\ell=0}^m \sum_{|\alpha|=\ell} x_\alpha(t)^2 \leq (1+x_0^2)\left(e^{Cte^{Ct}} - \frac{(Cte^{Ct})^{m+1}}{(m+1)!}\right).$$

Integrating with respect to t results in

$$\begin{aligned}
\int_0^T \sum_{\ell=0}^m \sum_{|\alpha|=\ell} x_\alpha(t)^2 \,\mathrm{d}t &\leq (1+x_0^2)\left(\underbrace{\int_0^T e^{Cte^{Ct}} \,\mathrm{d}t}_{=:A(K,T)} - \int_0^T \frac{(Cte^{Ct})^{m+1}}{(m+1)!} \,\mathrm{d}t\right) \\
&= (1+x_0^2)\left(A(K,T) - \frac{1}{(m+1)!}\sum_{\ell=0}^{m+1}\binom{m+1}{\ell}\int_0^T (Ct)^{2(m+1)-\ell}\,\mathrm{d}t\right) \\
&= (1+x_0^2)\left(A(K,T) - \frac{1}{(m+1)!}\sum_{\ell=0}^{m+1}\binom{m+1}{\ell}\frac{T(CT)^{2(m+1)-\ell}}{2(m+1)-k+1}\right) \\
&\leq (1+x_0^2)\left(A(K,T) - \frac{1}{(m+1)!}\sum_{\ell=0}^{m+1}\binom{m+1}{\ell}\frac{T(CT)^{2(m+1)-\ell}}{2(m+1)+1}\right) \\
&= (1+x_0^2)\left(A(K,T) - \frac{1}{(m+1)!}\frac{T(CT)^{2(m+1)}}{2(m+1)+1}\left(1+\frac{1}{CT}\right)^{m+1}\right).
\end{aligned}$$

By definition of k^* it thus follows for all $(m,n) \in J_p$ that

$$\int_0^T \left(\sum_{j=1}^p x_{k_j^*}(t)^2\right) \mathrm{d}t \leq (1+x_0^2)\left(A(K,T) - \frac{1}{(m+1)!}\frac{T(CT)^{2(m+1)}}{2(m+1)+1}\left(1+\frac{1}{CT}\right)^{m+1}\right).$$

Combining the above results, we obtain

$$\begin{aligned}
\hat{E}^2_{\text{approx}} &\leq \mathbb{E}\left[\sum_{j=1}^p |\Psi_{k_j^*} - \widetilde{\Psi_j}|^2\right] \\
&\quad \times \min_{(m,n)\in J_p} (1+x_0^2)\left(A(K,T) - \frac{1}{(m+1)!}\frac{T(CT)^{2(m+1)}}{2(m+1)+1}\left(1+\frac{1}{CT}\right)^{m+1}\right).
\end{aligned}$$

□

The approximation of the Ψ_j by neural networks according to Theorem 3.3 yields the following result.

Corollary 4.10 (Deep ReLU neural networks approximation of polynomials chaos)**.** *Let $p \in \mathbb{N}$. For any $\varepsilon \in (0, e^{-1}/p)$ there exists a neural network Φ_ε such that*

$$\mathbb{E}\left[\sum_{j=1}^p |\Psi_{k_j^*} - (R_\sigma \Phi_\varepsilon)_j|^2\right] \leq \varepsilon$$

with complexity given by

$$\begin{aligned}\operatorname{size}(\Phi_\varepsilon) &\le Cp \max_{j\in\{1,\dots,p\}} |k_j^*|^3 \log(1+|k_j^*|)|k_j^*|_0^2 \log(p\varepsilon^{-1}),\\ \operatorname{depth}(\Phi_\varepsilon) &\le C \max_{j\in\{1,\dots,p\}} |k_j^*| \log(1+|k_j^*|)^2 |k_j^*|_0 \log(1+|k_j^*|_0)\log(p\varepsilon^{-1}).\end{aligned}$$

Proof. Recall that $k^* := \arg\min_{k=(k_1,\dots,k_p)\in\mathcal{I}^p} \int_0^T \mathbb{E}[|X_t - \sum_j x_{k_j}(t)\Psi_{k_j}(W)|^2]\,\mathrm{d}t$. Let $j \in \{0,\dots,p\}$. By Theorem 3.3 (take $\Lambda = \{k_j^*\}$) there exists a neural network $\Phi_{\varepsilon,j}$ such that

$$\|\Psi_{k_j^*} - R_\sigma \Phi_{\varepsilon,j}\|_{L^2_\mu(\mathbb{R}^{|k_j^*|_0})} \le \frac{\varepsilon}{p}.$$

Moreover, there exists a positive constant C (independent of $|k_j^*|$, $|k_j^*|_0$, ε and of p) such that

$$\begin{aligned}\operatorname{size}(\Phi_{\varepsilon,j}) &\le C|k_j^*|^3 \log(1+|k_j^*|)|k_j^*|_0^2 \log(p\varepsilon^{-1}),\\ \operatorname{depth}(\Phi_{\varepsilon,j}) &\le C|k_j^*| \log(1+|k_j^*|)^2 |k_j^*|_0 \log(1+|k_j^*|_0)\log(p\varepsilon^{-1}).\end{aligned}$$

The result follows by parallelisation as in Proposition 3.2. □

4.4 Reconstruction error

For the following results, we consider the common activation function $\sigma = \mathrm{ReLU}$. The last error term in (4.6) is the approximation of the deterministic coefficient functions in the Wiener chaos expansion. We show that they can be approximated by neural networks due to the regularity of the corresponding ODE trajectories and Theorem 3.5.

Corollary 4.11 (Approximation of ODEs)**.** *Let $f : [t_0,t_1]\times\mathbb{R}^m \to \mathbb{R}^m \in C^k$ and Lipschitz with respect to the second variable, $k \in \mathbb{N}$, $p > 0$, and $x_0 \in \mathbb{R}^m$. Consider the Cauchy problem with $X : [t_0,t_1] \to \mathbb{R}^m$,*

$$\frac{\mathrm{d}X}{\mathrm{d}t}(t) = f(t,X(t)), \quad X(t_0) = x_0. \tag{4.9}$$

The ODE (4.9) has a unique solution $X : [t_0,t_1] \to \mathbb{R}^m \in C^{k+1}$ and for any $\varepsilon \in (0, (m(t_1-t_0))^{1/p}/2)$ there exists a neural network $\Phi_\varepsilon^X \in \mathcal{N}_{1,m}$ such that

$$\begin{aligned}\|R_\sigma(\Phi_\varepsilon^X) - X\|_{L^p([t_0,t_1])} &< \varepsilon,\\ \|R_\sigma(\Phi_\varepsilon^X)\|_\infty &\le K,\end{aligned}$$

where $K = K(t_0,t_1,X)$ is a constant, and

$$\begin{aligned}\operatorname{depth}(\Phi_\varepsilon^X) &\le (2+\lceil \log_2(k+1)\rceil)(12+k),\\ \operatorname{size}(\Phi_\varepsilon^X) &\le mc\left(\frac{\varepsilon}{(t_1-t_0)^{1/p}}\right)^{-\frac{1}{k+1}}.\end{aligned}$$

Proof. **Case** $m = 1$.
Under the assumptions on f, the problem (4.9) has a unique solution $X : [t_0,t_1] \to \mathbb{R}$ by the Cauchy-Lipschitz (Picard-Lindelöf) theorem. By induction on k it can be shown that $X \in C^{k+1}$. Note that the solution X is $(k,1)$-Hölder continuous. Let

$$T : \begin{cases} [-1/2,1/2] & \longrightarrow [t_0,t_1] \\ x & \longmapsto (x+\frac{1}{2})(t_1-t_0)+t_0 \end{cases}$$

which is a diffeomorphism, and define $Y = X \circ T : [-1/2, 1/2] \to \mathbb{R}$ which is $(k,1)$-Hölder continuous on $[-1/2,1/2]$. Let $B := \max_{\alpha\in\{0,\dots,k\}} \left\| \frac{\mathrm{d}^\alpha Y}{\mathrm{d}x^\alpha} \right\|_\infty$. Applying Theorem 3.5, there exists a constant $c = c(k,B) > 0$ and a neural network Φ_ε^Y such that

$$\begin{aligned}\operatorname{depth}(\Phi_\varepsilon^Y) &\leq (2 + \lceil \log_2(k+1) \rceil)(12+k),\\ \operatorname{size}(\Phi_\varepsilon^Y) &\leq c \left(\frac{\varepsilon}{(t_1-t_0)^{1/p}} \right)^{-\frac{1}{k+1}}.\end{aligned}$$

Moreover,

$$\begin{aligned}\|R_\sigma(\Phi_\varepsilon^Y) - Y\|_{L^p([-1/2,1/2])} &< \frac{\varepsilon}{(t_1-t_0)^{1/p}},\\ \|R_\sigma(\Phi_\varepsilon^Y)\|_\infty &\leq \lceil B \rceil.\end{aligned}$$

Now, note that

$$\begin{aligned}\|R_\sigma(\Phi_\varepsilon^Y) - Y\|^p_{L^p([-1/2,1/2])} &= \frac{1}{t_1-t_0} \int_{t_0}^{t_1} (R_\sigma(\Phi_\varepsilon^Y)(T^{-1}(x)) - Y(T^{-1}(x)))^p \,\mathrm{d}x,\\ &= \frac{1}{t_1-t_0} \|R_\sigma(\Phi_\varepsilon^Y) \circ T^{-1} - X\|^p_{L^p([t_0,t_1])}.\end{aligned}$$

If we write $\Phi_\varepsilon^Y = ((W^1,b^1),\dots,(W^L,b^L))$ then

$$\Phi_\varepsilon^X = ((\frac{1}{t_1-t_0}W^1, b^1 - \left(\frac{t_0}{t_1-t_0} + \frac{1}{2}\right) W^1), (W^2,b^2),\dots,(W^L,b^L))$$

satisfies $R_\sigma \Phi_\varepsilon^Y \circ T^{-1} = R_\sigma \Phi_\varepsilon^X$ and $\operatorname{size}(\Phi_\varepsilon^Y) = \operatorname{size}(\Phi_\varepsilon^X)$. Using the previous equation, we deduce

$$\|R_\sigma(\Phi_\varepsilon^X) - X\|_{L^p([t_0,t_1])} \leq \varepsilon.$$

Case $m > 1$.
Let $(\Phi_\varepsilon^{X_k})_{k=1}^m$ be m neural networks that approximate X_k with accuracy $\frac{\varepsilon}{m^{1/p}}$ in the L^p norm. Consider $\Phi_\varepsilon^X = P((\Phi_\varepsilon^{X_k})_{k=1}^m)$ as a parallelisation according to Proposition 3.2 with $\operatorname{size}(\Phi_\varepsilon^X) = \sum_{k=1}^m \operatorname{size}(\Phi_\varepsilon^{X_k})$ and $R_\sigma \Phi_\varepsilon^X = (R_\sigma \Phi_\varepsilon^{X_k})_{k=1}^m$. Then,

$$\begin{aligned}\|R_\sigma \Phi_\varepsilon^X - X\|^p_{L^p([t_0,t_1])} &\leq \sum_{j=1}^m \|R_\sigma \Phi_\varepsilon^{X_k} - X_k\|^p_{L^p([t_0,t_1])}\\ &\leq m \left(\frac{\varepsilon}{m^{1/p}}\right)^p\\ &= \varepsilon^p.\end{aligned}$$

□

Lemma 4.12. *With Assumption 2.1 and n-times continuously differentiable μ, σ, e_j with respect to their variables, for any $\varepsilon \in (0, \sqrt{pT}/2)$ there exists a neural network $\Phi_\varepsilon \in \mathcal{N}_{1,p}$ such that*

$$\sum_{j=1}^p \|x_{k_j^*} - \widetilde{x_j}\|^2_{L^2([0,T])} \leq \varepsilon^2.$$

The reconstruction error *can be bounded like*

$$\hat{E}^2_{\mathit{Recon}} \leq \varepsilon^2 \left(\sum_{j=1}^{p} \mathbb{E}[\widetilde{\Psi_j}^2] \right)$$

and

$$\begin{aligned} \operatorname{depth}(\Phi_\varepsilon) &\leq (2 + \lceil \log_2(n+1) \rceil)(12+n), \\ \operatorname{size}(\Phi_\varepsilon) &\leq pc \left(\frac{\varepsilon}{\sqrt{T}} \right)^{-\frac{1}{n+1}}. \end{aligned}$$

Proof. By the Cauchy-Schwarz inequality one has

$$\begin{aligned} \left| \sum_{j=1}^{p} x_{k_j^*}(t) \widetilde{\Psi_j} - \sum_{j=1}^{p} \widetilde{x_j}(t) \widetilde{\Psi_j} \right|^2 &= \left| \sum_{j=1}^{p} (x_{k_j^*}(t) - \widetilde{x_j}(t)) \widetilde{\Psi_j} \right|^2 \\ &\leq \left(\sum_{j=1}^{p} |x_{k_j^*}(t) - \widetilde{x_j}(t)|^2 \right) \left(\sum_{j=1}^{p} \widetilde{\Psi_j}^2 \right). \end{aligned}$$

Taking the expectation and integrating with respect to t leads to

$$\int_0^T \mathbb{E} \left[\left| \sum_{j=1}^{p} x_{k_j^*}(t) \widetilde{\Psi_j} - \sum_{j=1}^{p} \widetilde{x_j}(t) \widetilde{\Psi_j} \right|^2 \right] \mathrm{d}t \leq \left(\sum_{j=1}^{p} \| x_{k_j^*} - \widetilde{x_j} \|^2_{L^2([0,T])} \right) \left(\sum_{j=1}^{p} \mathbb{E}[\widetilde{\Psi_j}^2] \right).$$

Recall that μ, σ, e_j are n-times continuously differentiable with respect to their variables. By Corollary 4.11 for any $\varepsilon \in (0, \sqrt{pT}/2)$ there exists a neural network $\Phi_\varepsilon \in \mathcal{N}_{1,p}$ such that

$$\begin{aligned} \operatorname{depth}(\Phi_\varepsilon) &\leq (2 + \lceil \log_2(n+1) \rceil)(12+n), \\ \operatorname{size}(\Phi_\varepsilon) &\leq pc \left(\frac{\varepsilon}{\sqrt{T}} \right)^{-\frac{1}{n+1}}. \end{aligned}$$

Moreover,

$$\sum_{j=1}^{p} \| x_{k_j^*} - \widetilde{x_j} \|^2_{L^2([0,T])} \leq \varepsilon^2. \qquad \square$$

4.5 D-dimensional SDE

Let $H = L^2([0,T]; \mathbb{R}^d) \cong L^2([0,T]) \otimes \mathbb{R}^d$. Let $B := (B_t^1, \ldots, B_t^d)_{t \in [0,T]}$ be a d-dimensional Brownian motion defined on $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t \in [0,T]}, \mathbb{P})$, where $(\mathcal{F}_t)_t$ is its natural filtration and $\mathcal{F} := \sigma(W(h) : h \in H)$. Then,

$$W(h) := \sum_{k=1}^{d} \int_0^T h^k(t) \, \mathrm{d}W_t^k$$

is an *isonormal Gaussian process* for H.

Let $(\phi_{ij})_{j=1,i\geq 0}^{j=d}$ be an orthonormal basis of $L^2([0,T];\mathbb{R}^d)$. For instance, $\phi_{ij} = \psi_i \otimes e_j$, where $(\psi_i)_{i\geq 0}$ is a orthonormal basis of $L^2([0,T])$ and $(e_j)_{j=1}^d$ is the canonical basis of $\mathbb{R}^d$. Then, any random variable $F \in L^2(\Omega, \mathcal{F}, \mathbb{P})$ admits the following Wiener chaos expansion

$$F = \sum_{k=0}^{\infty} \sum_{\substack{|\alpha|=k \\ \alpha\in\mathcal{I}^d}} f_\alpha \left(\underbrace{\prod_{j=1}^{d} \prod_{i=1}^{\infty} H_{\alpha_i^j}(W(\phi_{ij}))}_{=:\Psi_\alpha} \right).$$

Consider the following d-dimensional Itô process

$$\mathrm{d}\mathbf{X}_t = \mu(t, \mathbf{X}_t)\,\mathrm{d}t + \sigma(t, \mathbf{X}_t) \cdot \mathrm{d}\mathbf{B}_t,$$

which in integral form reads

$$\mathbf{X}_t = \mathbf{X}_0 + \int_0^t \mu(s, \mathbf{X}_s)\,\mathrm{d}s + \int_0^t \sigma(s, \mathbf{X}_s) \cdot \mathrm{d}\mathbf{B}_s.$$

The Wiener chaos expansion for each component X_t^j of $\mathbf{X}_t$ is given by

$$X_t^j = \sum_{k=0}^{\infty} \sum_{\substack{|\alpha|=k \\ \alpha\in\mathcal{I}^d}} x_\alpha^j(t) \left(\prod_{j=1}^{d} \prod_{i=1}^{\infty} H_{\alpha_i^j}(W(\phi_{ij})) \right).$$

Following the proof of Theorem 2.3, it is possible to show that the $x_\alpha^j(t)$ satisfy a system of ordinary differential equation.

Suppose that we want to approximate each component X_t^j with p coefficients. Then, the analysis for the approximation of the coefficients $x_\alpha^j(t)$ Lemma 4.12 does not change qualitatively since by Proposition 3.2 the size is only multiplied by d. Since we have d times more Hermite polynomials the polynomial chaos Ψ_α, by Corollary 4.10 the size of the network again is multiplied by d.

5 Numerical experiments

This section illustrates that our SDEONet architecture is able to approximate the stochastic process $(X_t)_t$ at any time t in numerical computations with a reasonable number of parameters. To enforce the learning of the initial condition X_0, a second term is added to the loss, which then becomes

$$\mathcal{L}(\theta) = \frac{1}{B} \left(\sum_{i=1}^{B} \|X_{t_i} - R_\sigma \mathcal{N}_\theta^{m,p}(W^i, t_i)\|_2^2 + \sum_{i=1}^{B} \|X_0 - R_\sigma \mathcal{N}_\theta^{m,p}(W^i, 0)\|_2^2 \right). \tag{5.1}$$

To assess the performance of our model, we use different metrics. These are computed at each time step on the time grid to check if our model is able to approximate the stochastic process at any time t. The first two are the absolute L^2 error $\|X_t - \tilde{X}_t\|_{L^2}$ and the relative L^2 error $\frac{\|X_t - \tilde{X}_t\|_{L^2}}{\|X_t\|_{L^2}}$. They are approximated by a Monte-Carlo estimation

$$\|F\|_{L^2} \approx \left(\frac{1}{N} \sum_{i=1}^{N} |F^i|^2 \right)^{1/2},$$

where $(F_i)_{i=1}^N$ are realisations of F.

Another reasonable metric to consider is the Wasserstein 2-distance defined by

$$\mathcal{W}_2(\mu,\nu)^2 := \min_{\gamma\in\Pi(\mu,\nu)} \int_{\mathbb{R}^d} \|x-y\|_2^2 \,\mathrm{d}\gamma(x,y),$$

where $\Pi(\mu,\nu) := \left\{\gamma \in \mathcal{P}(\mathbb{R}^d\times\mathbb{R}^d) : (\pi_0)_\sharp\gamma = \mu, (\pi_1)_\sharp\gamma = \nu\right\}$ is the *transport plan* and π_0 and π_1 are the two projections of $\mathbb{R}^d\times\mathbb{R}^d$ onto its factors. When $d=1$, it is possible to approximate it by considering the empirical measures $\mu_n = \frac{1}{n}\sum_{i=1}^n \delta_{X_i}$, $\nu_n = \frac{1}{n}\sum_{i=1}^n \delta_{Y_i}$ and then to compute

$$\mathcal{W}_2(\mu_n,\nu_n)^2 = \frac{1}{n}\sum_{i=1}^n \|X_{(i)} - Y_{(i)}\|_2^2,$$

where $X_{(1)} \le X_{(2)} \le \cdots \le X_{(n)}$. For higher dimensions $d>1$, the computation of W_2 is quite elaborate. There are methods to approximate it, e.g. by the well-known Sinkhorn algorithm [8, 6].

Definition 5.1 (Entropy regularised optimal transport cost [6])**.** *Let μ,ν be two probability measures on $\mathbb{R}^d$ with finite second-moment. Then, the* entropy regularised optimal transport cost *is defined as*

$$T_\lambda(\mu,\nu) := \min_{\gamma\in\Pi(\mu,\nu)} \mathbb{E}_{(X,Y)\sim\gamma}\left[\|X-Y\|_{\ell^2}^2\right] + 2\lambda H(\gamma,\mu\otimes\nu),$$

where $\Pi(\mu,\nu) := \left\{\gamma\in\mathcal{P}(\mathbb{R}^d\times\mathbb{R}^d) : \int\gamma(x,\cdot)\,\mathrm{d}x = \nu, \int\gamma(\cdot,y)\,\mathrm{d}y = \mu\right\}$ is the set of transport plans *between μ and ν, $\lambda\ge 0$ is the* regularisation parameter*, and $H(\gamma,\mu\otimes\nu)$ is the* relative entropy *(or* Kullback-Leibler divergence*) of γ with respect to $\mu\otimes\nu$ defined by*

$$H(\gamma,\mu\otimes\nu) := \int_{\mathbb{R}^d\times\mathbb{R}^d} \log\left(\frac{\mathrm{d}\gamma(x,y)}{\mathrm{d}\mu(x)\,\mathrm{d}\nu(y)}\right)\mathrm{d}\gamma(x,y).$$

Note that $T_0(\mu,\nu) = \mathcal{W}_2^2(\mu,\nu)$ and that the choice $T_\lambda(\hat\mu_n,\hat\nu_n)$ is not optimal since it introduces a large bias. The *Sinkhorn divergence* defined by

$$S_\lambda(\mu,\nu) := T_\lambda(\mu,\nu) - \frac{1}{2}(T_\lambda(\mu,\mu) + T_\lambda(\nu,\nu))$$

is an estimator of $\mathcal{W}_2(\mu,\nu)^2$.

5.1 1D processes

In the next experiments the model is defined by $m=32$, $p=64$, 2 hidden layers of 256 neurons each. The model is learned on a dataset of 20,000 samples of X_t with $t\sim U(0,T)$ during 30 epochs with a learning rate of $3\cdot 10^{-4}$ with the Adam optimizer [22] and a batch size of 64.

5.1.1 Ornstein-Uhlenbeck process

The Ornstein-Uhlenbeck process is a crucial stochastic process in the area of mathematical physics and stochastic calculus. It is a continuous-time stochastic process that finds extensive application in emulating a diverse range of phenomena across multiple fields such as physics, finance, biology, and engineering. This process is useful for modelling mean-reversing behaviour, where a variable tends to

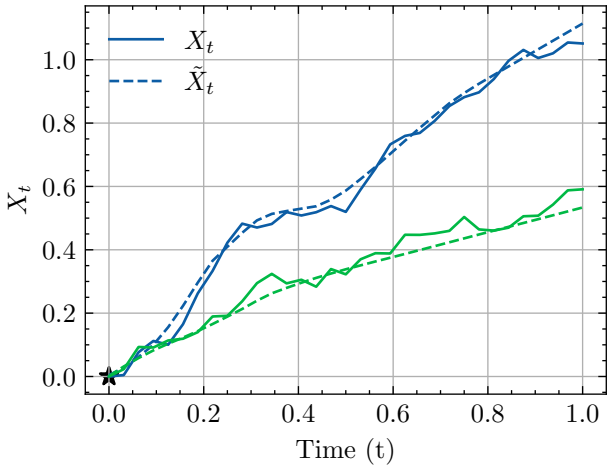


Figure 6: Trajectories of the true process X_t and the approximation $\tilde{X}_t$ (OU).

return to its mean over time. This makes it a valuable tool for understanding and modelling stable and self-correcting processes. It is defined by

$$\mathrm{d}X_t = -\theta(X_t - \mu)\,\mathrm{d}t + \sigma\,\mathrm{d}W_t,$$

where $\theta > 0$, μ and $\sigma > 0$ are parameters. Using Itô's formula, it is possible to get an explicit expression of X_t given X_0 by

$$X_t = X_0 e^{-\theta t} + \mu(1 - e^{-\theta t}) + \frac{\sigma}{\sqrt{2\theta}} W_{1-e^{-2\theta t}}.$$

all figures have to be as readable as fig 6 -> larger fonts, thicker lines

For the numerical experiments, we chose $\theta = 1$, $\mu = 1.2$, and $\sigma = 1.3$.

Examining the trajectories depicted in Figure 6, we observe that the approximation $\tilde{X}_t$ exhibits a notable smoother behaviour compared to the true stochastic process X_t. This behaviour is likely due to a small dimension of polynomial chaos. Moreover, we can discern the impact of the additional term incorporated into the loss function. This supplementary term plays a crucial role in aiding the model to effectively learning the initial state, represented by X_0, which is confirmed by Figure 7a. Figure 7b also shows that the model is able to learn the stochastic process at each time step t very accurately since the random variables X_t and $\tilde{X}_t$ are close in distribution.

5.1.2 Geometric Brownian motion

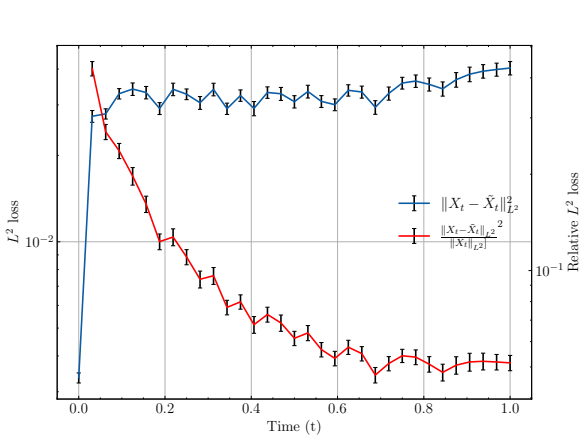
The Geometric Brownian motion (GBM) is a widely used stochastic process in finance, mathematical modelling, and statistical physics. In particular, this process is crucial for understanding and describing the pricing of financial assets and to model stock prices. GBM represents an extension of the classic Brownian movement, which incorporates exponential growth and is characterised by its capacity to capture the innate uncertainty and volatility linked with genuine financial markets.

The Geometric Brownian motion is defined by

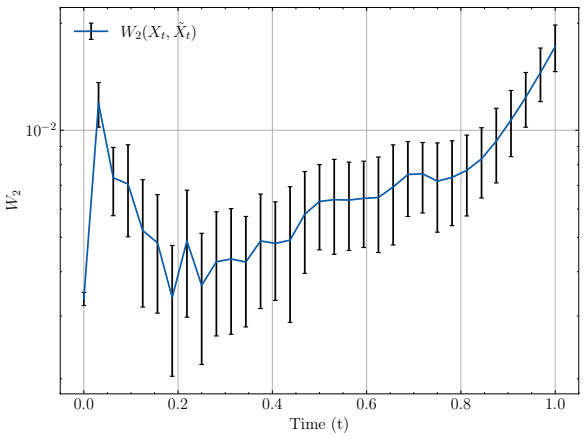
$$\mathrm{d}X_t = \mu X_t\,\mathrm{d}t + \sigma X_t\,\mathrm{d}W_t,$$

where μ and $\sigma > 0$ are parameters. Using Itô's formula, it is possible to obtain an explicit expression of X_t given X_0, namely

$$X_t = X_0 \exp\left(\left(\mu - \frac{\sigma^2}{2}\right)t + \sigma W_t\right).$$



(a) L^2 loss (blue) and relative L^2 loss (red) over time.



(b) $W_2(X_t, \tilde{X}_t)$

Figure 7: L^2 loss and Wasserstein 2-distance over time for the Ornstein-Uhlenbeck process, computed over 2,000 samples and averaged over 100 independent realisations. The error bars correspond to 3σ.

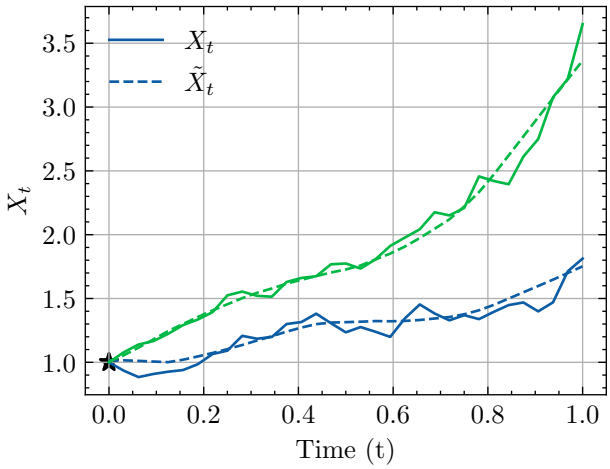


Figure 8: Trajectories of the true process X_t and the approximation $\tilde{X}_t$ (GBM).

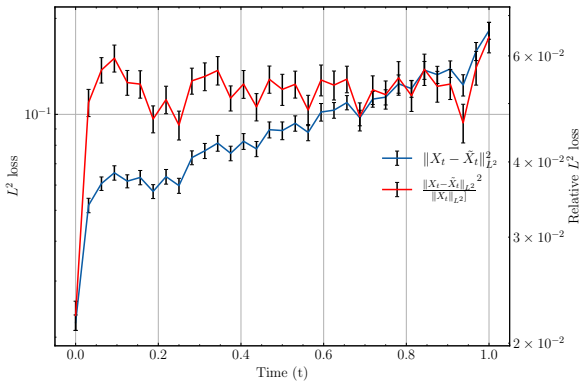
For the numerical experiments, we chose $\mu = 1.0$ and $\sigma = 0.3$.

The plot in Figure 8 shows a similar behaviour as for the Ornstein-Uhlenbeck process above. However, we see that the approximation is slightly worse due to the nature of the Geometric Brownian motion. In practice, we also notice that for large σ it becomes more difficult to learn the stochastic process.

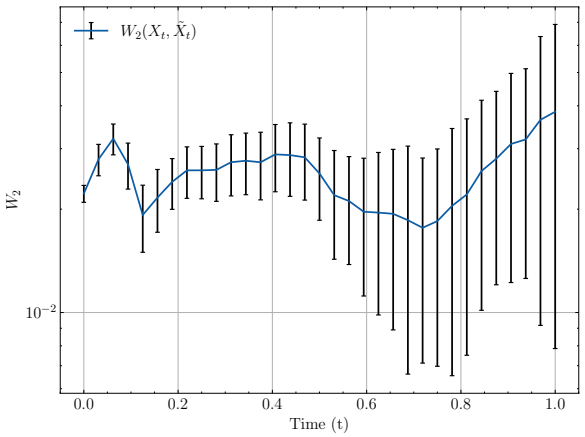
Figure 9a shows that with this choice of parameters μ and σ the model is able to learn the stochastic process with a small L^2 and relative L^2 error. Moreover, Figure 9b illustrates that the random variables X_t and $\tilde{X}_t$ are close in distribution at each time step t. This can be considered a more appropriate metric to assess the accuracy of the learned operator model.

5.2 Multi-dimensional

For the multidimensional experiment, we consider the Langevin process with the potential of the multivariate Normal distribution. Consider a particle subject to the force induced by a potential $V \in C^1(\mathbb{R}^d)$ (defined by $-\nabla V$), a friction and a random white noise. Let X_t be the position of the particle at time



(a) L^2 loss (blue) and relative L^2 loss (red) over time for the approximation of the Geometric Brownian Motion.



(b) $W_2(X_t, \tilde{X}_t)$

Figure 9: L^2 loss and Wasserstein 2-distance over time for the Geometric Brownian motion computed over 2,000 samples and averaged over 100 independent realisations. The error bars correspond to 3σ.

t, k the Boltzmann constant and T the temperature. Then, Newton's equation of motion leads to

$$m\frac{\mathrm{d}^2 X_t}{\mathrm{d}t^2} = -\nabla V(X_t) - \lambda m\frac{\mathrm{d}X_t}{\mathrm{d}t} + \sqrt{kT}\frac{\mathrm{d}B_t}{\mathrm{d}t},$$

where λ is friction coefficient and $(B_t)_{t\geq 0}$ is a standard Brownian motion. This is a second-order stochastic differential equation with initial conditions X_0 and $\frac{\mathrm{d}X_t}{\mathrm{d}t}(0)$, which can be written as

$$\begin{aligned}\mathrm{d}X_t &= V_t\,\mathrm{d}t,\\ \mathrm{d}V_t &= -(\nabla V(X_t) + \lambda m V_t)\,\mathrm{d}t + \sqrt{kT}\,\mathrm{d}B_t.\end{aligned}$$

If we assume that the friction coefficient λ is large and the particle moves slowly enough that the acceleration term can be neglected, then the stochastic differential equation can be simplified to

$$\mathrm{d}X_t = -\nabla V(X_t)\,\mathrm{d}t + \sqrt{2}\,\mathrm{d}B_t,$$

which is commonly called a *Langevin process*. Now, considering a large ensemble of particles that evolve independently under this motion, it is necessary to establish if the distribution of particles converges to a defined limit as $t \to \infty$. This question requires some assumptions regarding the potential V, which should prevent the particles from escaping infinitely. Typically, it is assumed that V is m-strongly convex and that the initial distribution has a finite second moment. With these assumptions, it can be shown that there exists a stationary distribution [35]. When writing the Fokker-Planck equation of this stochastic differential equation, we have that the probability density $p(\cdot, t)$ of X_t evolves according to

$$\frac{\partial p}{\partial t}(x,t) = \mathrm{div}(\nabla V(x)p(x,t) + \nabla_x p(x,t)),$$

where div is the divergence operator defined by $\mathrm{div}(f) := \sum_{j=1}^{d} \partial_j f$. From this equation, we immediately see that

$$p_\infty := \frac{\exp(-V)}{\int \exp(-V)}$$

is stationary. It can also be shown that the convergence is exponentially fast.

Consider the multivariate normal distribution with mean μ, covariance matrix Σ and probability density function

$$\gamma(x) := (2\pi)^{-d/2} \det(\Sigma)^{-1/2} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right).$$

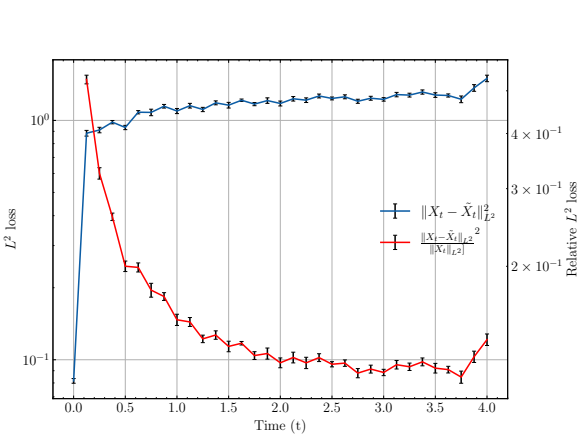
The associated potential $V := -\log\gamma$ is given by

$$V(x) = \frac{d}{2}\ln(2\pi) + \frac{1}{2}\ln(\det\Sigma) + \frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu).$$

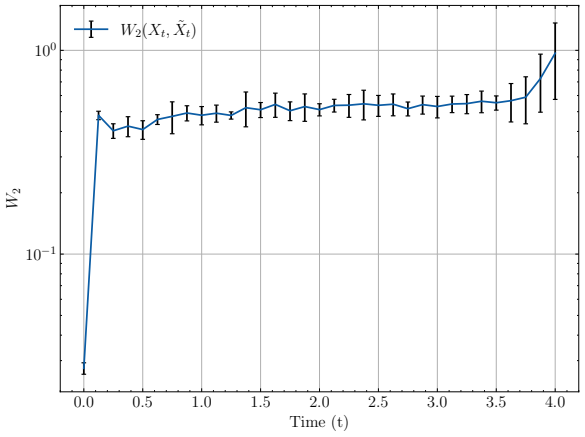
Therefore, the respective Langevin process can be written as

$$\mathrm{d}X_t = -\Sigma^{-1}(X_t-\mu)\,\mathrm{d}t + \sqrt{2}\,\mathrm{d}B_t.$$

For the numerical experiment, we have chosen $d = 5$, $\Sigma = I$ and $\mu = 2((-1)^j j)_{j=0}^{d-1}$. The neural network has the same architecture as in the one dimensional case, except that we now consider 64 coefficients for each component of the approximation $\tilde{X}_t$.



(a) L^2 loss (blue) and relative L^2 loss (red)



(b) $W_2(X_t, \tilde{X}_t)$ distance

Figure 10: L^2 loss and Wasserstein 2-distance over time for the Langevin process, computed over 2,000 samples and averaged over 10 independent realisations. The error bars correspond to 3σ.

The error plots in Figure 10 show that the model is still able to approximate the stochastic process X_t in higher dimensions, even though the task is structurally more complicated. We expect that an extended training effort would lead to an even better operator model.

6 Conclusion

In this work, we have developed a new NN architecture called SDEONet to approximate the solution of a SDE (1.1) using the notion of DeepONet [26] and a polynomial chaos expansion (2.6). Classical methods using polynomial chaos expansion for solving SDE [18] struggle to handle the $\binom{p+m}{p}$ coefficients, which increase very quickly when the number of basis elements m or the maximal degree p is increased. It hence is inevitable to devise an appropriate truncation and compression, that still allows for accurate results in practice. The method we have developed is a new strategy to learn a sparse Wiener chaos expansion of the solution of the SDE. The analysis shows that the size of the required neural networks is quite small due to the regularity of the coefficients and the use of Hermite polynomials.

The experiments have shown promising results with small relative L^2 and $\mathcal{W}_2$ errors. However, the stability of the model should be improved when the process has a large variance, like with Geometric Brownian motion. Concerning the multidimensional case, as discussed in Section 4.5, the experiment in Section 5.2 also suggests that our model is able to accurately approximate the solution of a multidimensional SDE without suffering from the "curse of dimensionality".

The presented work motivates the extension to a more complex class of equation, e.g. backward stochastic differential equations (BSDE). These equations are more challenging to solve, especially in high dimensions. Some recent works already have considered Wiener chaos expansions for solving BSDE [4].

References

[1] Christian Bayer, Martin Eigel, Leon Sallandt, and Philipp Trunschke, *Pricing high-dimensional Bermudan options with hierarchical tensor formats*, SIAM Journal on Financial Mathematics **14** (2023), no. 2, 383–406.

[2] Christian Beck, Sebastian Becker, Philipp Grohs, Nor Jaafari, and Arnulf Jentzen, *Solving the kolmogorov pde by means of deep learning*, Journal of Scientific Computing **88** (2021), 1–28.

[3] Sebastian Becker, Patrick Cheridito, and Arnulf Jentzen, *Deep optimal stopping*, The Journal of Machine Learning Research **20** (2019), no. 1, 2712–2736.

[4] Philippe Briand and Céline Labart, *Simulation of BSDEs by Wiener chaos expansion*, **24** (2014), no. 3.

[5] Robert H Cameron and William T Martin, *The orthogonal development of non-linear functionals in series of fourier-hermite functionals*, Annals of Mathematics (1947), 385–392.

[6] Lénaïc Chizat, Pierre Roussillon, Flavien Léger, François-Xavier Vialard, and Gabriel Peyré, *Faster Wasserstein distance estimation with the Sinkhorn divergence*, Proceedings of the 34th International Conference on Neural Information Processing Systems (Red Hook, NY, USA), NIPS'20, Curran Associates Inc., 2020.

[7] Albert Cohen and Ronald DeVore, *Approximation of high-dimensional parametric PDEs*, Acta Numerica **24** (2015), 1–159.

[8] Marco Cuturi, *Sinkhorn distances: Lightspeed computation of optimal transport*, Advances in Neural Information Processing Systems (C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, eds.), vol. 26, Curran Associates, Inc., 2013.

[9] Beichuan Deng, Yeonjong Shin, Lu Lu, Zhongqiang Zhang, and George Em Karniadakis, *Approximation rates of DeepONets for learning operators arising from advection–diffusion equations*, Neural Networks **153** (2022), 411–426.

[10] Martin Eigel, Robert Gruhlke, and David Sommer, *Less interaction with forward models in Langevin dynamics*, arXiv preprint arXiv:2212.11528 (2022).

[11] Martin Eigel, Charles Miranda, Janina Schütte, and David Sommer, *Approximating Langevin Monte Carlo with ResNet-like Neural Network architectures*, arXiv preprint arXiv:2311.03242 (2023).

[12] Oliver G Ernst, Antje Mugler, Hans-Jörg Starkloff, and Elisabeth Ullmann, *On the convergence of generalized polynomial chaos expansions*, ESAIM: Mathematical Modelling and Numerical Analysis **46** (2012), no. 2, 317–339.

[13] Zhiwei Gao, Liang Yan, and Tao Zhou, *Adaptive operator learning for infinite-dimensional bayesian inverse problems*, arXiv preprint arXiv:2310.17844 (2023).

[14] Alfredo Garbuno-Inigo, Nikolas Nüsken, and Sebastian Reich, *Affine invariant interacting Langevin dynamics for Bayesian inference*, SIAM Journal on Applied Dynamical Systems **19** (2020), no. 3, 1633–1658.

[15] Somdatta Goswami, Aniruddha Bora, Yue Yu, and George Em Karniadakis, *Physics-informed deep neural operator networks*, Machine Learning in Modeling and Simulation: Methods and Applications, Springer, 2023, pp. 219–254.

[16] Wolfgang Hackbusch, *Numerical tensor calculus*, Acta numerica **23** (2014), 651–742.

[17] Helge Holden, Bernt Øksendal, Jan Ubøe, Tusheng Zhang, Helge Holden, Bernt Øksendal, Jan Ubøe, and Tusheng Zhang, *Stochastic partial differential equations*, Springer, 1996.

[18] Tony Huschto, Mark Podolskij, and Sebastian Sager, *The asymptotic error of chaos expansion approximations for stochastic differential equations*, (2019), 145–165.

[19] Tony Huschto and Sebastian Sager, *Solving stochastic optimal control problems by a Wiener Chaos approach*, Vietnam Journal of Mathematics **42** (2014), no. 1, 83–113.

[20] Svante Janson, *Gaussian Hilbert spaces*, no. 129, Cambridge university press, 1997.

[21] Ioannis Karatzas and Steven E. Shreve, *Brownian motion and stochastic calculus*, Springer New York, 1998.

[22] Diederik P. Kingma and Jimmy Ba, *Adam: A method for stochastic optimization*, CoRR **abs/1412.6980** (2014).

[23] Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar, *Neural operator: Learning maps between function spaces with applications to pdes*, Journal of Machine Learning Research **24** (2023), no. 89, 1–97.

[24] Samuel Lanthaler, Siddhartha Mishra, and George E Karniadakis, *Error estimates for DeepONets: A deep learning framework in infinite dimensions*, **6** (2022), no. 1.

[25] Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew M. Stuart, and Anima Anandkumar, *Fourier neural operator for parametric partial differential equations*, 9th International Conference on Learning Representations, ICLR 2021, OpenReview.net, 2021.

[26] Lu Lu, Pengzhan Jin, and George Em Karniadakis, *DeepONet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators*, **3** (2021), no. 3, 218–229.

[27] Carlo Marcati and Christoph Schwab, *Exponential convergence of deep operator networks for elliptic partial differential equations*, SIAM Journal on Numerical Analysis **61** (2023), no. 3, 1513–1545.

[28] Paul-André Meyer, *Un cours sur les intégrales stochastiques (exposés 1 à 6)*, Séminaire de probabilités de Strasbourg **10** (1976), 245–400 (fre).

[29] Anthony Nouy, *Low-rank methods for high-dimensional approximation and model order reduction*, Model Reduction and Approximation: Theory and Algorithms **15** (2017), no. 171, 3672148.

[30] David Nualart, *The Malliavin calculus and related topics*, second ed., Probability and its Applications (New York), Springer-Verlag, Berlin, 2006. MR 2200233

[31] Bernt Øksendal, *Stochastic differential equations*, Springer Berlin Heidelberg, 2003.

[32] J. A. A. Opschoor, Ch. Schwab, and J. Zech, *Exponential ReLU DNN expression of holomorphic maps in high dimension*, Constructive Approximation **55** (2021), no. 1, 537–582.

[33] Ivan V Oseledets, *Tensor-train decomposition*, SIAM Journal on Scientific Computing **33** (2011), no. 5, 2295–2317.

[34] Philipp Petersen and Felix Voigtlaender, *Optimal approximation of piecewise smooth functions using deep ReLU neural networks*, Neural Networks **108** (2018), 296–330.

[35] Gareth O. Roberts and Richard L. Tweedie, *Exponential convergence of Langevin distributions and their discrete approximations*, Bernoulli **2** (1996), no. 4, 341 – 363.

[36] Christoph Schwab and Claude Jeffrey Gittelson, *Sparse tensor discretizations of high-dimensional parametric and stochastic pdes*, Acta Numerica **20** (2011), 291–467.

[37] Christoph Schwab and Jakob Zech, *Deep learning in high dimension: Neural network expression rates for analytic functions in* $L^2(\mathbb{R}^d, \gamma_d)$, SIAM/ASA Journal on Uncertainty Quantification **11** (2023), no. 1, 199–234.

[38] Daniel W. Stroock, *Homogeneous chaos revisited*, p. 1–7, Springer Berlin Heidelberg, 1987.

[39] Andrew M Stuart, *Inverse problems: a bayesian perspective*, Acta numerica **19** (2010), 451–559.

[40] Tianping Chen and Hong Chen, *Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems*, **6**, no. 4, 911–917.

[41] 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 Computing Surveys **56** (2023), no. 4, 1–39.