Generic bounds on the approximation error for physics-informed (and) operator learning

Tim De Ryck *

Siddhartha Mishra †

Abstract

We propose a very general framework for deriving rigorous bounds on the approximation error for physics-informed neural networks (PINNs) and operator learning architectures such as DeepONets and FNOs as well as for physics-informed operator learning. These bounds guarantee that PINNs and (physics-informed) DeepONets or FNOs will efficiently approximate the underlying solution or solution operator of generic partial differential equations (PDEs). Our framework utilizes existing neural network approximation results to obtain bounds on more involved learning architectures for PDEs. We illustrate the general framework by deriving the first rigorous bounds on the approximation error of physics-informed operator learning and by showing that PINNs (and physics-informed DeepONets and FNOs) mitigate the curse of dimensionality in approximating nonlinear parabolic PDEs.

1 Introduction

The efficient numerical approximation of partial differential equations (PDEs) is of paramount importance as PDEs mathematically describe an enormous range of interesting phenomena in the sciences and engineering. Machine learning techniques, particularly deep learning, are playing an increasingly important role in this context. For instance, given their universal approximation properties, deep neural networks serve as ansatz spaces for supervised learning of a variety of (parametric) PDEs [19, 70, 40, 53, 54] and references therein. In this setting, large amounts of training data might be required. However, this data is often acquired from expensive computer simulations or physical measurements [53], necessitating the design of learning frameworks that work with limited data. *Physics-informed neural networks (PINNs)*, proposed by [18, 42, 41] and popularized by [67, 68], are a prominent example of such a learning framework as the residual of the underlying PDE is minimized within the class of neural networks and in principle, little (or even no) training data is required. PINNs and their variants have proven to be a very powerful and computationally efficient framework for approximating solutions to PDEs, [69, 51, 55, 65, 76, 34, 35, 61, 59, 60, 2] and references therein.

Often in the context of PDEs, one needs to approximate the underlying solution operator that maps one infinite-dimensional function space into another [27, 39]. As neural networks can only map between finite dimensional spaces, a new field of *operator learning* is emerging wherein novel learning frameworks need to be designed in order to approximate operators. These include deep operator networks (DeepONets) [9, 49] and their variants as well as *neural operators* [39], which generalize neural networks to this setting. A variety of neural operators have been proposed, see [45, 46] but arguably, the most efficient form of neural operators is provided by the so-called *Fourier neural operators* (FNOs) [44]. Both DeepONets and FNOs have been very successfully deployed in

*Seminar for Applied Mathematics (SAM), D-MATH, ETH Zürich, Switzerland

†Seminar for Applied Mathematics (SAM), D-MATH and ETH AI center, ETH Zürich, Switzerland

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

scientific computing [50, 56, 8, 48, 47, 66] and references therein. Finally, one can combine PINNs and operator learning to design physics-informed DeepONets/FNOs [74, 47, 73, 22].

From a theoretical perspective, one needs to provide a rigorous guarantee that the learning framework can approximate the underlying PDE solution (operator) to desired accuracy. More precisely, given an error tolerance $\varepsilon > 0$, we need to rigorously prove that the approximation error of the neural network (operator) can be made smaller than ε. For *efficient approximation*, one has to further ensure that the computational complexity (measured in terms of the size) of the learning architecture grows at most *polynomially* in ε^{-1}. In particular, *exponential* growth has to be ruled out. As neural networks, DeepONets and FNOs are all *universal approximators* [13, 9, 49, 43, 38] of the underlying functions or operators, it is possible to show that the approximation error can be made as small as desired. However, these results do not guarantee efficient approximation as the underlying network size could still grow exponentially with decreasing error, see [77] for neural networks in very high spatial dimensions, [43] for DeepONets and [38] for FNOs. Hence, the real theoretical challenge in this context lies in proving efficient approximation results for the different learning architectures in scientific computing. Such efficient approximation results have mostly been obtained for neural networks in the supervised learning setting e.g. [23, 36, 32, 4, 63] and references therein. In contrast, there is a relative scarcity of such efficient approximation results for PINNs and operator learning with notable exceptions being [25, 15, 16] (for PINNs), [43] (for DeepONets) and [38] (for FNOs). Moreover, the underlying proofs in these works are often on a case-by-case basis and the overall abstract structure is not clearly identified. Finally, no similar rigorous approximation results for physics informed operator learning are available till date.

This paucity of generic efficient approximation results for PINNs and operator learning for PDEs sets the stage for the current paper where our main contribution is to propose a very general *framework* (Section 3) for proving bounds on the approximation error for space-time neural networks, PINNs, DeepONets, FNOs and physics-informed DeepONets and FNOs for very general PDEs. Consequently, we obtain the first rigorous bounds for physics-informed operator learning in literature. Our framework is based on the observation that error estimates for different types of neural network architectures can all be obtained from one another. As error estimates for neural network approximations of PDE solutions at a fixed time are the easiest to obtain, and hence constitute the largest proportion of currently available estimates, we devote particular attention to demonstrating how these available estimates can be used to obtain novel bounds on the approximation error for space-time networks, PINNs and (physics-informed) operator learning. Our results provide a *roadmap* for deriving mathematical guarantees for deep learning methods in scientific computing by simplifying the proofs, as the needed work essentially reduces to verifying a small number of assumptions. We demonstrate how the generic error bounds from Section 3 can be applied in practice in Section 4, among others by giving short alternative proofs for known results and also proving a number of novel results. In particular, we show in Section 4.1 that PINNs can overcome the curse of dimensionality for nonlinear parabolic PDEs such as the Allen-Cahn equation i.e., that the network size does not grow exponentially with increasing spatial dimension. Moreover, dimension-independent convergence rates are also obtained for (physics-informed) DeepONets and FNOs, provided that the PDE solutions are sufficiently smooth. These are the first results of their kind. We note that many of the proofs and some examples are deferred to the supplementary material (**SM**).

2 Preliminaries

2.1 Setting

Given $T > 0$ and $D \subset \mathbb{R}^d$ compact, consider the function $u : [0, T] \times D \to \mathbb{R}^m$, for $m \geq 1$, that belongs to a function space $\mathcal{H}$ and solves the following (time-dependent) PDE,

$$\mathcal{L}_a(u)(t, x) = 0 \quad \text{and} \quad u(x, 0) = u_0 \qquad \forall (t, x) \in [0, T] \times D, \tag{2.1}$$

where $u_0 \in \mathcal{Y} \subset L^2(D)$ is the initial condition and $\mathcal{L}_a \colon \mathcal{H} \to L^2([0, T] \times D)$ is a differential operator that can depend on a parameter (function) $a \in \mathcal{Z} \subset L^2(D)$. In our notation, we will often suppress the dependence of $\mathcal{L} := \mathcal{L}_a$ on a for simplicity. Depending on the context, one might want to recover one of the following mathematical objects: for fixed a and u_0, one might want to approximate $u(T, \cdot)$ or $u(\cdot, \cdot)$ with a neural network; a more challenging task would be to learn the *solution operator* $\mathcal{G} : \mathcal{X} \to L^2(\Omega) : v \mapsto u$, where $v \in \{u_0, a\}$, $\mathcal{X} \in \{\mathcal{Y}, \mathcal{Z}\}$ and $\Omega = D$ or

$\Omega = [0,T] \times D$. We will use this notation consistently throughout the paper, see **SM** A.1 for an overview.

2.2 Approximating PDEs with neural networks

Neural networks A (feedforward) neural network $u_\theta : \mathbb{R}^{d_0} \to \mathbb{R}^{d_L}$ is defined as a concatenation of affine maps $\mathcal{A}_l : \mathbb{R}^{d_{l-1}} \to \mathbb{R}^{d_l} : z \mapsto W_l z + b_l$ and an activation function $\sigma : \mathbb{R} \to \mathbb{R}$ that is applied component-wise, resulting in,

$$u_\theta(y) = \mathcal{A}_L \circ \sigma \circ \mathcal{A}_{L-1} \ldots\ldots\ldots \circ \sigma \circ \mathcal{A}_2 \circ \sigma \circ \mathcal{A}_1(y). \tag{2.2}$$

The weights and biases of the affine maps $\theta = \{W_l, b_l\}_{1 \le l \le L}$ are the trainable parameters. We will quantify the size of a neural network by its $\mathrm{depth}(u_\theta) := L$ and its $\mathrm{width}(u_\theta) := \max_l d_l$. In order to obtain a neural network that approximates the solution u of PDE (2.1) at time $t = T$, one chooses the parameters of $u_\theta : D \to \mathbb{R}$ such that a discretization (quadrature) of $\mathcal{J}(\theta) = \|u(T) - u_\theta\|_{L^2(D)}$ is minimized. The training data is acquired from either measurements or potentially expensive simulations.

PINNs Physics-informed neural networks (PINNs) are neural networks that are trained with a different, residual-based loss function. As the PDE solution u satisfies $\mathcal{L}(u) = 0$, the goal of physics-informed learning is to find a neural network $u_\theta : [0,T] \times D \to \mathbb{R}$ for which the PDE residual is approximately zero, $\mathcal{L}(u_\theta) \approx 0$. To ensure uniqueness, one also needs to require that the initial condition is satisfied i.e., $u_\theta(0,x) \approx u_0(x)$, and similarly for boundary conditions. In practice one minimizes a quadrature approximation of $\mathcal{J}(\theta) = \|\mathcal{L}(u_\theta)\|^2_{L^2([0,T]\times D)} + \|u_\theta(0,\cdot) - u_0\|^2_{L^2(D)}$, where additional terms can be added to (approximately) impose boundary conditions and augment the loss function using data. A desirable property of PINNs is that only very little or even no training data is needed to construct the loss function.

Operator learning In order to approximate operators, one needs to allow the input and output of the learning architecture to be infinite-dimensional. A possible approach is to use *deep operator networks* (DeepONets), as proposed in [9, 49]. Given m, fixed sensor locations $\{x_j\}_{j=1}^m \subset D$ and the corresponding *sensor values* $\{v(x_j)\}_{j=1}^m$ as input, a DeepONet can be formulated in terms of two (deep) neural networks: a *branch net* $\boldsymbol{\beta} : \mathbb{R}^m \to \mathbb{R}^p$ and a *trunk net* $\boldsymbol{\tau} : D \to \mathbb{R}^{p+1}$. The branch and trunk nets are then combined to approximate the underlying nonlinear operator as the following *DeepONet* $\mathcal{G}_\theta : \mathcal{X} \to L^2(D)$, with $\mathcal{G}_\theta(v)(y) = \tau_0(y) + \sum_{k=1}^p \beta_k(v)\tau_k(y)$. A second approach is that of *neural operators*, which generalize hidden layers by including a non-local integral operator [45], of which particularly *Fourier neural operators* (FNOs) [44] are already well-established. The practical implementation (i.e. discretization) of an FNO maps from and to the space of trigonometric polynomials of degree at most $N \in \mathbb{N}$, denoted by L^2_N, and can be identified with a finite-dimensional mapping that is a composition of affine maps and nonlinear layers of the form $\mathfrak{L}_l(z)_j = \sigma(W_l v_j + b_{l,j}\mathcal{F}_N^{-1}(P_l(k) \cdot \mathcal{F}_N(z)(k)_j))$, where the $P_l(k)$ are coefficients that define a non-local convolution operator via the discrete Fourier transform $\mathcal{F}_N$, see [38].

Physics-informed operator learning Both DeepONets and FNOs are trained by choosing a suitable probability measure μ on $\mathcal{X}$ and minimizing a quadrature approximation of $\mathcal{J}(\theta) = \|\mathcal{G}_\theta(v) - \mathcal{G}(v)\|_{L^2_{\mu\times dx}(\mathcal{X}\times\Omega)}$. Generating training sets might require many calls to an expensive PDE solver, leading to an enormous computational cost. In order to reduce or even fully eliminate the need for training data, *physics-informed operator learning* has been proposed in [74] for DeepONets and in [47] for FNOs. Similar to PINNs, the training procedure aims to minimize a quadrature approximation of $\mathcal{J}(\theta) = \|\mathcal{L}(\mathcal{G}_\theta)\|_{L^2_{\mu\times dx}(\mathcal{X}\times\Omega)}$.

3 General results

We propose a framework to obtain bounds on the approximation error for the various neural network architectures introduced in Section 2.2. Figure 1 visualizes how different types of error estimates can be obtained from one another. Every box shows the name of the network architecture, the form of the relevant loss and the theorem which proves the corresponding estimate for the approximation error. Every arrow in the flowchart represents a proof technique that allows one to transfer an error estimate from one type of method to another (see caption of Figure 1 for an overview of those techniques).

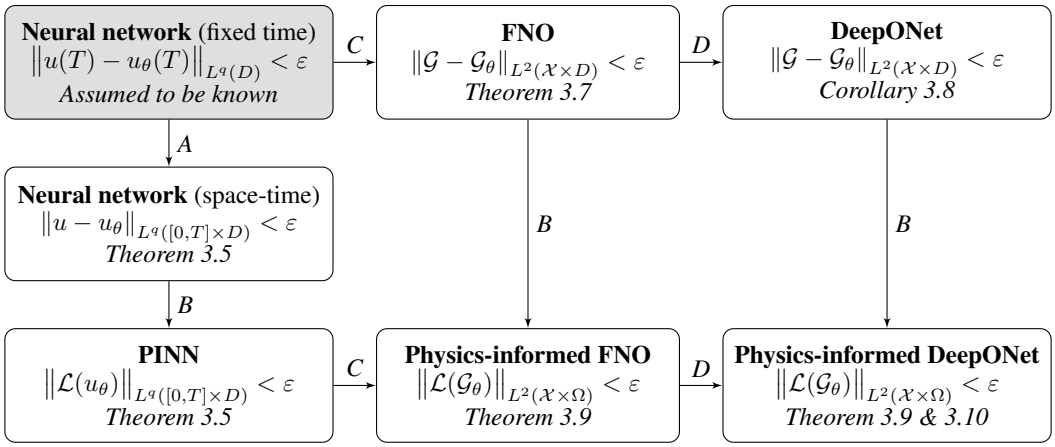

Figure 1: Flowchart of the structure of the results in this paper, with $q \in \{2, \infty\}$. The letters reflect the techniques used in the proofs: *A* uses Taylor approximations (Section 3.1), *B* is based on finite difference approximations (Section 3.1), *C* uses trigonometric polynomial interpolation (Section 3.2) and *D* uses the connection between FNOs and DeepONets (Section 3.2).

We give particular attention to the case where it is known that a neural network can efficiently approximate the solution to a time-dependent PDE at a fixed time. Such neural networks are usually obtained by emulating a classical numerical method. Examples include finite difference schemes, finite volume schemes, finite element methods, iterative methods and Monte Carlo methods, e.g. [36, 63, 10, 57]. More precisely, for $\varepsilon > 0$, we assume to have access to an operator $\mathcal{U}^\varepsilon : \mathcal{X} \times [0, T] \to \mathcal{H}$ that for any $t \in [0, T]$ maps any initial condition/parameter function $v \in \mathcal{X}$ to a neural network $\mathcal{U}^\varepsilon(v, t)$ that approximates the PDE solution $\mathcal{G}(v)(\cdot, t) = u(\cdot, t) \in L^q(D), q \in \{2, \infty\}$, at time t, as specified below. Moreover, we will assume that we know how its size depends on the accuracy ε. Explicit examples of the operator $\mathcal{U}^\varepsilon$ will be given in Section 4 and **SM** C.

Assumption 3.1. *Let $q \in \{2, \infty\}$. For any $B, \varepsilon > 0$, $\ell \in \mathbb{N}$, $t \in [0, T]$ and any $v \in \mathcal{X}$ with $\|v\|_{C^\ell} \leq B$ there exist a neural network $\mathcal{U}^\varepsilon(v, t) : D \to \mathbb{R}$ and a constant $C^B_{\varepsilon,\ell} > 0$ s.t.*

$$\left\|\mathcal{U}^\varepsilon(v, t) - \mathcal{G}(v)(\cdot, t)\right\|_{L^q(D)} \leq \varepsilon \quad \text{and} \quad \max_{t\in[0,T]} \left\|\mathcal{U}^\varepsilon(v, t)\right\|_{W^{\ell,q}(D)} \leq C^B_{\varepsilon,\ell}. \tag{3.1}$$

Remark 3.2. *For vanilla neural networks and PINNs one can set $\mathcal{X} := \{v\}$, $\mathcal{G}(v) := u$ and $v := u_0$ or $v := a$ in Assumption 3.1 above and Assumption 3.4 below.*

Under this assumption, we prove the existence of space-time neural networks and PINNs that efficiently approximate the PDE solution (Section 3.1), as well as FNOs and DeepONets (Section 3.2) and physics-informed FNOs and DeepONets (Section 3.3). Finally, we also prove a general result on the generalization error (Section 3.4).

3.1 Estimates for (physics-informed) neural networks

We will construct a space-time neural network u_θ for which both $\|u_\theta - u\|_{L^q([0,T]\times D)}$ and the PINN loss $\|\mathcal{L}(u_\theta)\|_{L^q([0,T]\times D)}$ are small. To accurately approximate the time derivatives of u we emulate Taylor expansions, whereas for the spatial derivatives we employ finite difference (FD) operators in our proofs. Depending on whether forward, backward or central differences are used, a FD operator might not be defined on the whole domain D, e.g. for $f \in C([0, 1])$ the (forward) operator $\Delta^+_h[f] := f(x + h) - f(x)$ is not well-defined for $x \in (1 - h, 1]$. This can be solved by resorting to piecewise-defined FD operators, e.g. a forward operator on $[0, 0.5]$ and a backward operator on $(0.5, 1]$. In a general domain Ω one can find a well-defined piecewise FD operator if Ω satisfies the following assumption, which is satisfied by many domains (e.g. rectangular, smooth).

Assumption 3.3. *There exists a finite partition $\mathcal{P}$ of Ω such that for all $P \in \mathcal{P}$ there exists $\varepsilon_P > 0$ and $v_P \in B^1_\infty = \{x \in \mathbb{R}^{\dim(\Omega)} : \|x\|_\infty \leq 1\}$ such that for all $x \in P$ it holds that $x + \varepsilon_P(v_P + B^1_\infty) \subset \Omega$.*

Additionally, we need to assume that the PINN error can be bounded in terms of the errors related to all relevant partial derivatives, denoted by $D^{(k,\boldsymbol{\alpha})} := D^k_t D^{\boldsymbol{\alpha}}_x := \partial^k_t \partial^{\alpha_1}_{x_1} \dots \partial^{\alpha_d}_{x_d}$, for $(k, \boldsymbol{\alpha}) \in \mathbb{N}^{d+1}_0$.

This assumption is valid for many classical solutions of PDEs. A few worked out examples can be found in **SM** D.5 (gravity pendulum) and **SM** D.6 (Darcy flow).

Assumption 3.4. *Let $k, \ell \in \mathbb{N}$, $q \in \{2, \infty\}$, $C > 0$ be independent from d. For all $v \in \mathcal{X}$ it holds,*

$$\left\|\mathcal{L}(\mathcal{G}_\theta(v))\right\|_{L^q([0,T]\times D)} \leq C \cdot \mathrm{poly}(d) \cdot \sum_{\substack{(k',\boldsymbol{\alpha})\in\mathbb{N}_0^{d+1} \\ k'\leq k, \|\boldsymbol{\alpha}\|_1 \leq \ell}} \left\|D^{(k',\boldsymbol{\alpha})}(\mathcal{G}-\mathcal{G}_\theta)\right\|_{L^q([0,T]\times D)}. \tag{3.2}$$

In this setting, we prove the following approximation result for space-time networks and PINNs.

Theorem 3.5. *Let $s, r \in \mathbb{N}$, let $u \in C^{(s,r)}([0,T] \times D)$ be the solution of the PDE* (2.1) *and let Assumption 3.1 be satisfied. There exists a constant $C(s,r) > 0$ such that for every $M \in \mathbb{N}$ and $\varepsilon, h > 0$ there exists a tanh neural network $u_\theta : [0,T] \times D \to \mathbb{R}$ for which it holds that,*

$$\|u_\theta - u\|_{L^q([0,T]\times D)} \leq C(\|u\|_{C^{(s,0)}} M^{-s} + \varepsilon). \tag{3.3}$$

and if additionally Assumption 3.3 and Assumption 3.4 hold then,

$$\begin{aligned} &\left\|\mathcal{L}(u_\theta)\right\|_{L^2([0,T]\times D)} + \|u_\theta - u\|_{L^2(\partial([0,T]\times D))} \\ &\quad \leq C \cdot \mathrm{poly}(d) \cdot \ln^k(M)(\|u\|_{C^{(s,\ell)}} M^{k-s} + M^{2k}(\varepsilon h^{-\ell} + C^B_{\varepsilon,\ell} h^{r-\ell})). \end{aligned} \tag{3.4}$$

Moreover, $\mathrm{depth}(u_\theta) \leq C \cdot \mathrm{depth}(\mathcal{U}^\varepsilon)$ *and* $\mathrm{width}(u_\theta) \leq CM \cdot \mathrm{width}(\mathcal{U}^\varepsilon)$.

Proof. We only provide a sketch of the full proof (**SM** B.2). The main idea is to divide $[0,T]$ into M uniform subintervals and construct a neural network that approximates a Taylor approximation in time of u in each subinterval. In the obtained formula, we approximate the monomials and multiplications by neural networks (**SM** A.7) and approximate the derivatives of u by finite differences and use (A.2) of **SM** A.2 to find an error estimate in $C^k([0,T], L^q(D))$-norm. We use again finite difference operators to prove that spatial derivatives of u are accurately approximated as well. The neural network will also approximately satisfy the initial/boundary conditions as $\|u_\theta - u\|_{L^2(\partial([0,T]\times D))} \lesssim C\mathrm{poly}(d)\|u_\theta - u\|_{H^1([0,T]\times D)}$, which follows from a Sobolev trace inequality. □

We note that the bounds (3.3) and (3.4) together imply that there exists a neural network for which the total error as well as the PINN loss can be made as small as possible, providing a solid theoretical foundation to PINNs for approximating the PDE (2.1).

3.2 Estimates for operator learning

In this section, we use Assumption 3.1 to prove estimates for DeepONets and FNOs. First, we prove a generic error estimate for FNOs. Using the known connection between FNOs and DeepONets (**SM** Lemma B.6) this result can then easily be applied to DeepONets (Corollary 3.8). In order to prove these error estimates, we need to assume that the operator $\mathcal{U}^\varepsilon$ from Assumption 3.1 is stable with respect to its input function, as specified in Assumption 3.6 below. Moreover, we will take the d-dimensional torus as domain $D = \mathbb{T}^d = [0, 2\pi)^d$ and assume periodic boundary conditions for simplicity in what follows. This is not a restriction, as for every Lipschitz subset of $\mathbb{T}^d$ there exists a (linear and continuous) $\mathbb{T}^d$-periodic extension operator of which also the derivatives are $\mathbb{T}^d$-periodic [38, Lemma 41].

Assumption 3.6. *Assumption 3.1 is satisfied and let $p \in \{2, \infty\}$. For every $\varepsilon > 0$ there exists a constant $C^\varepsilon_{\mathrm{stab}} > 0$ such that for all $v, v' \in \mathcal{X}$ it holds that,*

$$\left\|\mathcal{U}^\varepsilon(v,T) - \mathcal{U}^\varepsilon(v',T)\right\|_{L^2} \leq C^\varepsilon_{\mathrm{stab}} \left\|v - v'\right\|_{L^p}. \tag{3.5}$$

In this setting, we prove a generic approximation result for FNOs.

Theorem 3.7. *Let $r \in \mathbb{N}$, $T > 0$, let $\mathcal{G} : C^r(\mathbb{T}^d) \to C^r(\mathbb{T}^d)$ be an operator that maps a function u_0 to the solution $u(\cdot, T)$ of the PDE* (2.1) *with initial condition u_0, let Assumption 3.6 be satisfied and let $p^* \in \{2, \infty\} \setminus \{p\}$. Then there exists a constant $C > 0$ such that for every $\varepsilon > 0$, $N \in \mathbb{N}$ there is an FNO $\mathcal{G}_\theta : L^2_N(\mathbb{T}^d) \to L^2_N(\mathbb{T}^d)$ of depth $\mathcal{O}(\mathrm{depth}(\mathcal{U}^\varepsilon))$ and width $\mathcal{O}(N^d \mathrm{width}(\mathcal{U}^\varepsilon))$ with accuracy,*

$$\|\mathcal{G} - \mathcal{G}_\theta\|_{L^2} \leq C(\varepsilon + C^\varepsilon_{\mathrm{stab}} B N^{-r+d/p^*} + C^{CB}_{\varepsilon,r} N^{-r}). \tag{3.6}$$

Proof. We give a sketch of the proof, details can be found in **SM** B.3. Given function values of v on a uniform grid with grid size $1/N$, we use trigonometric polynomial interpolation (**SM** A.6) to reconstruct v and use this together with Assumption 3.1 to construct a neural network. The resulting approximation is then projected onto the space L^2_N, of trigonometric polynomials of degree at most $N \in \mathbb{N}$, again through trigonometric polynomial interpolation. □

A recent result, [38, Theorem 36] (**SM** Lemma B.6), shows that any error bound for FNOs also implies an error bound for DeepONets, by choosing the trunk nets as neural network approximations of the Fourier basis. We apply this result with $\varepsilon \sim \mathrm{poly}(1/N)$ to Theorem 3.7 to obtain the following generic error bound for DeepONets.

Corollary 3.8. *Assume the setting of Theorem 3.7. Then for every $\varepsilon > 0$, $N \in \mathbb{N}$ and every corresponding FNO $\mathcal{G}_\theta$ from Theorem 3.7 there exists a DeepONet $\mathcal{G}^*_\theta : \mathcal{X} \to L^2(D)$ with* $\mathrm{width}(\boldsymbol{\beta}) = \mathcal{O}(N^d)$, $\mathrm{depth}(\boldsymbol{\beta}) = \mathcal{O}(\mathrm{depth}(\mathcal{G}_\theta))$, $\mathrm{width}(\boldsymbol{\tau}) = \mathcal{O}(N^{d+1})$ *and* $\mathrm{depth}(\boldsymbol{\tau}) \leq 3$ *that satisfies* (3.6).

3.3 Estimates for physics-informed operator learning

Using the techniques from previous sections, we now present the very first theoretical result for physics-informed operator learning. We demonstrate that if an error estimate for a DeepONet/FNO and the growth of its derivatives are known (see **SM** D.1 on how to obtain these), then one can prove an error estimate for the corresponding physics-informed DeepONet/FNO. For simplicity, the following result focuses only on operators mapping to $C^r(D)$ but the generalization to e.g. $C^r([0,T] \times D)$ is immediate by considering $D' := [0,T] \times D$.

Theorem 3.9. *Consider an operator $\mathcal{G} : \mathcal{X} \to C^r(D)$, $r \in \mathbb{N}$, that satisfies Assumption 3.3 and Assumption 3.4 with $\ell \in N$. Let $\lambda^* \in (0,\infty]$, let $\lambda, C(\lambda) > 0$ with $\lambda \leq \lambda^*$ and let $\sigma : \mathbb{N} \to \mathbb{R}$ be a function such that for all $p \in \mathbb{N}$ there is a DeepONet/FNO $\mathcal{G}_\theta$ such that*

$$\left\|\mathcal{G}(v) - \mathcal{G}_\theta(v)\right\|_{L^2(D)} \leq Cp^{-\lambda} \qquad \text{and} \qquad \left|\mathcal{G}_\theta(v)\right|_{C^r(D)} \leq Cp^{\sigma(r)} \quad \forall r \in \mathbb{N}, v \in \mathcal{X}. \tag{3.7}$$

Then for all $\beta \in \mathbb{R}$ with $0 < \beta \leq \frac{(r-\ell)\lambda^* - \ell\sigma(r)}{r}$ there exists a constant $C^* > 0$ such that for all $v \in \mathcal{X}$ and $p \in \mathbb{N}$ it holds that

$$\left\|\mathcal{L}(\mathcal{G}_\theta(v))\right\|_{L^2(D)} \leq C^* p^{-\beta}. \tag{3.8}$$

Proof. For suitable $D^{\boldsymbol{\alpha}}$, use **SM** Lemma B.1 with $q = 2$, $f_1 = \mathcal{G}(v)$ and $f_2 = \mathcal{G}_\theta(v)$ together with (3.7) to find

$$\left\|D^{\boldsymbol{\alpha}}(\mathcal{G}(v) - \mathcal{G}_\theta(v))\right\|_{L^2(D)} \leq C(r,\lambda)(p^{-\lambda}h^{-\ell} + p^{\sigma(r)}h^{r-\ell}). \tag{3.9}$$

Let $\beta \in \mathbb{R}$ with $0 < \beta \leq \frac{(r-\ell)\lambda^* - \ell\sigma(r)}{r}$. We carefully balance terms by setting $h = p^{-\frac{\sigma(r)+\beta}{r-\ell}}$ and $\lambda = \frac{\ell}{r-\ell}\sigma(r) + \frac{r}{r-\ell}\beta$ to find (3.8). Conclude using Assumption 3.4. □

Finally, we use Theorem 3.5 to present an alternative error estimate for a physics-informed DeepONet in the case that Assumption 3.1 is satisfied. As this assumption is different from assuming access to an error bound for the corresponding DeepONet, it is interesting to use the techniques from the previous sections rather than directly apply Theorem 3.9. The proof of the following theorem can be found in **SM** B.4.

Theorem 3.10. *Let $s, r \in \mathbb{N}$, $T > 0$, let $\mathcal{G} : C^r(\mathbb{T}^d) \to C^{(s,r)}([0,T] \times \mathbb{T}^d)$ be an operator that maps a function u_0 to the solution u of the PDE* (2.1) *with initial condition u_0, let Assumption 3.1 and Assumption 3.6 be satisfied and let $p^* \in \{2,\infty\} \setminus \{p\}$. There exists a constant $C > 0$ such that for every $Z, N, M \in \mathbb{N}$, $\varepsilon, \rho > 0$ there is an DeepONet $\mathcal{G}_\theta : C^r(\mathbb{T}^d) \to L^2([0,T] \times \mathbb{T}^d)$ with Z^d sensors with accuracy,*

$$\left\|\mathcal{G}(v) - \mathcal{G}_\theta(v)\right\|_{L^2([0,T]\times\mathbb{T}^d)} \leq CM^\rho(\|u\|_{C^{(s,0)}}M^{-s} + M^{s-1}(\varepsilon + C^\varepsilon_{\mathrm{stab}}Z^{-r+d/p^*} + C^{CB}_{\varepsilon,r}N^{-r})) \tag{3.10}$$

and if additionally Assumption 3.3 and Assumption 3.4 hold then,

$$\left\|\mathcal{L}(\mathcal{G}_\theta(v))\right\|_{L^2([0,T]\times\mathbb{T}^d)} \leq CM^{k+\rho}(\|u\|_{C^{(s,\ell)}}M^{-s} + M^{s-1}N^\ell(\varepsilon + C^\varepsilon_{\mathrm{stab}}Z^{-r+d/p^*} + C^{CB}_{\varepsilon,r}N^{-r})), \tag{3.11}$$

for all v. Moreover, it holds that, $\mathrm{depth}(\boldsymbol{\beta}) = \mathrm{depth}(\mathcal{U}^\varepsilon)$, $\mathrm{width}(\boldsymbol{\beta}) = \mathcal{O}(M(Z^d + N^d\mathrm{width}(\mathcal{U}^\varepsilon)))$, $\mathrm{depth}(\boldsymbol{\tau}) = 3$ *and* $\mathrm{width}(\boldsymbol{\tau}) = \mathcal{O}(MN^d(N + \ln(N)))$.

3.4 A posteriori bound on the generalization error

Although the main focus of this paper is on the approximation error for different neural network architectures, we now demonstrate that it is possible to provide similar bounds for other sources of error, such as the generalization error. We therefore prove a general *a posteriori* upper bound on the generalization error of the all the considered neural network architectures. Consider $f : \mathcal{D} \to \mathbb{R}$ (an operator or function) and the neural network architecture $f_\theta : \mathcal{D} \to \mathbb{R}$, $\theta \in \Theta$, which includes all architectures of Section 2.2: neural networks ($\mathcal{D} = \Omega$, $f = u$ and $f_\theta = u_\theta$), PINNs ($\mathcal{D} = \Omega$, $f = 0$ and $f_\theta = \mathcal{L}(u_\theta)$), operator learning ($\mathcal{D} = \Omega \times \mathcal{X}$, $f = \mathcal{G}$ and $f_\theta = \mathcal{G}_\theta$) and physics-informed operator learning ($\mathcal{D} = \Omega \times \mathcal{X}$, $f = 0$ and $f_\theta = \mathcal{L}(\mathcal{G}_\theta)$). Given a training set $\mathcal{S} = \{X_1, \dots X_n\}$, where $\{X_i\}_{i=1}^n$ are iid random variables on $\mathcal{D}$ (according to a measure μ), the training error $\mathcal{E}_T$ and generalization error $\mathcal{E}_G$ are,

$$\mathcal{E}_T(\theta, \mathcal{S})^2 = \frac{1}{n} \sum_{i=1}^{n} \left| f(z_i) - f_\theta(z_i) \right|^2, \qquad \mathcal{E}_G(\theta)^2 = \int_{\mathcal{D}} \left| f_\theta(z) - f(z) \right|^2 d\mu(z), \tag{3.12}$$

where μ is a probability measure on $\mathcal{D}$. The following theorem provides a computable a posteriori error bound on the expectation of the generalization error for a general class of approximators. We refer to e.g. [6, 16] for bounds on d_Θ, c and $\mathfrak{L}$.

Theorem 3.11. *For $R > 0$ and $d_\Theta \in \mathbb{N}$, let $\Theta = [-R, R]^{d_\Theta}$ be the set of trainable parameters, and for every training set $\mathcal{S}$, let $\theta^*(\mathcal{S}) \in \Theta$ be an (approximate) minimizer of $\theta \mapsto \mathcal{E}_T(\theta, \mathcal{S})^2$, assume that $\theta \mapsto \mathcal{E}_G(\theta, \mathcal{S})^2$ and $\theta \mapsto \mathcal{E}_T(\theta)^2$ are bounded by $c > 0$ and Lipschitz continuous with Lipschitz constant $\mathfrak{L} > 0$. If $n \geq 2c^2 e^8 / (2R\mathfrak{L})^{d_\Theta/2}$ then it holds that*

$$\mathbb{E}\left[\mathcal{E}_G(\theta^*(\mathcal{S}))^2\right] \leq \mathbb{E}\left[\mathcal{E}_T(\theta^*(\mathcal{S}), \mathcal{S})^2\right] + \sqrt{\frac{2c^2(d_\Theta + 1)}{n} \ln\big(R\mathfrak{L}\sqrt{n}\big)}. \tag{3.13}$$

Proof. The proof (**SM** B.5) combines standard techniques, based on covering numbers and Hoeffding's inequality, with an error composition from [16]. □

For any type of neural network architecture of depth L, width W and weights bounded by R, one finds that $d_\Theta \sim LW(W + d)$. For tanh neural networks and operator learning architectures, one has that $\ln(\mathfrak{L}) \sim L \ln(dRW)$, whereas for physics-informed neural networks and DeepONets one finds that $\ln(\mathfrak{L}) \sim (k + \ell) L \ln(dRW)$ with k and ℓ as in Assumption 3.4 [43, 16]. Taking this into account, one also finds that the imposed lower bound on n is not very restrictive. Moreover, the RHS of (3.13) depends at most polynomially on L, W, R, d, k, ℓ and c. For physics-informed architectures, however, upper bounds on c often depend exponentially on L [16, 14].

Remark 3.12. *As Theorem 3.11 is an a posteriori error estimate, one can use the network sizes of the trained networks for L, W and R. The sizes stemming from the approximation error estimates of the previous sections can be disregarded for this result. Moreover, instead of considering the expected values of $\mathcal{E}_G$ and $\mathcal{E}_T$ in* (3.13)*, one can also prove that such an inequality holds with a certain probability (see **SM** B.5).*

4 Applications

We demonstrate the power and generality of the framework proposed in Section 3 by applying the presented theory to the following case studies. First, we demonstrate how these generic bounds can be used to overcome the curse of dimensionality (CoD) for linear Kolmogorov PDEs and nonlinear parabolic PDEs (Section 4.1). These are the first available results that overcome the CoD for nonlinear parabolic PDEs for PINNs and (physics-informed) operator learning. Next, we apply the results of Section 3.3 to both linear and nonlinear operators and provide bounds on the approximation error for physics-informed operator learning.

4.1 Overcoming the curse of dimensionality

For high-dimensional PDEs, it is not possible to obtain efficient approximation results using standard neural network approximation theory [77, 15] as they will lead to convergence rates that suffer from the CoD, meaning that the neural network size scales exponentially in the input dimension.

In literature, one has shown for some PDEs that their solution at a fixed time can be approximated to accuracy $\varepsilon > 0$ with a network that has size $\mathcal{O}(\text{poly}(d)\varepsilon^{-\beta})$, with $\beta > 0$ independent of d, and therefore *overcomes the CoD*.

Linear Kolmogorov PDEs We consider linear time-dependent PDEs of the following form.

Setting 4.1. *Let $s, r \in \mathbb{N}$, $u_0 \in C_0^2(\mathbb{R}^d)$ and let $u \in C^{(s,r)}([0,T] \times \mathbb{R}^d)$ be the solution of*

$$\mathcal{L}(u)(x,t) = \partial_t u(x,t) - \frac{1}{2}\text{Tr}(\sigma(x)\sigma(x)^T \Delta_x[u](x,t)) - \mu(x)^T \nabla_x[u](x,t) = 0, \quad u(0,x) = u_0(x) \tag{4.1}$$

for all $(x,t) \in D \times [0,T]$, where $\sigma : \mathbb{R}^d \to \mathbb{R}^{d\times d}$ and $\mu : \mathbb{R}^d \to \mathbb{R}^d$ are affine functions and for which $\|u\|_{C^{(s,2)}}$ grows at most polynomially in d. For every $\varepsilon > 0$, there is a neural network $\widehat{u}_0$ of width $\mathcal{O}(\text{poly}(d)\varepsilon^{-\beta})$ such that $\|u_0 - \widehat{u}_0\|_{L^\infty(\mathbb{R}^d)} < \varepsilon$.

Prototypical examples of such *linear Kolmogorov PDEs* include the heat equation and the Black-Scholes equation. In [23, 7, 36] the authors construct a neural network that approximates $u(T)$ and overcomes the CoD by emulating Monte-Carlo methods based on the Feynman-Kac formula. In [16] one has proven that PINNs overcome the CoD as well, in the sense that the network size grows as $\mathcal{O}(\text{poly}(d\rho_d)\varepsilon^{-\beta})$, with ρ_d as defined in **SM** (C.10). For a subclass of Kolmogorov PDEs it is known that $\rho_d = \text{poly}(d)$, such that the CoD is fully overcome.

We demonstrate that the generic bounds of Section 3 (Theorem 3.5) can be used to provide a much shorter proof for this result. **SM** Lemma C.6 verifies that Assumption 3.1 is indeed satisfied. The full proof can be found in **SM** C.2.

Theorem 4.2. *Assume that Setting 4.1 holds. For every $\sigma, \varepsilon > 0$ and $d \in \mathbb{N}$, there is a tanh neural network u_θ of depth $\mathcal{O}(\text{depth}(\widehat{u}_0))$ and width $\mathcal{O}(\text{poly}(d\rho_d)\varepsilon^{-(2+\beta)\frac{r+\sigma}{r-2}\frac{s+1}{s-1}-\frac{1+\sigma}{s-1}})$ such that,*

$$\left\|\mathcal{L}(u_\theta)\right\|_{L^2([0,T]\times[0,1]^d)} + \|u_\theta - u\|_{L^2(\partial([0,T]\times[0,1]^d))} \leq \varepsilon. \tag{4.2}$$

Nonlinear parabolic PDEs Next, we consider nonlinear parabolic PDEs as in Section 4.3, which typically arise in the context of nonlinear diffusion-reaction equations that describe the change in space and time of some quantities, such as in the well-known *Allen-Cahn equation* [1].

Setting 4.3. *Let $s, r \in \mathbb{N}$ and for $u_0 \in \mathcal{X} \subset C^r(\mathbb{T}^d)$ let $u \in C^{(s,r)}([0,T] \times \mathbb{T}^d)$ be the solution of*

$$\mathcal{L}(u)(x,t) = \partial_t u(t,x) - \Delta_x u(t,x) - F(u(t,x)) = 0, \qquad u(0,x) = u_0(x), \tag{4.3}$$

for all $(t,x) \in [0,T] \times D$, with period boundary conditions, where $F : \mathbb{R} \to \mathbb{R}$ is a polynomial and for which $\|u\|_{C^{(s,2)}}$ grows at most polynomially in d. For every $\varepsilon > 0$, there is a neural network $\widehat{u}_0$ of width $\mathcal{O}(\text{poly}(d)\varepsilon^{-\beta})$ such that $\|u_0 - \widehat{u}_0\|_{L^\infty(\mathbb{T}^d)} < \varepsilon$. Let μ, resp. μ^*, be the normalized Lebesgue measure on $[0,T] \times \mathbb{T}^d$, resp. $\partial([0,T] \times \mathbb{T}^d)$.

In [32] the authors have proven that ReLU neural networks overcome the CoD in the approximation of $u(T)$. We have reproven this result in **SM** Lemma C.14 for tanh neural networks to show that Assumption 3.1 is satisfied. Using Theorem 3.5 we can now prove that PINNs overcome the CoD for nonlinear parabolic PDEs. The proof is analogous to that of Theorem 4.2.

Theorem 4.4. *Assume Setting 4.3. For every $\sigma, \varepsilon > 0$ and $d \in \mathbb{N}$ there is a tanh neural network u_θ of depth $\mathcal{O}(\text{depth}(\widehat{u}_0) + \text{poly}(d)\ln(1/\varepsilon))$ and width $\mathcal{O}(\text{poly}(d)\varepsilon^{-(2+\beta)\frac{r+\sigma}{r-2}\frac{s+1}{s-1}-\frac{1+\sigma}{s-1}})$ such that,*

$$\left\|\mathcal{L}(u_\theta)\right\|_{L^2([0,T]\times\mathbb{T}^d,\mu)} + \|u - u_\theta\|_{L^2(\partial([0,T]\times\mathbb{T}^d,\mu^*))} \leq \varepsilon. \tag{4.4}$$

Similarly, one can use the results from Section 3.2 to obtain estimates for (physics-informed) DeepONets for nonlinear parabolic PDEs (4.3) such as the Allen-Cahn equation. In particular, a dimension-independent convergence rate can be obtained if the solution is smooth enough, which improves upon the result of [43], which incurred the CoD. For simplicity, we present results for $C^{(2,r)}$ functions, rather than $C^{(s,r)}$ functions, as we found that assuming more regularity did not necessarily further improve the convergence rate. The proof is given in **SM** B.4.

Theorem 4.5. *Assume Setting 4.3 and let $\mathcal{G} : \mathcal{X} \to C^r(\mathbb{T}^d) : u_0 \mapsto u(T)$ and $\mathcal{G}^* : \mathcal{X} \to C^{(2,r)}([0,T] \times \mathbb{T}^d) : u_0 \mapsto u$. For every $\sigma, \varepsilon > 0$, there exists a DeepONets $\mathcal{G}_\theta$ and $\mathcal{G}_\theta^*$ such that*

$$\|\mathcal{G} - \mathcal{G}_\theta\|_{L^2(\mathbb{T}^d\times\mathcal{X})} \leq \varepsilon, \qquad \left\|\mathcal{L}(\mathcal{G}_\theta^*)\right\|_{L^2([0,T]\times\mathbb{T}^d\times\mathcal{X})} \leq \varepsilon. \tag{4.5}$$

Moreover, for $\mathcal{G}_\theta$ we have $\mathcal{O}(\varepsilon^{-\frac{d+\sigma}{r}})$ sensors and,

$$\begin{aligned} \mathrm{width}(\boldsymbol{\beta}) &= \mathcal{O}(\varepsilon^{-\frac{(d+\sigma)(2+\beta)}{r}}), \quad \mathrm{depth}(\boldsymbol{\beta}) = \mathcal{O}(\ln(1/\varepsilon)), \\ \mathrm{width}(\boldsymbol{\tau}) &= \mathcal{O}(\varepsilon^{-\frac{d+1+\sigma}{r}}), \qquad \mathrm{depth}(\boldsymbol{\tau}) = 3, \end{aligned} \tag{4.6}$$

whereas for $\mathcal{G}_\theta^*$ we have $\mathcal{O}(\varepsilon^{-\frac{(3+\sigma)d}{r-2}})$ sensors and,

$$\begin{aligned} \mathrm{width}(\boldsymbol{\beta}) &= \mathcal{O}(\varepsilon^{-1-\frac{(3+\sigma)(d+r(2+\beta))}{r-2}}), \quad \mathrm{depth}(\boldsymbol{\beta}) = \mathcal{O}(\ln(1/\varepsilon)), \\ \mathrm{width}(\boldsymbol{\tau}) &= \mathcal{O}(\varepsilon^{-1-\frac{(3+\sigma)(d+1)}{r-2}}), \qquad \mathrm{depth}(\boldsymbol{\tau}) = 3. \end{aligned} \tag{4.7}$$

4.2 Error bounds for physics-informed operator learning

We demonstrate how Theorem 3.9 can be used to generalize available error estimates for DeepONets and FNOs, e.g. [43, 38] and **SM** D.1, to estimates for their physics-informed counterparts.

Linear operators In the simplest case, the operator $\mathcal{G}$ of interest is linear. In [43, Theorem D.2], a general error bound for ReLU DeepONets for linear operators has been established, which still holds for tanh DeepONets. Using Theorem 3.9 it is then straightforward to prove convergence rates for physics-informed DeepONets for solution operators of linear PDEs (2.1).

Consider an operator $\mathcal{G} : \mathcal{X} \to L^2(\mathbb{T}^d) : v \mapsto u$ as in Section 2.1, where v is the parameter/initial condition and u the solution of the PDE (2.1). Following [43], we fix the measure μ on $L^2(\mathbb{T}^d)$ as a Gaussian random field, such that v allows the Karhunen-Loève expansion $v = \sum_{k\in\mathbb{Z}^d} \alpha_k X_k \mathbf{e}_k$, where $|\alpha_k| \leq \exp(-\ell|k|)$ with $\ell > 0$, the $X_k \sim \mathcal{N}(0,1)$ are iid Gaussian random variables and $\{\mathbf{e}_k\}_{k\in\mathbb{Z}^d}$ is the standard Fourier basis (**SM** A.5). In this setting, we can prove the following approximation result, the proof of which can be found in **SM** D.3. The result can be generalized to other data distributions μ for which a convergence result for DeepONets can be proven, as in [43].

Theorem 4.6. *Assume the setting above and that of Assumption 3.4, and assume that $\mathcal{G}(v) \in C^{\ell+1}(\mathbb{T}^d)$ for all $v \in \mathcal{X}$. For all $\beta > 0$ there exists a constant $C > 0$ such that for any $p \in \mathbb{N}$ there exists a DeepONet $\mathcal{G}_\theta$ with p sensors and branch and trunk nets such that*

$$\left\|\mathcal{L}(\mathcal{G}_\theta))\right\|_{L^2(L^2(\mathbb{T}^d),\mu)} \leq Cp^{-\beta}. \tag{4.8}$$

Moreover, $\mathrm{size}(\boldsymbol{\tau}) \leq Cp^{\frac{d+1}{d}}$, $\mathrm{depth}(\boldsymbol{\tau}) = 3$, $\mathrm{size}(\boldsymbol{\beta}) \leq p$ *and* $\mathrm{depth}(\boldsymbol{\beta}) = 1$.

Nonlinear operators For nonlinear PDEs a general result like Theorem 4.6 can not be obtained from the currently available tools. Instead one needs to use Theorem 3.9 for every PDE of interest on a case-by-case basis. In the **SM**, we demonstrate this for a nonlinear ODE (gravity pendulum with external force, **SM** D.5) and an elliptic PDE (Darcy flow, **SM** D.6).

5 Related work and discussion

This is the first paper to rigorously expose the connections between the different deep learning frameworks from Section 2.2 for generic PDEs. Until now, most available results focus on providing generic results for one specific method. In [31] and [24] one uses neural networks that approximate solutions to a generic ODE/PDE at a fixed time to construct space-time neural networks. A generalization to PINNs is not immediate as the proof involves the emulation of the forward Euler method. We have overcome this difficulty by constructing space-time neural networks using Taylor expansions instead (Theorem 3.5). To bound the approximation error of PINNs one can use the generic error bounds in Sobolev norms of e.g. [25, 26] for very general activation functions or the more concrete bounds [15] for tanh neural networks. In both approaches, the only assumption is that the solution of the PDE has sufficient Sobolev regularity. As a consequence, these results incur the curse of dimensionality and are not applicable to high-dimensional PDEs. The authors of [15] analyze PINNs based on three theoretical questions related to approximation, stability and generalization. Other theoretical analyses of PINNs include e.g. [71, 72, 30]. For DeepONets, convergence rates for advection-diffusion equations are presented in [17] and a clear workflow for obtaining generic error estimates as well as worked out examples can be found in [43]. Similar results are obtained for FNOs in [38]. A comprehensive comparison of DeepONets and FNOs is the topic of [50]. To the best of

the authors' knowledge, no theoretical results for physics-informed operator learning are currently available. Unrelated to the approximation error, we also report generic bounds on the expected value of the generalization error of all the aforementioned deep learning architectures, in the form of an a posteriori error estimate on the generalization error.

A second goal of the paper is to prove that deep learning-based frameworks can overcome the curse of dimensionality (CoD). PDEs for which the curse of dimensionality has been overcome include linear Kolmogorov PDEs e.g. [23, 36], nonlinear parabolic PDEs [32] and elliptic PDEs [4, 10, 57]. By assuming that the initial data lies in a Barron class, the authors of [52] proved for elliptic PDEs that the Deep Ritz Method [20] can overcome the CoD. Since the Barron class is a Banach algebra [10] it is possible that our results, which mostly only involve multiplications and additions of neural networks, can be extended to Barron functions. For PINNs, it is proven that they can overcome the CoD for linear Kolmogorov PDEs [16]. We give an alternative proof of this result, improve the convergence rate (Theorem 4.2) and additionally prove that PINNs can also overcome the CoD for nonlinear parabolic PDEs (Theorem 4.4). DeepONets and FNOs can overcome the CoD in many cases [43, 38] but we note that this does not yet include nonlinear parabolic PDEs such as the Allen-Cahn equation. In Theorem 4.5 we prove that dimension-independent convergence rates can be obtained if the solution is sufficiently regular. Similar results are expected to hold for e.g. elliptic PDEs by using the results from [4, 10, 57].

It is evident that the generic bounds presented here can only be obtained under suitable assumptions. These should always be checked to prevent misleading claims about mathematical guarantees for the considered deep learning methods. We briefly discuss how restrictive these are and whether they can be relaxed. Assuming the existence of a neural network that approximates the solution of PDE at a fixed time (Assumption 3.1) is of course essential, but such a result can usually be obtained by emulating an existing numerical method. Proving a bound on the Sobolev norm of that network is always possible as we only consider smooth networks. Assumption 3.3 holds for many domains, including rectangular and smooth ones. Assumption 3.4 and Assumption 3.6 also hold for a very broad class of PDEs, much like the assumption on the size of the neural network approximation in Setting 4.1 and 4.3 holds for most functions of interest. Therefore, the assumption that the PDE solution is $C^{(s,r)}$-regular seems to be the most restrictive. However, results like Theorem 3.5 could be extended to e.g. Sobolev regular functions by using the Bramble-Hilbert lemma instead of Taylor expansions. Another restriction is that we exclusively focused on neural networks with the tanh activation function. This was only for simplicity of exposition. All results still hold for other sigmoidal activation functions, as well as more general smooth activation functions, which might give rise to slightly different convergence rates. A last restriction is that the obtained rates are not optimal, but this is not the goal of our framework. In particular, for PINNs for low-dimensional PDEs it is beneficial to use e.g. [26, 15].

Optimizing the obtained convergence rates and comparing with optimal ones is one direction for future research. Previously mentioned possibilities include extending to more general activation functions and less regular functions. Another direction is to make the connection between our results and that of [10] where they prove that Barron spaces are Banach algebras and use this to obtain dimension-independent convergence rates for PDEs with initial data in a Barron class by emulating numerical methods.

Here, we have considered the approximation and generalization errors in the present analysis. It is clear that the bounds on the generalization error may not be sharp, as in traditional deep learning. Obtaining sharper bounds will be an interesting topic for further investigation. Finally, there is no explicit bound on the training (optimization) errors. Obtaining such bounds will be considered in the future.

Acknowledgments and Disclosure of Funding

The research of SM was performed under a project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant Agreement No. 770880).

References

[1] S. M. Allen and J. W. Cahn. A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening. *Acta metallurgica*, 27(6):1085–1095, 1979.

[2] G. Bai, U. Koley, S. Mishra, and R. Molinaro. Physics informed neural networks (PINNs) for approximating nonlinear dispersive PDEs. *arXiv preprint arXiv:2104.05584*, 2021.

[3] A. Barth, A. Jentzen, A. Lang, and C. Schwab. *Numerical Analysis of Stochastic Ordinary Differential Equations*. ETH Zürich, 2018.

[4] C. Beck, L. Gonon, and A. Jentzen. Overcoming the curse of dimensionality in the numerical approximation of high-dimensional semilinear elliptic partial differential equations. *arXiv preprint arXiv:2003.00596*, 2020.

[5] C. Beck, F. Hornung, M. Hutzenthaler, A. Jentzen, and T. Kruse. Overcoming the curse of dimensionality in the numerical approximation of Allen-Cahn partial differential equations via truncated full-history recursive multilevel Picard approximations. *Journal of Numerical Mathematics*, 28(4):197–222, 2020.

[6] C. Beck, A. Jentzen, and B. Kuckuck. Full error analysis for the training of deep neural networks, 2020.

[7] J. Berner, P. Grohs, and A. Jentzen. Analysis of the generalization error: Empirical risk minimization over deep artificial neural networks overcomes the curse of dimensionality in the numerical approximation of Black-Scholes partial differential equations. *SIAM Journal on Mathematics of Data Science*, 2(3):631–657, 2020.

[8] S. Cai, Z. Wang, L. Lu, T. A. Zaki, and G. E. Karniadakis. DeepM&Mnet: Inferring the electroconvection multiphysics fields based on operator approximation by neural networks. *Journal of Computational Physics*, 436:110296, 2021.

[9] T. Chen and H. Chen. Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE Transactions on Neural Networks*, 6(4):911–917, 1995.

[10] Z. Chen, J. Lu, and Y. Lu. On the representation of solutions to elliptic PDEs in Barron spaces. *arXiv preprint arXiv:2106.07539*, 2021.

[11] A. Cohen, R. Devore, and C. Schwab. Analytic regularity and polynomial approximation of parametric and stochastic elliptic PDEs. *Analysis and Applications*, 9(01):11–47, 2011.

[12] G. Constantine and T. Savits. A multivariate Faa di Bruno formula with applications. *Transactions of the American Mathematical Society*, 348(2):503–520, 1996.

[13] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[14] T. De Ryck, A. D. Jagtap, and S. Mishra. Error estimates for physics informed neural networks approximating the Navier-Stokes equations. *arXiv preprint arXiv:2203.09346*, 2022.

[15] T. De Ryck, S. Lanthaler, and S. Mishra. On the approximation of functions by tanh neural networks. *Neural Networks*, 2021.

[16] T. De Ryck and S. Mishra. Error analysis for physics informed neural networks (PINNs) approximating Kolmogorov PDEs. *arXiv preprint arXiv:2106.14473*, 2021.

[17] B. Deng, Y. Shin, L. Lu, Z. Zhang, and G. E. Karniadakis. Convergence rate of DeepONets for learning operators arising from advection-diffusion equations. *arXiv preprint arXiv:2102.10621*, 2021.

[18] M. Dissanayake and N. Phan-Thien. Neural-network-based approximations for solving partial differential equations. *Communications in Numerical Methods in Engineering*, 1994.

[19] W. E, J. Han, and A. Jentzen. Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. *Communications in Mathematics and Statistics*, 5(4):349–380, 2017.

[20] W. E and B. Yu. The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems. *Communications in Mathematics and Statistics*, 6(1):1–12, 2018.

[21] R. A. Fisher. The wave of advance of advantageous genes. *Annals of eugenics*, 7(4):355–369, 1937.

[22] S. Goswami, M. Yin, Y. Yu, and G. E. Karniadakis. A physics-informed variational DeepONet for predicting crack path in quasi-brittle materials. *Computer Methods in Applied Mechanics and Engineering*, 391:114587, 2022.

[23] P. Grohs, F. Hornung, A. Jentzen, and P. Von Wurstemberger. A proof that artificial neural networks overcome the curse of dimensionality in the numerical approximation of Black-Scholes partial differential equations. *arXiv preprint arXiv:1809.02362*, 2018.

[24] P. Grohs, F. Hornung, A. Jentzen, and P. Zimmermann. Space-time error estimates for deep neural network approximations for differential equations. *arXiv preprint arXiv:1908.03833*, 2019.

[25] I. Gühring, G. Kutyniok, and P. Petersen. Error bounds for approximations with deep ReLU neural networks in $W^{s,p}$ norms. *Analysis and Applications*, 18(05):803–859, 2020.

[26] I. Gühring and M. Raslan. Approximation rates for neural networks with encodable weights in smoothness spaces. *Neural Networks*, 134:107–130, 2021.

[27] W. H. Guss and R. Salakhutdinov. On universal approximation by neural networks with uniform guarantees on approximation of infinite dimensional maps. *arXiv preprint arXiv:1910.01545*, 2019.

[28] P. Henry-Labordere. Counterparty risk valuation: A marked branching diffusion approach. *Available at SSRN 1995503*, 2012.

[29] P. Henry-Labordere, X. Tan, and N. Touzi. A numerical algorithm for a class of BSDEs via the branching process. *Stochastic Processes and their Applications*, 124(2):1112–1140, 2014.

[30] B. Hillebrecht and B. Unger. Certified machine learning: A posteriori error estimation for physics-informed neural networks. *arXiv preprint arXiv:2203.17055*, 2022.

[31] F. Hornung, A. Jentzen, and D. Salimova. Space-time deep neural network approximations for high-dimensional partial differential equations. *arXiv preprint arXiv:2006.02199*, 2020.

[32] M. Hutzenthaler, A. Jentzen, T. Kruse, and T. A. Nguyen. A proof that rectified deep neural networks overcome the curse of dimensionality in the numerical approximation of semilinear heat equations. *SN partial differential equations and applications*, 1(2):1–34, 2020.

[33] M. Hutzenthaler, A. Jentzen, B. Kuckuck, and J. L. Padgett. Strong L^p-error analysis of nonlinear Monte Carlo approximations for high-dimensional semilinear partial differential equations. *arXiv preprint arXiv:2110.08297*, 2021.

[34] A. D. Jagtap and G. E. Karniadakis. Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, 28(5):2002–2041, 2020.

[35] A. D. Jagtap, E. Kharazmi, and G. E. Karniadakis. Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. *Computer Methods in Applied Mechanics and Engineering*, 365:113028, 2020.

[36] A. Jentzen, D. Salimova, and T. Welti. A proof that deep artificial neural networks overcome the curse of dimensionality in the numerical approximation of Kolmogorov partial differential equations with constant diffusion and nonlinear drift coefficients. *arXiv preprint arXiv:1809.07321*, 2018.

[37] A. N. Kolmogorov. Étude de l'équation de la diffusion avec croissance de la quantité de matière et son application à un problème biologique. *Bull. Univ. Moskow, Ser. Internat., Sec. A*, 1:1–25, 1937.

[38] N. Kovachki, S. Lanthaler, and S. Mishra. On universal approximation and error bounds for Fourier Neural Operators. *arXiv preprint arXiv:2107.07562*, 2021.

[39] N. Kovachki, Z. Li, B. Liu, K. Azizzadensheli, K. Bhattacharya, A. Stuart, and A. Anandkumar. Neural operator: Learning maps between function spaces. *arXiv preprint arXiv:2108.08481v3*, 2021.

[40] G. Kutyniok, P. Petersen, M. Raslan, and R. Schneider. A theoretical analysis of deep neural networks and parametric PDEs. *Constructive Approximation*, pages 1–53, 2021.

[41] I. E. Lagaris, A. Likas, and P. G. D. Neural-network methods for boundary value problems with irregular boundaries. *IEEE Transactions on Neural Networks*, 11:1041–1049, 2000.

[42] I. E. Lagaris, A. Likas, and D. I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, 9(5):987–1000, 2000.

[43] S. Lanthaler, S. Mishra, and G. E. Karniadakis. Error estimates for DeepONets: A deep learning framework in infinite dimensions. *Transactions of Mathematics and Its Applications*, 6(1):tnac001, 2022.

[44] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar. Fourier neural operator for parametric partial differential equations, 2020.

[45] Z. Li, N. B. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. M. Stuart, and A. Anandkumar. Neural operator: Graph kernel network for partial differential equations. *CoRR*, abs/2003.03485, 2020.

[46] Z. Li, N. B. Kovachki, K. Azizzadenesheli, B. Liu, A. M. Stuart, K. Bhattacharya, and A. Anandkumar. Multipole graph neural operator for parametric partial differential equations. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 6755–6766. Curran Associates, Inc., 2020.

[47] Z. Li, H. Zheng, N. Kovachki, D. Jin, H. Chen, B. Liu, K. Azizzadenesheli, and A. Anandkumar. Physics-informed neural operator for learning partial differential equations. *arXiv preprint arXiv:2111.03794*, 2021.

[48] C. Lin, Z. Li, L. Lu, S. Cai, M. Maxey, and G. E. Karniadakis. Operator learning for predicting multiscale bubble growth dynamics. *The Journal of Chemical Physics*, 154(10):104118, 2021.

[49] L. Lu, P. Jin, G. Pang, Z. Zhang, and G. E. Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.

[50] L. Lu, X. Meng, S. Cai, Z. Mao, S. Goswami, Z. Zhang, and G. E. Karniadakis. A comprehensive and fair comparison of two neural operators (with practical extensions) based on fair data. *Computer Methods in Applied Mechanics and Engineering*, 393:114778, 2022.

[51] L. Lu, X. Meng, Z. Mao, and G. E. Karniadakis. DeepXDE: A deep learning library for solving differential equations. *SIAM Review*, 63(1):208–228, 2021.

[52] Y. Lu, J. Lu, and M. Wang. A priori generalization analysis of the deep Ritz method for solving high dimensional elliptic partial differential equations. In *Conference on Learning Theory*, pages 3196–3241. PMLR, 2021.

[53] K. O. Lye, S. Mishra, and D. Ray. Deep learning observables in computational fluid dynamics. *Journal of Computational Physics*, page 109339, 2020.

[54] K. O. Lye, S. Mishra, D. Ray, and P. Chandrashekar. Iterative surrogate model optimization (ISMO): An active learning algorithm for pde constrained optimization with deep neural networks. *Computer Methods in Applied Mechanics and Engineering*, 374:113575, 2021.

[55] Z. Mao, A. D. Jagtap, and G. E. Karniadakis. Physics-informed neural networks for high-speed flows. *Computer Methods in Applied Mechanics and Engineering*, 360:112789, 2020.

[56] Z. Mao, L. Lu, O. Marxen, T. A. Zaki, and G. E. Karniadakis. DeepM&Mnet for hypersonics: Predicting the coupled flow and finite-rate chemistry behind a normal shock using neural-network approximation of operators. *Journal of Computational Physics*, 447:110698, 2021.

[57] T. Marwah, Z. Lipton, and A. Risteski. Parametric complexity bounds for approximating pdes with neural networks. *Advances in Neural Information Processing Systems*, 34:15044–15055, 2021.

[58] H. P. McKean. Application of brownian motion to the equation of Kolmogorov-Petrovskii-Piskunov. *Communications on pure and applied mathematics*, 28(3):323–331, 1975.

[59] S. Mishra and R. Molinaro. Estimates on the generalization error of physics-informed neural networks for approximating a class of inverse problems for PDEs. *IMA Journal of Numerical Analysis*, 2021.

[60] S. Mishra and R. Molinaro. Physics informed neural networks for simulating radiative transfer. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 270:107705, 2021.

[61] S. Mishra and R. Molinaro. Estimates on the generalization error of physics informed neural networks (PINNs) for approximating PDEs. *IMA Journal of Numerical Analysis*, 2022.

[62] B. Øksendal. *Stochastic differential equations*. Springer, 2003.

[63] J. A. Opschoor, P. C. Petersen, and C. Schwab. Deep ReLU networks and high-order finite element methods. *Analysis and Applications*, 18(05):715–770, 2020.

[64] J. A. Opschoor, C. Schwab, and J. Zech. Exponential ReLU DNN expression of holomorphic maps in high dimension. *Constructive Approximation*, pages 1–46, 2021.

[65] G. Pang, L. Lu, and G. E. Karniadakis. fPINNs: Fractional physics-informed neural networks. *SIAM journal of Scientific computing*, 41:A2603–A2626, 2019.

[66] J. Pathak, S. Subramanian, P. Harrington, S. Raja, A. Chattopadhyay, M. Mardani, T. Kurth, D. Hall, Z. Li, K. Azizzadenesheli, p. Hassanzadeh, K. Kashinath, and A. Anandkumar. Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators. *arXiv preprint arXiv:2202.11214*, 2022.

[67] M. Raissi and G. E. Karniadakis. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125–141, 2018.

[68] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[69] M. Raissi, A. Yazdani, and G. E. Karniadakis. Hidden fluid mechanics: A Navier-Stokes informed deep learning framework for assimilating flow visualization data. *arXiv preprint arXiv:1808.04327*, 2018.

[70] C. Schwab and J. Zech. Deep learning in high dimension: Neural network expression rates for generalized polynomial chaos expansions in uq. *Analysis and Applications*, 17(01):19–55, 2019.

[71] Y. Shin, J. Darbon, and G. E. Karniadakis. On the convergence and generalization of physics informed neural networks. *arXiv preprint arXiv:2004.01806*, 2020.

[72] Y. Shin, Z. Zhang, and G. E. Karniadakis. Error estimates of residual minimization using neural networks for linear equations. *arXiv preprint arXiv:2010.08019*, 2020.

[73] S. Wang and P. Perdikaris. Long-time integration of parametric evolution equations with physics-informed DeepONets. *arXiv preprint arXiv:2106.05384*, 2021.

[74] S. Wang, H. Wang, and P. Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed DeepOnets. *arXiv preprint arXiv:2103.10974*, 2021.

[75] J. Yang, Q. Du, and W. Zhang. Uniform L^p-bound of the Allen-Cahn equation and its numerical discretization. *International Journal of Numerical Analysis & Modeling*, 15, 2018.

[76] L. Yang, X. Meng, and G. E. Karniadakis. B-PINNs: Bayesian physics-informed neural networks for forward and inverse pde problems with noisy data. *Journal of Computational Physics*, 425:109913, 2021.

[77] D. Yarotsky. Error bounds for approximations with deep ReLU networks. *Neural Networks*, 94:103–114, 2017.

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] See Section 3 and Section 4.
   (b) Did you describe the limitations of your work? [Yes] See Section 5.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 5.
   (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] All assumptions are either stated in the theorem statement or described in the text right above the theorem statement.
   (b) Did you include complete proofs of all theoretical results? [Yes] For each result we mention where the proof can be found.
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)? [N/A]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [N/A]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (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)? [N/A]
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? [N/A]
   (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? [N/A]
   (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]

A Notation and preliminaries

We introduce notation and preliminary results regarding finite differences, Sobolev spaces, the Legendre basis, the Fourier basis, trigonometric polynomial interpolation and neural network approximation theory.

A.1 Overview of used notation

Table 1: Glossary of used notation.

Symbol	Description	Page
σ	tanh activation function	
d	spatial dimension of domain	
$\mathbb{T}^d$	periodic torus, identified with $[0, 2\pi)^d$	
D	general d-dimensional spatial domain	p. 2
Ω	general domain, either $\Omega = D$ or $\Omega = [0, T] \times D$	p. 2
$\partial\Omega$	boundary of Ω	
$\mathcal{H}$	function space of PDE solution	p. 2
$\mathcal{Y}$	function space of parameters for $\mathcal{L}$, e.g. $\mathcal{L}_a$ with $a \in \mathcal{Y}$	p. 2
$\mathcal{Z}$	function space of initial conditions for the PDE (2.1)	p. 2
$\mathcal{X}$	input function space of the operator $\mathcal{G}$	p. 2
$\mathcal{G}$	operator of interest, $\mathcal{G} : \mathcal{X} \to L^2(\Omega)$	p. 2
$\mathcal{L}$, $\mathcal{L}_a$	differential operator that describes the PDE (with parameter a)	p. 2
r, s	regularity of the PDE solution, $u \in C^r(D)$ or $u \in C^{(s,r)}([0, T] \times D)$	p. 3
$D^{(k,\boldsymbol{\alpha})}$	$D^{(k,\boldsymbol{\alpha})} := D_t^k D_x^{\boldsymbol{\alpha}} := \partial_t^k \partial_{x_1}^{\alpha_1} \dots \partial_{x_d}^{\alpha_d}$, for $(k, \boldsymbol{\alpha}) \in \mathbb{N}_0^{d+1}$	p. 5
ℓ	upper bound on $\lVert\boldsymbol{\alpha}\rVert_1$	p. 5
q	see Assumption 3.1	p. 4
p	see Assumption 3.6	p. 5
$\mathrm{poly}(d)$	a polynomial in d	p. 5
C_0^r	subset of C^r functions with compact support	
$\Delta_h^{\boldsymbol{\alpha},r}$	finite difference operator; if the variable is time: $\Delta_{h,t}^{\boldsymbol{\alpha},s}$	p. 15
$\mathcal{J}_N$	grid point indices, $\mathcal{J}_N = \{0, \dots, 2N\}^d$	p. 16
$\mathcal{K}_N$	Fourier wavenumbers $\mathcal{K}_N = \{k \in \mathbb{Z}^d \mid \lvert k\rvert_\infty \leq N\}$	
L^2	Space of square-integrable functions	
H^s	Sobolev space of smoothness s, with norm $\lVert \cdot \rVert_{H^s}$	p. 16
L_N^2	$L_N^2 \subset L^2$ trigonometric polynomials of degree $\leq N$	p. 17

A.2 Finite differences

For $h > 0$, $\boldsymbol{\alpha} \in \mathbb{N}_0^d$, $r \in \mathbb{N}$ and $\ell := \lVert\boldsymbol{\alpha}\rVert_1$, we define a finite difference operator $\Delta_h^{\boldsymbol{\alpha},r}$ as,

$$\Delta_h^{\boldsymbol{\alpha},r}[f](t, x) = \sum_j c_j^{\boldsymbol{\alpha},r} f(t, x + h b_j^{\boldsymbol{\alpha},r}), \tag{A.1}$$

for $f \in C^{r+\ell}(\mathbb{R}^d)$, where the number of non-zero terms in the summation can be chosen to be finite and only dependent on ℓ and r and where the choice of $b_j^{\boldsymbol{\alpha},r} \in \mathbb{R}^d$ allows to approximate $D_x^{\boldsymbol{\alpha}} f$ up to accuracy $\mathcal{O}(h^r)$. This means that for any $f \in C^{r+\ell}(\mathbb{R}^d)$ it holds for all x that,

$$\left| h^{-\ell} \cdot \Delta_h^{\boldsymbol{\alpha},r}[f](t, x) - D_x^{\boldsymbol{\alpha}} f(t, x) \right| \leq c_{\ell,r} \left| f(t, \cdot) \right|_{C^{r+\ell}} h^r \quad \text{for } h > 0, \tag{A.2}$$

where $c_{\ell,r} > 0$ does not depend on f and h. Similarly, we can define a finite difference operator $\Delta_{h,t}^{k,s}[f](t, x)$ to approximate $D_t^k f(t, x)$ to accuracy $\mathcal{O}(h^s)$.

A.3 Sobolev spaces

Let $d \in \mathbb{N}$, $k \in \mathbb{N}_0$, $1 \leq p \leq \infty$ and let $\Omega \subseteq \mathbb{R}^d$ be open. For a function $f : \Omega \to \mathbb{R}$ and a (multi-)index $\boldsymbol{\alpha} \in \mathbb{N}_0^d$ we denote by

$$D^{\boldsymbol{\alpha}} f = \frac{\partial^{|\boldsymbol{\alpha}|} f}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}} \tag{A.3}$$

the classical or distributional (i.e. weak) derivative of f. We denote by $L^p(\Omega)$ the usual Lebesgue space and for we define the Sobolev space $W^{k,p}(\Omega)$ as

$$W^{k,p}(\Omega) = \{f \in L^p(\Omega) : D^{\boldsymbol{\alpha}} f \in L^p(\Omega) \text{ for all } \boldsymbol{\alpha} \in \mathbb{N}_0^d \text{ with } |\boldsymbol{\alpha}| \leq k\}. \tag{A.4}$$

For $p < \infty$, we define the following seminorms on $W^{k,p}(\Omega)$,

$$|f|_{W^{m,p}(\Omega)} = \left(\sum_{|\boldsymbol{\alpha}|=m} \|D^{\boldsymbol{\alpha}} f\|_{L^p(\Omega)}^p \right)^{1/p} \qquad \text{for } m = 0, \ldots, k, \tag{A.5}$$

and for $p = \infty$ we define

$$|f|_{W^{m,\infty}(\Omega)} = \max_{|\boldsymbol{\alpha}|=m} \|D^{\boldsymbol{\alpha}} f\|_{L^\infty(\Omega)} \qquad \text{for } m = 0, \ldots, k. \tag{A.6}$$

Based on these seminorms, we can define the following norm for $p < \infty$,

$$\|f\|_{W^{k,p}(\Omega)} = \left(\sum_{m=0}^{k} |f|_{W^{m,p}(\Omega)}^p \right)^{1/p}, \tag{A.7}$$

and for $p = \infty$ we define the norm

$$\|f\|_{W^{k,\infty}(\Omega)} = \max_{0 \leq m \leq k} |f|_{W^{m,\infty}(\Omega)}. \tag{A.8}$$

The space $W^{k,p}(\Omega)$ equipped with the norm $\|\cdot\|_{W^{k,p}(\Omega)}$ is a Banach space.

We denote by $C^k(\Omega)$ the space of functions that are k times continuously differentiable and equip this space with the norm $\|f\|_{C^k(\Omega)} = \|f\|_{W^{k,\infty}(\Omega)}$.

Lemma A.1 (Continuous Sobolev embedding)**.** *Let $d, \ell \in \mathbb{N}$ and let $k \geq d/2 + \ell$. Then there exists a constant $C > 0$ such that for any $f \in H^k(\mathbb{T}^d)$ it holds that*

$$\|f\|_{C^\ell(\mathbb{T}^d)} \leq C \|f\|_{H^k(\mathbb{T}^d)}. \tag{A.9}$$

A.4 Notation for Legendre basis

In a one-dimensional setting, we denote for $j \in \mathbb{N}_0$ the j-th Legendre polynomial by L_j. Following the notation of [64], it holds that $L_j(x) = \sum_{j=0}^{\ell} c_\ell^j x^\ell$ where, with $m(\ell) := (j - \ell)/2$,

$$c_\ell^j = \begin{cases} 0 & j - \ell \{0, \ldots, j\} \cup (2\mathbb{Z} + 1), \\ (-1)^m 2^{-j} \binom{j}{m} \binom{j+\ell}{j} \sqrt{2j+1} & j - \ell \{0, \ldots, j\} \cup 2\mathbb{Z}, \end{cases} \tag{A.10}$$

where each polynomial is normalized in $L^2([-1, 1], \lambda/2)$, where λ is the Lebesgue measure. Similarly, the tensorized Legendre polynomials,

$$L_{\boldsymbol{\nu}}(x) = \prod_{j=1}^{d} L_{\nu_j}(x_j) \qquad \text{for all } \boldsymbol{\nu} \in \mathbb{N}_0^d, \tag{A.11}$$

constitute an orthonormal basis of $L^2([-1, 1]^d, \lambda/2^d)$. By considering the lexicographic order on $\mathbb{N}_0^d$, of which we denote the enumeration by $\kappa : \mathbb{N} \to \mathbb{N}_0^d$, one can defined an ordered basis $(L_j)_{j \in \mathbb{N}}$ by setting $L_j := L_{\kappa(j)}$.

From [64, eq. (2.19)] it also follows that,

$$\forall s \in \mathbb{N}_0, \boldsymbol{\nu} \in \mathbb{N}_0^d : \|L_{\boldsymbol{\nu}}\|_{C^s([-1,1]^d)} \leq \prod_{j=1}^{d} (1 + 2\nu_j)^{1/2+2s}. \tag{A.12}$$

A.5 Notation for Standard Fourier basis

Using the notation from [43], we introduce the following "standard" real Fourier basis $\{\mathbf{e}_\kappa\}_{\kappa\in\mathbb{Z}^d}$ in d dimensions. For $\kappa = (\kappa_1, \dots, \kappa_d) \in \mathbb{Z}^d$, we let $\sigma(\kappa)$ be the sign of the first non-zero component of κ and we define

$$\mathbf{e}_\kappa := C_\kappa \begin{cases} 1, & \sigma(\kappa) = 0, \\ \cos\big(\langle \kappa, x\rangle\big), & \sigma(\kappa) = 1, \\ \sin\big(\langle \kappa, x\rangle\big), & \sigma(\kappa) = -1, \end{cases} \tag{A.13}$$

where the factor $C_\kappa > 0$ ensures that $\mathbf{e}_\kappa$ is properly normalized, i.e. that $\|\mathbf{e}_\kappa\|_{L^2(\mathbb{T}^d)} = 1$. Next, let $\kappa : \mathbb{N} \to \mathbb{Z}^d$ be a fixed enumeration of $\mathbb{Z}^d$, with the property that $j \mapsto |\kappa(j)|_\infty$ is monotonically increasing, i.e. such that $j \le j'$ implies that $|\kappa(j)|_\infty \le |\kappa(j')|_\infty$. This will allow us to introduce an $\mathbb{N}$-indexed version of the Fourier basis,

$$\mathbf{e}_j(x) := \mathbf{e}_{\kappa(j)}(x), \quad \forall j \in \mathbb{N}. \tag{A.14}$$

Finally we note that

$$\|\mathbf{e}_\kappa\|_{C^s([0,2\pi]^d)} \le \|\kappa\|_\infty^s. \tag{A.15}$$

A.6 Trigonometric polynomial interpolation

For $N \in \mathbb{N}$, let $x_j = \frac{2\pi j}{2N+1}$ and let $y_j \in \mathbb{R}$ for all $j \in \mathcal{J}_N = \{0, \dots, 2N+1\}^d$. We will construct an operator

$$\mathcal{Q}_N : \mathbb{R}^{|\mathcal{J}_N|} \to L^2(\mathbb{T}^d) : y \mapsto \mathcal{Q}_N(y), \tag{A.16}$$

where $\mathcal{Q}_N(y)$ is a trigonometric polynomial of degree at most N such that $\mathcal{Q}_N(y)(x_j) = y_j$ for all $j \in \mathcal{J}_N$. We construct this polynomial using the discrete Fourier transform and its inverse. For $k \in \mathcal{K}_N = \{-N, \dots, N\}^d$, we define the discrete Fourier transform as,

$$X_k(y) = \sum_{j\in\mathcal{J}_N} y_j \exp\big(-i\langle k, x_j\rangle\big), \tag{A.17}$$

and the trigonometric interpolation polynomial as,

$$\begin{aligned} \mathcal{Q}_N(y)(z) &= \frac{1}{|\mathcal{K}_N|} \sum_{k\in\mathcal{K}_N} X_k(y) \exp\big(i\langle k, z\rangle\big) \\ &= \frac{1}{|\mathcal{K}_N|} \sum_{k\in\mathcal{K}_N} \sum_{j\in\mathcal{J}_N} y_j \cos\big(\langle k, z - x_j\rangle\big) \\ &= \frac{1}{|\mathcal{K}_N|} \sum_{k\in\mathcal{K}_N} \sum_{j\in\mathcal{J}_N} y_j \Big(\cos\big(\langle k, x_j\rangle\big)\cos\big(\langle k, z\rangle\big) - \sin\big(\langle k, x_j\rangle\big)\sin\big(\langle k, z\rangle\big)\Big) \\ &= \frac{1}{|\mathcal{K}_N|} \sum_{k\in\mathcal{K}_N} \sum_{j\in\mathcal{J}_N} y_j a_{k,j} \mathbf{e}_k(z), \end{aligned} \tag{A.18}$$

where,

$$a_{k,j} = \begin{cases} 1, & \sigma(k) = 0, \\ \cos\big(\langle k, x_j\rangle\big), & \sigma(k) = 1, \\ \sin\big(\langle k, x_j\rangle\big), & \sigma(k) = -1, \end{cases} \tag{A.19}$$

with σ as in **SM** A.5. We can also define an encoder $\mathcal{E}_N$ by,

$$\mathcal{E}_N : C(\mathbb{T}^d) \to \mathbb{R}^{|\mathcal{J}_N|} : f \mapsto (f(x_j))_{j\in\mathcal{J}_N}. \tag{A.20}$$

The composition $\mathcal{Q}_N \circ \mathcal{E}_N$ is called the pseudo-spectral projection onto the space of trigonometric polynomials of degree at most N and has the following property [38].

Lemma A.2. *For $s, k \in \mathbb{N}_0$ with $s > d/2$ and $s \ge k$, and $f \in C^s(\mathbb{T}^d)$ it holds that*

$$\big\|f - (\mathcal{Q}_N \circ \mathcal{E}_N)(f)\big\|_{H^k(\mathbb{T}^d)} \le C(s,d) N^{-(s-k)} \|f\|_{H^s(\mathbb{T}^d)}, \tag{A.21}$$

for a constant $C(s,d) > 0$ that only depends on s and d.

A.7 Neural network approximation theory

We recall some basic results on the approximation of functions by tanh neural networks in this section. All results are adaptations from results in [15]. The following two lemmas address the approximation of univariate monomials and the multiplication operator.

Lemma A.3 (Approximation of univariate monomials, Lemma 3.2 in [15])**.** *Let $k \in \mathbb{N}_0, s \in 2\mathbb{N}-1$, $M > 0$ and define $f_p : [-M, M] \to \mathbb{R} : x \mapsto x^p$ for all $p \in \mathbb{N}$. For every $\varepsilon > 0$, there exists a shallow tanh neural network $\psi_{s,\varepsilon} : [-M, M] \to \mathbb{R}^s$ of width $\frac{3(s+1)}{2}$ such that*

$$\max_{p \le s} \left\| f_p - (\psi_{s,\varepsilon})_p \right\|_{W^{k,\infty}} \le \varepsilon. \tag{A.22}$$

Lemma A.4 (Shallow approximation of multiplication of d numbers, Corollary 3.7 in [15])**.** *Let $d \in \mathbb{N}$, $k \in \mathbb{N}_0$ and $M > 0$. Then for every $\varepsilon > 0$, there exist a shallow tanh neural network $\widehat{\times}_d^{\varepsilon} : [-M, M]^d \to \mathbb{R}$ of width $3 \left\lceil \frac{d+1}{2} \right\rceil \left| P_{d,d} \right|$ (or 4 if $d = 2$) such that*

$$\left\| \widehat{\times}_d^{\varepsilon}(x) - \prod_{i=1}^{d} x_i \right\|_{W^{k,\infty}} \le \varepsilon. \tag{A.23}$$

B Additional material for Section 3

B.1 Auxiliary results for Section 3

Lemma B.1. *Let $q \in [1, \infty]$, $r, \ell \in \mathbb{N}$ with $\ell \le r$ and $f_1, f_2 \in C^{(0,r)}([0, T] \times D)$. If Assumption 3.3 holds then there exists a constant $C(r) > 0$ such that for any $\boldsymbol{\alpha} \in \mathbb{N}_0^d$ with $\ell := \|\boldsymbol{\alpha}\|_1$ it holds that*

$$\left\| D_x^{\boldsymbol{\alpha}}(f_1 - f_2) \right\|_{L^q} \le C(\|f_1 - f_2\|_{L^q} h^{-\ell} + \max_{j=1,2} \left| f_j \right|_{C^{(0,r)}} h^{r-\ell}) \qquad \forall h > 0. \tag{B.1}$$

Proof. From the triangle inequality and (A.2) the existence of a constant $C(r) > 0$ follows such that,

$$\begin{aligned} \left\| D_x^{\boldsymbol{\alpha}}(f_1 - f_2) \right\|_{L^q} &\le \max_{j=1,2} \left\| D^{\boldsymbol{\alpha}} f_j - h^{-\ell} \cdot \Delta_h^{\boldsymbol{\alpha},r}[f_j] \right\|_{L^q} + C(r) h^{-\ell} \|f_1 - f_2\|_{L^q} \\ &\le c_{\ell,r} \max_{j=1,2} \left| f_j \right|_{C^{(0,r)}} h^{r-\ell} + C(r) h^{-\ell} \|f_1 - f_2\|_{L^q}. \end{aligned}$$

□

Lemma B.2. *Using the notation of the proof of Theorem 3.5 (**SM** B.2), it holds that*

$$\left\| D^{(k,\boldsymbol{\alpha})}(\widetilde{u} - \widehat{u}) \right\|_{C^0} \le \delta. \tag{B.2}$$

Proof. Using the Faà di Bruno formula [12] and its consequences for estimating the norms of derivatives of compositions [15, Lemma A.7] one can prove for sufficiently regular functions g_1, g_2, h_1, h_2 and a suitable multi-index $\boldsymbol{\beta}$ estimates of the form,

$$\left\| D^{\boldsymbol{\beta}}(g_1 \circ h_1 - g_2 \circ h_2) \right\|_{C^0} \le C(\|g_1 - g_2\|_{C^{\|\boldsymbol{\beta}\|_1}} + \|h_1 - h_2\|_{C^{\|\boldsymbol{\beta}\|_1}}), \tag{B.3}$$

assuming that the compositions are well-defined and where the constant $C > 0$ may depend on g_1, g_2, h_1, h_2 and their derivatives. Using this theorem we can prove that

$$\left\| D^{(k,\boldsymbol{\alpha})} \widehat{u} - D^{(k,\boldsymbol{\alpha})} \sum_{m=1}^{M} \sum_{i=0}^{s-1} \frac{\Delta_{1/M,t}^{i,s-i}[\widehat{u}_m^{\varepsilon}](t_m, x)}{M^{-i} i!} \cdot \widehat{\varphi}_i^{\delta}(t - t_m) \cdot \Phi_m^M(t) \right\| < C\delta. \tag{B.4}$$

Because the size of the neural network $\widehat{\times}_\delta$ in the definition of $\widehat{u}$ does not depend on its accuracy δ (see Lemma A.4) we can rescale δ and therefore set $C = 1/2$ in the above inequality.

Next, we observe that,

$$\begin{aligned} &D^{(k,\boldsymbol{\alpha})} \sum_{m=1}^{M} \sum_{i=0}^{s-1} \frac{\Delta_{1/M,t}^{i,s-i}[\widehat{u}_m^{\varepsilon}](t_m, x)}{M^{-i} i!} \cdot (\widehat{\varphi}_i^{\delta} - \varphi_i)(t - t_m) \cdot \Phi_m^M(t) \\ &= \sum_{m=1}^{M} \sum_{i=0}^{s-1} \frac{\Delta_{1/M,t}^{i,s-i}[D_x^{\boldsymbol{\alpha}} \widehat{u}_m^{\varepsilon}](t_m, x)}{M^{-i} i!} \cdot \sum_{n=0}^{k} \binom{k}{n} \partial_t^n (\widehat{\varphi}_i^{\delta} - \varphi_i)(t - t_m) \cdot \partial_t^{k-n} \Phi_m^M(t) \end{aligned} \tag{B.5}$$

Analogously to before, because the sizes of the neural networks $\widehat{\varphi}_i^\delta$ are independent of their accuracy δ we can rescale δ such that $\|(\text{B.5})\|_{C^0} \leq \delta/2$. The claim follows by the triangle inequality,

$$\left\| D^{(k,\boldsymbol{\alpha})}(\widetilde{u} - \widehat{u}) \right\|_{C^0} \leq \|(\text{B.4})\|_{C^0} + \|(\text{B.5})\|_{C^0} \leq \delta. \tag{B.6}$$

□

Lemma B.3. *Let $\Delta_{h,t}^{k,s}$ be a finite difference operator cf. Section 3.3 and **SM** A.2, let $1 \leq j \leq d$, let $1 \leq q \leq \infty$, let $\ell \in \mathbb{N}_0$ and let $\boldsymbol{\alpha} \in \mathbb{N}_0^d$ with $\|\boldsymbol{\alpha}\|_1 = \ell$. Let $u, \widehat{u} \in C^{(s,\ell)}([-2h, 2h] \times D)$ such that for all $t \in [-2h, 2h]$,*

$$\left\| D_x^{\boldsymbol{\alpha}}(u(t,\cdot) - \widehat{u}(t,\cdot)) \right\|_{L^q(D)} \leq \varepsilon. \tag{B.7}$$

Then there exists $c_s > 0$ holds that,

$$\left\| D^{k,\boldsymbol{\alpha}} \left(\sum_{i=0}^{s-1} \frac{\Delta_{h,t}^{i,s-i}[\widehat{u}](0,x)}{h^i i!} t^i - u(t,\cdot) \right) \right\|_{L^q} \leq c_s \left(\varepsilon h^{-k} + |D_x^{\boldsymbol{\alpha}} u|_{C^{(s,0)}} h^{s-k} \right). \tag{B.8}$$

Proof. Let $t \in [-2h, 2h]$, $\boldsymbol{\alpha} \in \mathbb{N}_0^d$ with $\|\boldsymbol{\alpha}\|_1 = \ell$ and $x \in \mathbb{R}^d$ be arbitrary. We first observe that,

$$D^{k,\boldsymbol{\alpha}} \sum_{i=0}^{s-1} \frac{\Delta_{h,t}^{i,s-i}[u](0,x)}{h^i i!} t^i = \sum_{i=k}^{s-1} \frac{\Delta_{h,t}^{i,s-i}[D_x^{\boldsymbol{\alpha}} u](0,x)}{h^i (i-k)!} t^{i-k}. \tag{B.9}$$

Taylor's theorem then guarantees the existence of $\xi_{t,x} \in [-2h, 2h]$ such that

$$\begin{aligned} &D^{k,\boldsymbol{\alpha}} \left(\sum_{i=0}^{s-1} \frac{\Delta_{h,t}^{i,s-i}[u](0,x)}{h^i i!} t^i - u(t,\cdot) \right) \\ &= \sum_{i=0}^{s-1-k} \left[\frac{\Delta_{h,t}^{i+k,s-i-k}[D_x^{\boldsymbol{\alpha}} u](0,x)}{h^{i+k} i!} t^i - \frac{D^{i+k,\boldsymbol{\alpha}} u(0,x)}{i!} t^i \right] + \frac{D^{s,\boldsymbol{\alpha}} u(\xi_{t,x}, x)}{(s-k)!} t^{s-k}. \end{aligned} \tag{B.10}$$

Now observe that because of assumption (B.7) and the definition and properties (A.2) of the finite difference operator, there exists a constant $C_s > 0$ such that,

$$\begin{aligned} &\left\| \Delta_{h,t}^{i+k,s-i-k}[D_x^{\boldsymbol{\alpha}} \widehat{u}](0,x) - \Delta_{h,t}^{i+k,s-i-k}[D_x^{\boldsymbol{\alpha}} u](0,x) \right\|_{L^q} \leq C_s \varepsilon, \\ &\left| \frac{\Delta_{h,t}^{i+k,s-i-k}[D_x^{\boldsymbol{\alpha}} u](0,x)}{h^{i+k}} - D^{i+k,\boldsymbol{\alpha}} u(0,x) \right| \leq C_s |D_x^{\boldsymbol{\alpha}} u|_{C^{(s,0)}} h^{r-i-k}. \end{aligned} \tag{B.11}$$

Combining all previous results provides us with the existence of a constant $c_s > 0$ such that,

$$\begin{aligned} &\left\| D^{k,\boldsymbol{\alpha}} \left(\sum_{i=0}^{s-1} \frac{\Delta_{h,t}^{i,s-i}[\widehat{u}](0,x)}{h^i i!} t^i - u(t,\cdot) \right) \right\|_{L^q} \\ &\leq \sum_{i=0}^{s-1-k} \left[\frac{C_s \varepsilon}{h^{i+k} i!} h^i + \frac{C_s}{i!} |D_x^{\boldsymbol{\alpha}} u|_{C^{(s,0)}} h^{s-i-k} h^i \right] + \frac{1}{(s-k)!} |D_x^{\boldsymbol{\alpha}} u|_{C^{(s,0)}} h^{s-k} \\ &\leq c_s \left(\varepsilon h^{-k} + |D_x^{\boldsymbol{\alpha}} u|_{C^{(s,0)}} h^{s-k} \right). \end{aligned} \tag{B.12}$$

□

Definition B.4. *Let $C > 0$, $N \in \mathbb{N}$, $0 < \varepsilon < 1$ and $\alpha = \ln\big(CN^k/\varepsilon\big)$. For every $1 \le j \le N$, we define the function $\Phi_j^N : [0,T] \to [0,1]$ by*

$$\begin{aligned}
\Phi_1^N(t) &= \frac{1}{2} - \frac{1}{2}\sigma\left(\alpha\left(t - \frac{T}{N}\right)\right),\\
\Phi_j^N(t) &= \frac{1}{2}\sigma\left(\alpha\left(t - \frac{T(j-1)}{N}\right)\right) - \frac{1}{2}\sigma\left(\alpha\left(t - \frac{Tj}{N}\right)\right),\\
\Phi_N^N(t) &= \frac{1}{2}\sigma\left(\alpha\left(t - \frac{T(N-1)}{N}\right)\right) + \frac{1}{2}.
\end{aligned} \tag{B.13}$$

The functions $\{\Phi_j^N\}_j$ approximate a partition of unity in the sense that for every j it holds on I_j^N that for some $\varepsilon > 0$,

$$1 - \sum_{v=-1}^{1} \Phi_{j+v}^N \lesssim \varepsilon \quad \text{and} \quad \sum_{\substack{|v|\ge 2,\\ j+v\in\{1,\dots,N\}}} \Phi_{j+v}^N \lesssim \varepsilon. \tag{B.14}$$

This is made exact in [15, Section 4].

Theorem B.5. *Let $k \in \mathbb{N} \cup \{0\}$, $q \in \{2, \infty\}$, $\xi > 0$ and $s \in \mathbb{N}$. Let μ be a probability measure on D and let $f \in C^s([0,T], L^q(\mu))$. Assume that for every $0 \le \ell \le k$ there is a constant $\mathcal{C}_\ell^* > 0$ for which it holds that for every $N \in \mathbb{N}$ there exist functions $\{p_j^N\}_{j=1}^N$ that satisfy for all $1 \le j \le N$,*

$$\left|f - p_j^N\right|_{C^\ell(J_j^N, L^q(\mu))} = \max_{t\in\left[\frac{(j-2)T}{N}, \frac{(j+1)T}{N}\right]} \left\|D_t^\ell(f(t,\cdot) - p_j^N(t,\cdot))\right\|_{L^q(\mu)} \le \mathcal{C}_\ell^* N^{-s+\ell} + \xi. \tag{B.15}$$

Let $\mathcal{C}_k := \max\{\max_{0\le\ell\le k}\mathcal{C}_\ell^*, \|f\|_{C^k([0,T],L^q(\mu))}, 1\}$. There exists a constant $C(k) > 0$ that only depends on k such that for all $N \ge 3$ it holds that,

$$\left\|f - \sum_{j=1}^N p_j^N \cdot \Phi_j^N\right\|_{C^k([0,T],L^q(\mu))} \le C\ln^k(N)\left[\frac{\mathcal{C}_k}{N^{s-k}} + \xi N^k\right]. \tag{B.16}$$

Proof. We follow the proof of [15, Theorem 5.1]. All steps of the proofs are identical, with the only difference being that the $W^{k,\infty}([0,1]^d)$-norm of [15] is replaced by the $C^k([0,T], L^2(\mu))$-norm in this work. Following [15], one divides the domain $[0,T]$ into intervals $I_i^N = [t_{i-1}, t_i]$, with $t_i = iT/N$ and $N \in \mathbb{N}$ large enough. On each of these intervals, f locally can be approximated (in Sobolev norm) by p_j^N, by virtue of the assumptions of the theorem. A global approximation can then be constructed by multiplying each p_j^N with an approximation of the indicator function of the corresponding intervals and summing over all intervals.

We now highlight the main steps in the proof. *Step 2a* (as in [15]) results in the following estimate,

$$\left\|f - \sum_{j=1}^N f\cdot\Phi_j^N\right\|_{C^k(I_i^N, L^q(\mu))} \le C\|f\|_{C^k(I_i^N, L^q(\mu))}\left(\varepsilon + N^{k+1}\ln^k\left(\frac{CN^k}{\varepsilon}\right)\varepsilon\right). \tag{B.17}$$

Step 2b results in the estimate,

$$\left\|\sum_{j=1}^N (f - p_j^N)\cdot\Phi_j^{N,d}\right\|_{C^k(I_i^N, L^q(\mu))} \le C\ln^k\left(\frac{CN^k}{\varepsilon}\right)\left[\frac{\mathcal{C}_k}{N^{s-k}} + \xi N^k + \mathcal{C}_k N^{k+1}\varepsilon\right], \tag{B.18}$$

Putting everything together, we find that if $CN^k \ge \varepsilon e$,

$$\begin{aligned}
&\left\|f - \sum_{j=1}^N p_j^N\cdot\Phi_j^N\right\|_{C^k([0,T],L^q(\mu))}\\
&\quad\le C\ln^k\left(\frac{CN^k}{\varepsilon}\right)\left[(\|f\|_{C^k(I_i^N,L^q(\mu))} + \mathcal{C}_k)N^{k+1}\varepsilon + \frac{\mathcal{C}_k}{N^{s-k}} + \xi N^k\right].
\end{aligned} \tag{B.19}$$

In particular, if we set $N^{k+1}\varepsilon = N^{-s+k}$ and $N \geq 3$, then we find that

$$\left\| f - \sum_{j=1}^{N} p_j^N \cdot \Phi_j^N \right\|_{C^k([0,T],L^q(\mu))} \leq C \ln^k(N) \left[\frac{\|f\|_{C^k(I_i^N, L^q(\mu))} + \mathcal{C}_k}{N^{s-k}} + \xi N^k \right]. \tag{B.20}$$

□

Lemma B.6. *Let $\mathcal{G}_\theta : \mathcal{X} \to \mathcal{H}$ be a tanh FNO with grid size $N \in \mathbb{N}$ and let $B > 0$. For every $\varepsilon > 0$, there exists a tanh DeepONet $\mathcal{G}_\theta^* : \mathcal{X} \to \mathcal{H}$ with N^d sensors and N^d branch and trunk nets such that*

$$\sup_{\|v\|_{L^\infty} \leq B} \sup_{x \in \mathbb{T}^d} \left| \mathcal{G}_\theta^*(v)(x) - \mathcal{G}_\theta(v)(x) \right| \leq \varepsilon. \tag{B.21}$$

Furthermore, $\mathrm{width}(\boldsymbol{\beta}) \sim N^d$, $\mathrm{depth}(\boldsymbol{\beta}) \sim \ln(N)$, $\mathrm{width}(\boldsymbol{\tau}) \sim N^d(N + \ln(N/\varepsilon))$ *and* $\mathrm{depth}(\boldsymbol{\tau}) = 3$.

Proof. This is a consequence of [38, Theorem 36] and Lemma D.1 with $\varepsilon \leftarrow N^d\varepsilon$. □

B.2 Proof of Theorem 3.5

Proof. **Step 1: construction.** To define the approximation, we divide $[0,T]$ into M subintervals of the form $[t_{m-1}, t_m]$, where $t_m = mT/M$ with $1 \leq m \leq M$. One could approximate u on every subinterval by an s-th order accurate Taylor approximation around t_m, provided that one has access to $D_t^i u(\cdot, t_m)$ for $0 \leq i \leq s-1$. As those values are unknown, we resort to the finite difference approximation $D_t^i u(\cdot, t_m) \approx M^i \cdot \Delta_{1/M,t}^{i,s-i}[\mathcal{U}^\varepsilon(u_0, t_m)]$, which is a neural network. See **SM** A.2 for an overview of the notation for finite difference operators. Moreover, we replace the univariate monomials $\varphi_i : [0,T] \to \mathbb{R} : t \mapsto t^i$ in the Taylor approximation by neural networks $\widehat{\varphi}_i^\delta : [0,T] \to \mathbb{R}$ with $\|\varphi_i - \widehat{\varphi}_i^\delta\|_{C^{k+1}} \lesssim \delta$. Lemma A.3 guarantees that the output of $(\widehat{\varphi}_i^\delta)_{i=1}^{s-1}$ can be obtained using a shallow network with width $2(s+1)$ (independent of δ). The multiplication operator is replaced by a shallow neural network $\widehat{\times}_\delta : [-a,a]^2 \to \mathbb{R}$ (for suitable $a > 0$) for which $\| \times - \widehat{\times}_\delta \|_{C^{k+1}} \lesssim \delta$. By Lemma A.4 only four neurons are needed for this network. This results in the following approximation for $f \in C^0([0,T] \times D)$,

$$\widehat{N}_m^\delta[f](t,x) := \sum_{i=0}^{s-1} \widehat{\times}_\delta \left(\frac{\Delta_{1/M,t}^{i,s-i}[f](t_m, x)}{M^{-i} i!}, \widehat{\varphi}_i^\delta(t - t_m) \right) \quad \forall t \in [0,T],\ x \in D,\ 1 \leq m \leq M. \tag{B.22}$$

Next, we patch together these individual approximations by (approximately) multiplying them with a NN approximation of a partition of unity, denoted by $\Phi_1^M, \ldots, \Phi_M^M : [0,T] \to [0,1]$, as introduced in Definition B.4 in **SM** B. Every Φ_m^M can be thought of as a NN approximation of the indicator function on $[t_{m-1}, t_m]$. For any $\varepsilon, \delta > 0$, we then define our final neural network approximation $\widehat{u} : [0,T] \times D \to \mathbb{R}$ as,

$$\widehat{u}(t,x) := \sum_{m=1}^{M} \widehat{\times}_\delta \left(\widehat{N}_m^\delta[\mathcal{U}^\varepsilon(u_0, t_m)](t,x), \Phi_m^M(t) \right) \quad \forall t \in [0,T],\ x \in D. \tag{B.23}$$

Step 2: error estimate. In order to facilitate the proof, we introduce the intermediate approximations $\widetilde{u} : [0,T] \times D \to \mathbb{R}$ and $N_m : C^0(D) \times [0,T] \times D \to \mathbb{R}$ by,

$$\widetilde{u}(t,x) := \sum_{m=1}^{M} N_m[\widehat{u}_m^\varepsilon](t,x) \cdot \Phi_m^M(t) := \sum_{m=1}^{M} \sum_{i=0}^{s-1} \frac{\Delta_{1/M,t}^{i,s-i}[\widehat{u}_m^\varepsilon](t_m, x)}{M^{-i} i!} \cdot \varphi_i(t - t_m) \cdot \Phi_m^M(t), \tag{B.24}$$

where $\widehat{u}_m^\varepsilon = \mathcal{U}^\varepsilon(u_0, t_m)$. Note that $\widehat{u}$ can be obtained from $\widetilde{u}$ by replacing the multiplication operator and the monomials by neural networks. Since these the size of these networks are independent of their accuracy δ, we can assume without loss of generality that $\|D^{(k,\boldsymbol{\alpha})}(\widetilde{u} - \widehat{u})(t,\cdot)\|_{L^q} \leq \delta$ (see Lemma B.2) for any relevant $D^{(k,\boldsymbol{\alpha})}$ and t.

It remains to prove that $D^{(k,\boldsymbol{\alpha})}\widetilde{u} \approx D^{(k,\boldsymbol{\alpha})}u$. Combining the observation that $D^{(k,\boldsymbol{\alpha})}N_m[\widehat{u}_m^\varepsilon] = D_t^k N_m[D_x^{\boldsymbol{\alpha}}\widehat{u}_m^\varepsilon]$ with Lemma B.3 lets us conclude that for all $0 \leq k \leq s-1$ and $t \in [t_{m-2}, t_{m+2}]$,

$$\left\|D^{(k,\boldsymbol{\alpha})}(N_m[\widehat{u}_m^\varepsilon](t,\cdot) - u(t,\cdot))\right\|_{L^q} \leq C(r)M^k(\left\|D_x^{\boldsymbol{\alpha}}(\widehat{u}_m^\varepsilon - u)(\cdot,t_m)\right\|_{L^q} + |u|_{C^{(s,\ell)}}M^{-s}) \tag{B.25}$$

We use Theorem B.5 with $f \leftarrow u$, $p_j^N \leftarrow N_m[D_x^{\boldsymbol{\alpha}}\widehat{u}_m^\varepsilon]$, $\xi \leftarrow C(r)M^k\left\|D_x^{\boldsymbol{\alpha}}(\widehat{u}_m^\varepsilon - u)(\cdot,t_m)\right\|_{L^q}$, $\mathcal{C}_\ell^* \leftarrow C(s)|u|_{C^{(k,\ell)}}$, $N \leftarrow M$ to find that,

$$\left\|D^{(k,\boldsymbol{\alpha})}(\widehat{u} - u)\right\|_{L^q} \leq C\ln^k(M)(\|u\|_{C^{(s,\ell)}}M^{k-s} + M^{2k}\left\|D_x^{\boldsymbol{\alpha}}(\widehat{u}_m^\varepsilon - u)(\cdot,t_m)\right\|_{L^q}), \tag{B.26}$$

where $C(r,s) > 0$ only depend on r and s. Finally, using Lemma B.1 to bound $\left\|D_x^{\boldsymbol{\alpha}}(\widehat{u}_m^\varepsilon - u)(\cdot,t_m)\right\|_{L^q}$ and combining this with Assumption 3.1 proves (3.4).

Step 3: size estimate. The following holds,

$$\mathrm{depth}(\widehat{u}) \leq C\mathrm{depth}(\mathcal{U}^\varepsilon), \mathrm{width}(\widehat{u}) \leq CM\mathrm{width}(\mathcal{U}^\varepsilon). \tag{B.27}$$

□

B.3 Proof of Theorem 3.7

Proof. **Step 1: construction.** Let $N \in \mathbb{N}$, let $\mathcal{E}_N : C^0(T^d) \to \mathbb{R}^{|\mathcal{J}_N|}$ be an encoder and $\mathcal{Q}_N : \mathbb{R}^{|\mathcal{J}_N|} \to L_N^2$ be a trigonometric polynomial interpolation operator, cf. **SM** A.6. If we let $\widehat{\mathcal{G}} = \mathcal{U}^\varepsilon \circ \mathcal{Q}_N \circ \mathcal{E}_N$ then we can define an FNO $\mathcal{G}_\theta : L_N^2(\mathbb{T}^d) \to L_N^2(\mathbb{T}^d)$ as $\mathcal{G}_\theta(u_0)(x) = (\mathcal{Q}_N \circ \mathcal{E}_N \circ \widehat{\mathcal{G}})(u_0)(x)$.

Step 2: error estimate. We decompose the L^2-error of the FNO using the triangle inequality and the inequality $\|\mathcal{U}^\varepsilon - \widehat{\mathcal{G}}\|_{L^2} \leq C_{\mathrm{stab}}^\varepsilon\|u_0 - \mathcal{Q}_N \circ \mathcal{E}_N \circ u_0\|_{L^p}$, which follows from Assumption 3.6,

$$\|\mathcal{G} - \mathcal{G}_\theta\|_{L^2} \leq \|\mathcal{G} - \mathcal{U}^\varepsilon\|_{L^2} + C_{\mathrm{stab}}^\varepsilon\|u_0 - \mathcal{Q}_N \circ \mathcal{E}_N \circ u_0\|_{L^p} + \|\widehat{\mathcal{G}} - \mathcal{G}_\theta\|_{L^2}. \tag{B.28}$$

First, we find using a Sobolev embedding result (Lemma A.1) and Lemma A.2 that,

$$\left\|u_0 - (\mathcal{Q}_N \circ \mathcal{E}_N)(u_0)\right\|_{L^p} \leq \left\|u_0 - (\mathcal{Q}_N \circ \mathcal{E}_N)(u_0)\right\|_{H^{d/p^*}} \leq C(d,r)N^{-r+d/p^*}\|u_0\|_{H^r}, \tag{B.29}$$

where p^* is such that $1/p + 1/p^* = 1/2$. Next, we observe that for any $u_0 \in \mathcal{X}$ with $\|u_0\|_{C^r} \leq B$ that $\left\|(\mathcal{Q}_N \circ \mathcal{E}_N)(u_0)\right\|_{H^r(\mathbb{T}^d)} \leq CB =: \overline{B}$. Hence, by applying Lemma A.2 to the second and last term of (B.28) we find that,

$$\|\mathcal{G} - \mathcal{G}_\theta\|_{L^2} \leq C(\varepsilon + C_{\mathrm{stab}}^\varepsilon BN^{-r+d/p^*} + C_{\varepsilon,r}^{\overline{B}}N^{-r}). \tag{B.30}$$

Step 3: size estimate. As for any FNO, the width is equal to $N^d\mathrm{width}(\mathcal{U}^\varepsilon)$. The depth in this case is equal to $\mathrm{depth}(\mathcal{U}^\varepsilon)$.

□

B.4 Proof of Theorem 3.10

Proof. **Step 1: construction.**

Let $\varepsilon > 0$ and $n, N \in \mathbb{N}$. We first introduce some notation. Let $\mathcal{J}_N = \{0,\ldots,2N+1\}^d$, $\mathcal{K}_N = \{-N,\ldots,N\}^d$, let $\{\mathbf{e}_j\}_{j\in\mathbb{N}}$ be an ordered Fourier basis, as described in **SM** A.5, and let $\{\widehat{\mathbf{e}}_j\}_{j\in\mathbb{N}}$ be a neural network approximation of the same basis such that

$$\max_{k\in\mathcal{K}_N}\|\mathbf{e}_k - \widehat{\mathbf{e}}_k\|_{C^r} \leq \eta, \tag{B.31}$$

cf. Lemma D.1. Using notation from **SM** A.6, let $\mathcal{Q}_N : \mathbb{R}^{|\mathcal{J}_N|} \to C(\mathbb{T}^d)$ be the trigonometric polynomial interpolation operator as in (A.18) and let $\mathcal{E}_N : C(\mathbb{T}^d) \to \mathbb{R}^{|\mathcal{J}_N|}$ be the encoder as in (A.20). We define

$$\widehat{\mathcal{Q}}_N : \mathbb{R}^{|\mathcal{J}_N|} \to C(\mathbb{T}^d) : y \mapsto \frac{1}{|\mathcal{K}_N|}\sum_{k\in\mathcal{K}_N}\sum_{j\in\mathcal{J}_N} y_j a_{k,j}\widehat{\mathbf{e}}_k, \tag{B.32}$$

with coefficients $a_{k,j}$ as in (A.19), as a neural network approximation of $\mathcal{Q}_N$.

Inspired by the proof of Theorem 3.5 (and using its notation as well), we define $\widehat{\mathcal{G}} : C(\mathbb{T}^d) \to L^2(\mu)$ by

$$\widehat{\mathcal{G}}(u_0)(t,x) = \sum_{m=1}^{M} \sum_{i=0}^{s-1} \frac{\Delta_{1/M}^{i,s-i}[\mathcal{U}^\varepsilon(\mathcal{Q}_Z \circ \mathcal{E}_Z \circ u_0, t_m)](t_m, x)}{M^{-i} i!} \cdot \varphi_i^\delta(t - t_m)\Phi_m^M(t), \tag{B.33}$$

Then it holds that

$$\begin{aligned}(\mathcal{Q}_N \circ \mathcal{E}_N \circ \widehat{\mathcal{G}})(u_0)(t,x) &= \sum_{k \in \mathcal{K}_N} \sum_{j \in \mathcal{J}_N} \sum_{m=1}^{M} \sum_{i=0}^{s-1} \frac{a_{k,j}}{|\mathcal{K}_N|} \frac{\Delta_{1/M}^{i,s-i}[\mathcal{U}^\varepsilon(\mathcal{Q}_Z \circ \mathcal{E}_Z \circ u_0, t_m)](t_m, x_j)}{M^{-i} i!} \cdot \Psi_{i,m,k}(t,x) \\ \Psi_{i,m,k}(t,x) &= \varphi_i^\delta(t - t_m)\Phi_m^M(t)\mathbf{e}_k(x).\end{aligned} \tag{B.34}$$

Now for every i, m, k let $\Psi_{i,m,k} : [0,T] \times \mathbb{T}^d \to \mathbb{R}$ be defined as,

$$\widehat{\Psi}_{i,m,k}(t,x) = \widehat{\times}_\delta \left(\varphi_i^\delta(t - t_m), \Phi_m^M(t), \widehat{\mathbf{e}}_k(x) \right), \tag{B.35}$$

where $\widehat{\times}_\delta$ is a neural network approximation of the multiplication operator. We can then construct a DeepONet as

$$\begin{aligned}\mathcal{G}_\theta(u_0)(t,x) &= \sum_{j=1}^{p} \beta_j(u_0)\tau_j(t,x) \\ &= \sum_{i=0}^{s-1} \sum_{m=1}^{M} \sum_{k \in \mathcal{K}_N} \left[\sum_{j \in \mathcal{J}_N} \frac{a_{k,j}}{|\mathcal{K}_N|} \frac{\Delta_{1/M}^{i,s-i}[\mathcal{U}^\varepsilon(\mathcal{Q}_Z \circ \mathcal{E}_Z \circ u_0, t_m)](t_m, x_j)}{M^{-i} i!} \right] \cdot \widehat{\Psi}_{i,m,k}(t,x).\end{aligned} \tag{B.36}$$

We see that we need to set $p = sM(2N+1)^d$ and that the trunk nets are given by $\tau_j \sim \widehat{\Psi}_{i,m,k}$, up to a different indexing.

Step 2: error estimate. First we use Assumption 3.4 to see that

$$\left\| \mathcal{L}(\mathcal{G} - \mathcal{G}_\theta) \right\|_{L^2} \leq C \sum_{k,\boldsymbol{\alpha}} \left\| D^{(k,\boldsymbol{\alpha})}(\mathcal{G} - \mathcal{G}_\theta) \right\|_{L^2}. \tag{B.37}$$

Next, we observe that using Assumption 3.1, Assumption 3.6 and (B.29) it holds that for all t,

$$\left\| (\mathcal{U}^\varepsilon(\mathcal{Q}_Z \circ \mathcal{E}_Z \circ u_0) - \mathcal{G}(u_0))(\cdot, t) \right\|_{L^2} \leq \varepsilon + C_{\text{stab}}^\varepsilon CBZ^{-r+d/p^*}. \tag{B.38}$$

One can then use Theorem 3.5, but by replacing ε by (B.38) in the error bound (3.4), to find that

$$\left\| D^{(k,\boldsymbol{\alpha})}(\mathcal{G} - \widehat{\mathcal{G}}) \right\|_{L^2} \leq C \ln^k(M)(\|u\|_{C^{(s,\ell)}} M^{k-s} + M^{2k}((\varepsilon + C_{\text{stab}}^\varepsilon Z^{-r+d/p^*})h^{-\ell} + C_{\varepsilon,\ell}^{CB} h^{r-\ell})) \tag{B.39}$$

Then, using the observation that $D^{(k,\boldsymbol{\alpha})}(\text{Id} - \mathcal{Q}_N \circ \mathcal{E}_N)\widehat{\mathcal{G}} = D_x^{\boldsymbol{\alpha}}(\text{Id} - \mathcal{Q}_N \circ \mathcal{E}_N) D_t^k \widehat{\mathcal{G}}$ we find that

$$\left\| D^{(k,\boldsymbol{\alpha})}(\text{Id} - \mathcal{Q}_N \circ \mathcal{E}_N)\widehat{\mathcal{G}}(u_0) \right\|_{L^2} \leq CN^{-(r-\ell)} \left\| D_t^k \widehat{\mathcal{G}}(u_0) \right\|_{H^r}, \tag{B.40}$$

which can be combined with the estimate

$$\left\| D_t^k \widehat{\mathcal{G}}(u_0) \right\|_{H^r} \leq M^{s-1} \cdot M^k \ln^k(M) \left\| \mathcal{U}^\varepsilon(\mathcal{Q}_Z \circ \mathcal{E}_Z \circ u_0) \right\|_{H^r} \leq M^{s+k-1} \ln^k(M) C_{\varepsilon,r}^{\overline{B}}, \tag{B.41}$$

where we used that for $u_0 \in \mathcal{X}$ with $\|u_0\|_{C^r} \leq B$ it holds $\left\| (\mathcal{Q}_N \circ \mathcal{E}_N)(u_0) \right\|_{H^r(\mathbb{T}^d)} \leq CB =: \overline{B}$. Next, we make the rough estimate that,

$$\left\| D^{(k,\boldsymbol{\alpha})}(\widehat{\mathcal{Q}}_N - \mathcal{Q}_N) \circ \mathcal{E}_N)\widehat{\mathcal{G}}(u_0) \right\|_{L^2} \leq CN^d M^{s+k-1} \ln^k(M) \max_k \|\mathbf{e}_k - \widehat{\mathbf{e}}_k\|_{C^r}. \tag{B.42}$$

Finally, using Lemma B.2 we find that

$$\left\| D^{(k,\boldsymbol{\alpha})}(\widehat{\mathcal{Q}}_N \circ \mathcal{E}_N \circ \widehat{\mathcal{G}} - \mathcal{G}_\theta)(u_0) \right\|_{L^2} \leq \delta. \tag{B.43}$$

By setting $\eta = N^{\ell-r-d}$, $h = 1/N$ and using that $M^{2k} \leq M^{k+s-1}$ and $C^{\overline{B}}_{\varepsilon,\ell} \leq C^{\overline{B}}_{\varepsilon,r}$ we find,

$$\left\|\mathcal{L}(\mathcal{G} - \mathcal{G}_\theta)\right\|_{L^2} \leq C \ln^k(M)(\|u\|_{C^{(s,\ell)}} M^{k-s} + M^{k+s-1}((\varepsilon + C^{\varepsilon}_{\text{stab}} Z^{-r+d/p^*})N^\ell + C^{\overline{B}}_{\varepsilon,r} N^{\ell-r})). \tag{B.44}$$

We conclude by using that $\ln^k(M) \leq CM^\rho$ for any $\rho > 0$.

Step 3: size estimate. It follows immediately that $\text{depth}(\boldsymbol{\beta}) = \text{depth}(\mathcal{U}^\varepsilon)$, $\text{width}(\boldsymbol{\beta}) = \mathcal{O}(M(Z^d + N^d \text{width}(\mathcal{U}^\varepsilon)))$, $\text{depth}(\boldsymbol{\tau}) = 3$ and $\text{width}(\boldsymbol{\tau}) = \mathcal{O}(MN^d(N + \ln(N)))$.

□

B.5 Proof of Theorem 3.11

Proof. Define the random variable $Y = \mathcal{E}_G(\theta^*(\mathcal{S}))^2 - \mathcal{E}_T(\theta^*(\mathcal{S}), \mathcal{S})^2$. Then if follows from equation (4.8) in the proof of [16, Theorem 5] that

$$\mathbb{P}(Y > \varepsilon^2) \leq \left(\frac{2R\mathfrak{L}}{\varepsilon^2}\right)^{d_\Theta} \exp\left(\frac{-2\varepsilon^4 n}{c^2}\right), \tag{B.45}$$

since $\mathbb{P}(Y > \varepsilon^2) = 1 - \mathbb{P}(\mathcal{A})$, where $\mathcal{A}$ is as defined in the proof of [16, Theorem 5]. It follows that

$$\mathbb{E}[Y] = \mathbb{E}[Y \mathbb{1}_{Y \leq \varepsilon^2}] + \mathbb{E}[Y \mathbb{1}_{Y > \varepsilon^2}] \leq \varepsilon^2 + c\mathbb{P}\left(Y > \varepsilon^2\right). \tag{B.46}$$

Setting $\varepsilon^2 = c\mathbb{P}\left(Y > \varepsilon^2\right)$ leads to

$$\mathbb{E}[Y] \leq 2\varepsilon^2 = \sqrt{\frac{2c^2}{n} \ln\left(\frac{c}{\varepsilon^2}\left(\frac{2R\mathfrak{L}}{\varepsilon^2}\right)^{d_\Theta}\right)}. \tag{B.47}$$

For $\varepsilon < 1$, and using that $\ln(x) \leq \sqrt{x}$ for all $x > 0$, this equality implies that

$$\varepsilon^{d_\Theta + 1} \leq \frac{2c^3(2R\mathfrak{L})^{d_\Theta/2}}{n}. \tag{B.48}$$

Hence, we find that if $n \geq 2c^2 e^8/(2R\mathfrak{L})^{d_\Theta/2}$ then $\varepsilon^{d_\Theta+1} \leq ce^{-8}(2R\mathfrak{L})^{d_\Theta}$ which implies that

$$\left[\ln\left(\frac{c}{\varepsilon^2}\left(\frac{2R\mathfrak{L}}{\varepsilon^2}\right)^{d_\Theta}\right)\right]^{-1/2} \leq \frac{1}{2\sqrt{2}}. \tag{B.49}$$

Using once more that $\varepsilon^2 = c\mathbb{P}\left(Y > \varepsilon^2\right)$ and (B.49) gives us,

$$\begin{aligned}\mathbb{E}[Y] &\leq \sqrt{\frac{2c^2}{n} \ln\left(c(2R\mathfrak{L})^{d_\Theta}\left(\frac{\sqrt{2n}}{c}\left[\ln\left(\frac{c}{\varepsilon^2}\left(\frac{2R\mathfrak{L}}{\varepsilon^2}\right)^{d_\Theta}\right)\right]^{-1/2}\right)^{d_\Theta+1}\right)} \\ &\leq \sqrt{\frac{2c^2}{n}\ln\big((a\mathfrak{L}\sqrt{n})^{d_\Theta+1}\big)} = \sqrt{\frac{2c^2(d_\Theta+1)}{n}\ln\big(a\mathfrak{L}\sqrt{n}\big)}.\end{aligned} \tag{B.50}$$

□

C Additional material for Section 4.1

C.1 Auxiliary results

Lemma C.1. *Let $\varepsilon > 0$, let $(\Omega, \mathcal{F}, \mathcal{P})$ be a probability space, and let $X : \Omega \to \mathbb{R}$ be a random variable that satisfies $\mathbb{E}\left[|X|\right] \leq \varepsilon$. Then it holds that $\mathbb{P}(|X| \leq \varepsilon) > 0$.*

Proof. This result is [23, Proposition 3.3]. □

Lemma C.2. *Let $\gamma \in \{0,1\}$, $\beta \in [1,\infty)$, $\alpha_0, \alpha_1, x_0, x_1, x_2, \ldots \in [0,\infty)$ satisfy for all $k \in \mathbb{N}_0$ that*

$$x_k \leq \mathbb{1}_{\mathbb{N}}(k)(\alpha_0 + \alpha_1 k)\beta^k + \sum_{l=0}^{k-1} (k-l)^{\gamma} \beta^{(k-l)} \left[x_l + x_{\max\{l-1,0\}} \right]. \tag{C.1}$$

Then it holds for all $k \in \mathbb{N}_0$ that

$$x_k \leq \frac{(\alpha_0 + \alpha_1)\beta^k \mathbb{1}_{\mathbb{N}}(k)}{(4+\gamma)^{1/2}(1+2^{(1+\gamma)/2})^{-k}} = \begin{cases} \mathbb{1}_{\mathbb{N}}(k)(\alpha_0+\alpha_1)2^{-1}(1+2^{1/2})^k\beta^k & : \gamma = 0 \\ \mathbb{1}_{\mathbb{N}}(k)(\alpha_0+\alpha_1)5^{-1/2}(3\beta)^k & : \gamma = 1. \end{cases} \tag{C.2}$$

Proof. This result is [33, Corollary 4.3]. □

Lemma C.3. *Let $\alpha \in [1,\infty)$, $x_0, x_1, \ldots \in [0,\infty)$ satisfy for all $k \in \mathbb{N}_0$ that $x_k \leq \alpha x_{k-1}^k$. Then it holds for all $k \in \mathbb{N}_0$ that*

$$x_k \leq \alpha^{(k+1)!} x_0^{k!} \tag{C.3}$$

Proof. We provide a proof by induction. First of all, it is clear that $x_0 \leq \alpha x_0$. For the induction step, assume that $x_{k-1} \leq \alpha^{k!} x_0^{(k-1)!}$ for an arbitrary $k \in \mathbb{N}_0$. We calculate that

$$x_k \leq \alpha \left(\alpha^{k!} x_0^{(k-1)!} \right)^k \leq \alpha^{(k+1)!} x_0^{k!}. \tag{C.4}$$

This proves the statement. □

Lemma C.4. *Let $\ell \in \mathbb{N}$, $f \in C^{\ell}(\mathbb{R},\mathbb{R})$, $h \in C^{\ell}(\mathbb{T}^d,\mathbb{R})$ and let B_ℓ denote the ℓ-th Bell number. Then it holds that*

$$|f \circ h|_{C^{\ell}(\mathbb{R})} \leq \|f\|_{C^{\ell}(\mathbb{R})} \left(B_\ell \|h\|_{C^{\ell-1}(\mathbb{T}^d)}^{\ell} + |h|_{C^{\ell}(\mathbb{T}^d)} \right). \tag{C.5}$$

Proof. Let Π be the set of all partitions of the set $\{1,\ldots,\ell\}$, let $\alpha \in \mathbb{N}_0^d$ such that $\|\alpha\|_1 = \ell$ and let $\iota : \mathbb{N}^{\ell} \to \mathbb{N}^d$ be a map such that $D^{\alpha} = \frac{\partial^{\ell}}{\prod_{j=1}^{\ell} x_{\iota(j)}}$. Then the Faà di Bruno formula can be reformulated as [12],

$$\begin{aligned} D^{\alpha} f(h(x)) &= \sum_{\pi \in \Pi} f^{(|\pi|)}(h(x)) \cdot \prod_{B \in \pi} \frac{\partial^{|B|} h(x)}{\prod_{j \in B} \partial x_{\iota(j)}} \\ &= \sum_{\substack{\pi \in \Pi, \\ |\pi| \geq 2}} f^{(|\pi|)}(h(x)) \cdot \prod_{B \in \pi} \frac{\partial^{|B|} h(x)}{\prod_{j \in B} \partial x_{\iota(j)}} + f'(h(x)) D^{\alpha} h(x). \end{aligned} \tag{C.6}$$

Combining this formula with the definition of the Bell number as $B_\ell = |\Pi|$, we find the following upper bound,

$$\begin{aligned} |f \circ h|_{C^{\ell}(\mathbb{R})} &\leq \sum_{\pi \in \Pi} \|f\|_{C^{\ell}(\mathbb{R})} \|h\|_{C^{\ell-1}(\mathbb{R})}^{\ell} + \|f\|_{C^1(\mathbb{R})} |h|_{C^{\ell}(\mathbb{R})} \\ &\leq \|f\|_{C^{\ell}(\mathbb{R})} \left(B_\ell \|h\|_{C^{\ell-1}(\mathbb{R})}^{\ell} + |h|_{C^{\ell}(\mathbb{R})} \right). \end{aligned} \tag{C.7}$$

□

C.2 Proof of Theorem 4.2

Definition C.5. *Let $(\Omega, \mathcal{F}, \mu)$ be a measure space and let $q > 0$. For every $\mathcal{F}/\mathcal{B}(\mathbb{R}^d)$-measurable function $f : \Omega \to \mathbb{R}^d$, we define*

$$\|f\|_{\mathcal{L}^q(\mu, \|\cdot\|_{\mathbb{R}^d})} := \left[\int_{\Omega} \|f(\omega)\|_{\mathbb{R}^d}^q \mu(d\omega) \right]^{1/q}. \tag{C.8}$$

Let $(\Omega, \mathcal{F}, P, (\mathbb{F}_t)_{t\in[0,T]})$ be a stochastic basis, $D \subseteq \mathbb{R}^d$ a compact set and, for every $x \in D$, let $X^x : \Omega \times [0,T] \to \mathbb{R}^d$ be the solution, in the Itô sense, of the following stochastic differential equation,

$$dX_t^x = \mu(X_t^x)dt + \sigma(X_t^x)dB_t, \quad X_0^x = x, \quad x \in D, t \in [0,T], \tag{C.9}$$

where B_t is a standard d-dimensional Brownian motion on $(\Omega, \mathcal{F}, P, (\mathbb{F}_t)_{t\in[0,T]})$. The existence of X^x is guaranteed by [3, Theorem 4.5.1].

As in [16, Theorem 3.3] we define ρ_d as

$$\rho_d := \max_{x\in D} \sup_{\substack{s,t\in[0,T],\\ s<t}} \frac{\|X_s^x - X_t^x\|_{\mathcal{L}^q(P,\|\cdot\|_{\mathbb{R}^d})}}{|s-t|^{\frac{1}{p}}} < \infty, \tag{C.10}$$

where X^x is the solution, in the Itô sense, of the SDE (C.9), $q > 2$ is independent of d and $\|\cdot\|_{\mathcal{L}^q(P,\|\cdot\|_{\mathbb{R}^d})}$ is as in Definition C.5.

Lemma C.6. *In Setting 4.1, Assumption 3.1 and Assumption 3.6 are satisfied with*

$$\left\|u(\cdot,t) - \mathcal{U}^\varepsilon(\varphi,t)\right\|_{L^2(\mu)} \leq \varepsilon, \qquad C^B_{\varepsilon,\ell} = CB \cdot \mathrm{poly}(d\rho_d), \qquad C^\varepsilon_{\mathrm{stab}} = 1, \qquad p = \infty, \tag{C.11}$$

where $t \in [0,T]$ and $\varphi \in C_0^2(\mathbb{R}^d)$. Moreover, there exists $C^* > 0$ (independent of d) for which it holds that $\mathrm{depth}(\mathcal{U}^\varepsilon) \leq C^* \mathrm{depth}(\widehat{\varphi}^\varepsilon)$ *and* $\{\mathrm{width}, \mathrm{size}\}(\mathcal{U}^\varepsilon) \leq C^* \varepsilon^{-2} \{\mathrm{width}, \mathrm{size}\}(\widehat{\varphi}^\varepsilon)$.

Proof. It follows from the Feynman-Kac formula that $u(t,x) = \mathbb{E}\left[\varphi(X_t^x)\right]$ [62]. Replacing φ by a neural network $\widehat{\varphi}^\varepsilon$ with $\|\varphi - \widehat{\varphi}^\varepsilon\|_{C^0} \leq \varepsilon$ gives us for any probability measure μ that,

$$\left\|\mathbb{E}\left[\varphi(X_t^x)\right] - \mathbb{E}\left[\widehat{\varphi}^\varepsilon(X_t^x)\right]\right\|_{L^2(\mu)} \leq \|\varphi - \widehat{\varphi}^\varepsilon\|_{C^0}. \tag{C.12}$$

Using [16, Lemma A.2] (which is based on [23]) we find,

$$\mathbb{E}\left[(I)\right] := \mathbb{E}\left[\left(\int_D \left|\mathbb{E}\left[\widehat{\varphi}^\varepsilon(X_t^x)\right] - \frac{1}{m}\sum_{i=1}^m \widehat{\varphi}^\varepsilon(X_t^x(\omega_m))\right|^2 \mu(dx)\right)^{1/2}\right] \leq \frac{2\|\widehat{\varphi}^\varepsilon\|_{C^0}}{\sqrt{m}}. \tag{C.13}$$

From [16, Lemma A.5], for all $x \in \mathbb{R}^d$, $t \in [0,T]$ and $\omega \in \Omega$ it holds that

$$X_t^x(\omega) = \sum_{i=1}^d \left(X_t^{e_i}(\omega) - X_t^0(\omega)\right) x_i + X_t^0(\omega). \tag{C.14}$$

Using this equality, together with Hölder's inequality and the boundedness of $\|X_t^x\|_{L^p}$ [16, Lemma A.5] we find that,

$$\begin{aligned}
\mathbb{E}\left[(II_{\boldsymbol{\alpha}})\right] &:= \mathbb{E}\left[\left(\int_D \left|\mathbb{E}\left[D_x^{\boldsymbol{\alpha}}\widehat{\varphi}^\varepsilon(X_t^x)\right] - \frac{1}{m}\sum_{i=1}^m D_x^{\boldsymbol{\alpha}}\widehat{\varphi}^\varepsilon(X_t^x(\omega_m))\right|^2 \mu(dx)\right)^{1/2}\right] \\
&\leq C \cdot \mathrm{poly}(d\rho_d) \cdot \mathbb{E}\left[\left(\int_D \left|\mathbb{E}\left[\widehat{\varphi}^\varepsilon(X_t^x)\right] - \frac{1}{m}\sum_{i=1}^m \widehat{\varphi}^\varepsilon(X_t^x(\omega_m))\right|^4 \mu(dx)\right)^{1/4}\right] \\
&\leq CB \cdot \mathrm{poly}(d\rho_d)
\end{aligned} \tag{C.15}$$

Combining the previous results gives us,

$$\mathbb{E}\left[\sqrt{m} \cdot (I) + \sum_{\|\boldsymbol{\alpha}\|_1 \leq \ell} (II_{\boldsymbol{\alpha}})\right] \leq CB \cdot \mathrm{poly}(d\rho_d). \tag{C.16}$$

If we combine this with Lemma C.1 then we find the existence of $(\omega_i^*)_{i=1}^m$ such that for

$$\mathcal{U}^\varepsilon(\varphi,t)(x) = \frac{1}{m}\sum_{i=1}^m \widehat{\varphi}^\varepsilon\left(\sum_{i=1}^d \left(X_t^{e_i}(\omega_i^*) - X_t^0(\omega_i^*)\right)x_i + X_t^0(\omega_i^*)\right) \tag{C.17}$$

it holds that

$$\left\|\mathbb{E}\left[\widehat{\varphi}^\varepsilon(X_t^x)\right] - \mathcal{U}^\varepsilon(\varphi,t)\right\|_{L^2(D)} \leq \frac{2C}{\sqrt{m}}. \tag{C.18}$$

and by setting $m = \varepsilon^{-2}$ and (C.12) we find that,

$$\left\|u(\cdot,t) - \mathcal{U}^\varepsilon(\varphi,t)\right\|_{L^2(D)} \leq \varepsilon, \qquad C^B_{\varepsilon,\ell} = CB\cdot \mathrm{poly}(\rho_d), \qquad C^\varepsilon_{\mathrm{stab}} = 1, \qquad p=\infty. \tag{C.19}$$

Moreover, it holds that

$$\mathrm{depth}(\mathcal{U}^\varepsilon) \leq C^*\mathrm{depth}(\widehat{\varphi}^\varepsilon), \qquad \{\mathrm{width},\mathrm{size}\}(\mathcal{U}^\varepsilon) \leq C^*\varepsilon^{-2}\{\mathrm{width},\mathrm{size}\}(\widehat{\varphi}^\varepsilon) \tag{C.20}$$

where we write $C^* = C\mathrm{poly}(d\rho_d)$. □

We can now present the proof of the actual theorem.

Proof of Theorem 4.2. We use Theorem 3.5 with $k=1$ and $\ell=2$ and combine the result with Lemma C.6. We find that for every $M\in\mathbb{N}$ and $\delta,h>0$ it holds that,

$$\begin{aligned}&\left\|\mathcal{L}(\widehat{u}-u)\right\|_{L^q([0,T]\times D)} + \|\widehat{u}-u\|_{L^2(\partial([0,T]\times D))}\\ &\quad\leq CB\cdot\mathrm{poly}(d\rho_d)\cdot\ln(M)(\|u\|_{C^{(1,2)}}M^{1-s} + M^2(\delta h^{-2} + h^{r-2})).\end{aligned} \tag{C.21}$$

Set $\delta = h^r$, $M^{-1-s} = \delta^{1-2/r}$ we find that

$$\left\|\mathcal{L}(\widehat{u}-u)\right\|_{L^q([0,T]\times D)} + \|\widehat{u}-u\|_{L^2(\partial([0,T]\times D))} \leq CB\cdot\mathrm{poly}(d\rho_d)\ln\big(1/\delta\big)\delta^{\frac{r-2}{r}\frac{s-1}{s+1}} \tag{C.22}$$

Using that $\ln(M)\leq CM^\sigma$ for arbitrarily small $\sigma>0$, we find that we should set

$$\delta = \varepsilon^{\frac{r+\sigma}{r-2}\frac{s+1}{s-1}}, \qquad M = \varepsilon^{\frac{-1-\sigma}{s-1}} \tag{C.23}$$

□

C.3 Nonlinear parabolic equations

Some examples of nonlinear parabolic PDEs of the type (4.3) are:

- The *Kolmogorov–Petrovsky–Piskunov (KPP) equation* [37] is a celebrated model that is often used to model wave propagation and population genetics. The model is particularly useful for systems that exhibit phase transitions. One obtains the KPP equation if one chooses a sufficiently smooth nonlinearity F that satisfies the requirements $F(0)=F(1)=0$, $F'(0)=r>0$, $F(u)>0$ and $F'(u)<r$ for all $0<u<1$. Well-known examples include the *Fisher equation* [21] with $F(u)=ru(1-u)$ and the *Allen-Cahn equation* [1] with $F(u)=ru(1-u^2)$.
- *Branching diffusion processes* give a probabilistic representation of the KPP equation for the case where $F(u)=\beta(\sum_{k=0}^\infty a_k u^k - u)$ with $a_k\geq 0$ and $\sum_k a_k = 1$. In this setting, the PDE (4.3) describes a d-dimensional branching Brownian motion, where every particle in the system dies in an an exponential time of parameter β and created k i.i.d. descendants with probability a_k [29, 58].
- Finally, the PDE (4.3) arises in the context of credit valuation adjustment when pricing derivative contracts to compute the counterparty risk valuation, e.g. [28]. The dimension d corresponds to the number of underlying assets and can be very high.

C.4 Multilevel Picard approximations

In what follows, we will provide a definition of a particular kind of MLP approximation (cf. [33]) and a theorem that quantifies the accuracy of the approximation. First, we rigorously introduce the setting of the nonlinear parabolic PDE (4.3) that is under consideration, cf. [33, Setting 3.2 with $p \leftarrow 0$]. We choose the d-dimensional torus $\mathbb{T}^d = [0, 2\pi)^d$ as domain and impose periodic boundary conditions. This setting allows us to use the results of [33], which are set in $\mathbb{R}^d$, and yet still consider a bounded domain so that the error can be quantified using an uniform probability measure.

Setting C.7. *Let $d, m \in \mathbb{N}$, $T, L, \mathcal{L} \in [0, \infty)$, let $(\mathbb{T}^d, \mathcal{B}(\mathbb{T}^d), \mu)$ be a probability space where μ is the rescaled Lebesgue measure, let $g \in C(\mathbb{T}^d, \mathbb{R}) \cap L^2(\mu)$, let $F \in C(\mathbb{R}, \mathbb{R})$, assume for all $x \in \mathbb{T}^d$, $y, z \in \mathbb{R}$ that*

$$\big|F(y) - F(z)\big| \leq L|y - z|, \qquad \max\{\big|F(y)\big|, \big|g(x)\big|\} \leq \mathcal{L}. \tag{C.24}$$

Let $u_d \in C^{1,2}([0, T] \times \mathbb{R}^d, \mathbb{R}) \cap L^2(\mu)$ satisfy for all $t \in [0, T], x \in \mathbb{R}^d$ that

$$(\partial_t u_d)(t, x) = (\Delta_x u_d)(t, x) + F(u_d(t, x)), \qquad u_d(0, x) = g(x). \tag{C.25}$$

Assume that for every $\varepsilon > 0$ there exists a neural network $\widehat{F}_\varepsilon$, a neural network $\widehat{g}_\varepsilon$ and a neural network $\mathcal{I}_\varepsilon$ with depth $\operatorname{depth}(\mathcal{I}_\varepsilon) = \operatorname{depth}(\widehat{F}_\varepsilon)$ such that

$$\left\|\widehat{F}_\varepsilon - F\right\|_{C^0(\mathbb{R})} \leq \varepsilon, \quad \|\widehat{g}_\varepsilon - g\|_{L^2(\mu)} \leq \varepsilon, \quad \|\mathcal{I}_\varepsilon - \mathrm{Id}\|_{C^0([-1-\mathcal{L}, 1+\mathcal{L}])} \leq \varepsilon. \tag{C.26}$$

Note that for some of the equations introduced in Section C.3 the nonlinearity F might not be globally Lipschitz and hence does not satisfy (C.24). However, it is easy to argue or rescale g [43, 5] such that u_d is globally bounded by some constant C. For instance, for the Allen-Cahn equation it holds that if $\|g\|_{L^\infty} \leq 1$ then $\big\|u_d(t, \cdot)\big\|_{L^\infty} \leq 1$ for any $t \in [0, T]$ [75]. One can then define a 'smooth', globally Lipschitz, bounded function $\widetilde{F} : \mathbb{R} \to \mathbb{R}$ such that $\widetilde{F}(v) = F(v)$ for $|v| \leq C$ and such that $\widetilde{F}(v) = 0$ for $|v| > 2C$. This will then also ensure the existence of a neural network $\widehat{F}$ that is close to $\widetilde{F}$ in $C^0(\mathbb{R})$-norm.

In this setting, multilevel Picard approximations can be introduced. We follow the definition of [33].

Definition C.8 (MLP approximation)**.** *Assume Setting C.7. Let $\Theta = \bigcup_{n \in \mathbb{N}} \mathbb{Z}^n$, let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space, let $Y^\theta \colon \Omega \to [0, 1]$, $\theta \in \Theta$, be i.i.d. random variables, assume for all $\theta \in \Theta$, $r \in (0, 1)$ that $\mathbb{P}(Y^\theta \leq r) = r$, let $\mathcal{U}^\theta \colon [0, T] \times \Omega \to [0, T]$, $\theta \in \Theta$, satisfy for all $t \in [0, T]$, $\theta \in \Theta$ that $\mathcal{U}^\theta_t = t + (T - t)Y^\theta$, let $W^\theta \colon [0, T] \times \Omega \to \mathbb{R}^d$, $\theta \in \Theta$, be independent standard Brownian motions, assume that $(\mathcal{U}^\theta)_{\theta \in \Theta}$ and $(W^\theta)_{\theta \in \Theta}$ are independent, and let $U^\theta_n \colon [0, T] \times \mathbb{T}^d \times \Omega \to \mathbb{R}$, $n \in \mathbb{Z}$, $\theta \in \Theta$, satisfy for all $n \in \mathbb{N}_0$, $\theta \in \Theta$, $t \in [0, T]$, $x \in \mathbb{T}^d$ that*

$$\begin{aligned} U^\theta_n(t, x) = {} & \frac{\mathbb{1}_{\mathbb{N}}(n)}{m^n} \left[\sum_{k=1}^{m^n} g(x + W^{(\theta,0,-k)}_{T-t})\right] \\ & + \sum_{i=0}^{n-1} \frac{(T-t)}{m^{n-i}} \left[\sum_{k=1}^{m^{n-i}} (F(U^{(\theta,i,k)}_i) - \mathbb{1}_{\mathbb{N}}(i) F(U^{(\theta,-i,k)}_{i-1}))(\mathcal{U}^{(\theta,i,k)}_t, x + W^{(\theta,i,k)}_{\mathcal{U}^{(\theta,i,k)}_t - t})\right]. \end{aligned} \tag{C.27}$$

Example C.9. *In order to improve the intuition of the reader regarding Definition C.8, we provide explicit formulas for the multilevel Picard approximation* (C.27) *for $n = 0$ and $n = 1$,*

$$U^\theta_0(t, x) = 0 \quad \textit{and} \quad U^\theta_1(t, x) = \frac{1}{m} \left[\sum_{k=1}^{m} g(x + W^{(\theta,0,-k)}_{T-t})\right] + (T - t)F(0). \tag{C.28}$$

Finally, we provide a result on the accuracy of MLP approximations at single space-time points.

Theorem C.10. *It holds for all $n \in \mathbb{N}_0$, $t \in [0, T]$, $x \in \mathbb{T}^d$ that*

$$\left(\mathbb{E}\left[\left|U^0_n(t, x) - u(t, x)\right|^2\right]\right)^{1/2} \leq \frac{\mathcal{L}(T + 1)\exp(LT)(1 + 2LT)^n}{m^{n/2}\exp\big(-m/2\big)}. \tag{C.29}$$

Proof. This result is [33, Corollary 3.15] with $p \leftarrow 0$, $\mathfrak{p} \leftarrow 2$ and $\mathcal{L} \leftarrow \mathcal{L}/2$. □

C.5 Neural network approximation of nonlinear parabolic equations

In this section, we will prove that the solution of the nonlinear parabolic PDE as in Setting C.7 can be approximated with a neural network without the curse of dimensionality. At this point, we do not specify the activation function, with the only restriction being that the considered neural networks should be expressive enough to satisfy (C.26). By emulating an MLP approximation and using that F, g and the identity function can be approximated using neural networks, the following theorem can be proven.

Theorem C.11. *Assume Setting C.7. For every $\varepsilon, \sigma > 0$ and $t \in [0, T]$ there exists a neural network $\widehat{u}_\varepsilon : \mathbb{T}^d \to \mathbb{R}$ such that*

$$\left\|\widehat{u}_\varepsilon(\cdot) - u(t,\cdot)\right\|_{L^2(\mu)} \leq \varepsilon. \tag{C.30}$$

In addition, $\widehat{u}$ satisfies that

$$\begin{aligned} &\mathrm{depth}(\widehat{u}_\varepsilon) \leq \mathrm{depth}(\widehat{g}_\delta) + \log_{C_2}(3C_1 \exp\big(m/2\big)/\varepsilon)\mathrm{depth}(\widehat{F}_\delta), \\ &\mathrm{width}(\widehat{u}_\varepsilon), \mathrm{size}(\widehat{u}_\varepsilon) \leq (\mathrm{size}(\widehat{g}_\delta) + \mathrm{size}(\widehat{F}_\delta) + \mathrm{size}(\mathcal{I}_\delta)) \left(\frac{4C_1 \exp\big(m/2\big)}{\varepsilon}\right)^{2+3\sigma}, \end{aligned} \tag{C.31}$$

where

$$\begin{aligned} &C_1 = (T+1)(1 + \mathcal{L}\exp(LT)), \qquad C_2 = 5 + 3LT, \\ &\delta = \frac{\varepsilon^2}{9C_1^2 \exp\big(m/2\big)}, \qquad m = C_2^{2(1+1/\sigma)}. \end{aligned} \tag{C.32}$$

Proof. **Step 1: construction of the neural network.** Let $\varepsilon, \delta > 0$ be arbitrary and let $\widehat{F} = \widehat{F}_\delta$, $\widehat{g} = \widehat{g}_\delta$ and $\mathcal{I} = \mathcal{I}_\delta$ as in Setting C.7. We then define for all $n \in \mathbb{N}$ and $\theta \in \Theta$,

$$\begin{aligned} \widehat{U}_n^\theta(t,x) = &\frac{\mathbb{1}_{\mathbb{N}}(n)}{m^n}\left[\sum_{k=1}^{m^n} (\mathcal{I}^{n-1} \circ \widehat{g})(x + W_{T-t}^{(\theta,0,-k)})\right] \\ &+ \sum_{i=0}^{n-1} \frac{(T-t)}{m^{n-i}}\left[\sum_{k=1}^{m^{n-i}} ((\mathcal{I}^{n-i-1} \circ \widehat{F})(\widehat{U}_i^{(\theta,i,k)}) - \mathbb{1}_{\mathbb{N}}(i)(\mathcal{I}^{n-i} \circ \widehat{F})(\widehat{U}_{i-1}^{(\theta,-i,k)}))(\mathcal{U}_t^{(\theta,i,k)}, x + W_{\mathcal{U}_t^{(\theta,i,k)}-t}^{(\theta,i,k)})\right], \end{aligned} \tag{C.33}$$

with notation and random variables cf. Definition C.8. Note that for every $t \in [0, T]$, $n \in \mathbb{N}$, $\theta \in \Theta$, every realization of the random variable $\widehat{U}_n^\theta(t,\cdot)$ is a neural network that maps from $\mathbb{T}^d$ to $\mathbb{R}$.

Let $n \in \mathbb{N}_0$, $m \in \mathbb{N}$ and $t \in [0, T]$ be arbitrary. Integrating the square of the error bound of Theorem C.10 and Fubini's theorem tell us that

$$\begin{aligned} \mathbb{E}\left[\int_{\mathbb{T}^d} \left|U_n^0(t,x) - u(t,x)\right|^2 d\mu(x)\right] &= \int_{\mathbb{T}^d} \mathbb{E}\left[\left|U_n^0(t,x) - u(t,x)\right|^2\right] d\mu(x) \\ &\leq \frac{4\mathcal{L}^2(T+1)^2 \exp(2LT)(1+2LT)^{2n}}{m^n \exp(-m)}. \end{aligned} \tag{C.34}$$

From Lemma C.1 it then follows that

$$\mathbb{P}\left(\int_{\mathbb{T}^d} \left|U_n^0(t,x) - u(t,x)\right|^2 d\mu(x) \leq \frac{4\mathcal{L}^2(T+1)^2 \exp(2LT)(1+2LT)^{2n}}{m^n \exp(-m)}\right) > 0. \tag{C.35}$$

As a result, there exists $\overline{\omega} = \overline{\omega}(t,n,m) \in \Omega$ and a realization $U_n^0(\overline{\omega})$ such that

$$\left\|U_n^0(\overline{\omega})(t,\cdot) - u(t,\cdot)\right\|_{L^2(\mu)} \leq \frac{\mathcal{L}(T+1)\exp(LT)(1+2LT)^n}{m^{n/2}\exp\big(-m/2\big)}. \tag{C.36}$$

We define

$$\omega : [0,T] \times \mathbb{N}^2 \to \Omega : (t,n,m) \mapsto \overline{\omega}(t,n,m) \tag{C.37}$$

and set for every $1 \leq k \leq n$,

$$\widehat{U}_{k,\omega}^\theta(t,x) = \widehat{U}_k^\theta(\omega(t,n,m))(t,x) \quad \text{and} \quad U_{k,\omega}^\theta(t,x) = U_k^\theta(\omega(t,n,m))(t,x) \tag{C.38}$$

for all $k \in \mathbb{N}_0$ and all $\theta \in \Theta$. We then define our approximation as $\widehat{U}^0_{n,\omega}(t,\cdot)$.

Step 2: error estimate. We will quantify how well $\widehat{U}^0_{n,\omega}$ approximates $U^0_{n,\omega}$. Using the calculation that for $f_1, f_2 \in C^1(\mathbb{R})$ and $h_1, h_2 \in L^2(\mu)$ it holds that

$$\begin{aligned}\|f_1 \circ h_1 - f_2 \circ h_2\|_{L^2(\mu)} &\leq \|f_1 \circ h_1 - f_2 \circ h_1\|_{L^2(\mu)} + \|f_2 \circ h_1 - f_2 \circ h_2\|_{L^2(\mu)} \\ &\leq \|f_1 - f_2\|_{C^0(\mathbb{R})} + |f_2|_{Lip(\mathbb{R})}\|h_1 - h_2\|_{L^2(\mu)},\end{aligned} \tag{C.39}$$

and the fact that $2n \leq 2^n$ for $n \in \mathbb{N}$ we find that it holds for every $\theta \in \Theta$ that,

$$\begin{aligned}&\left\|\widehat{U}^\theta_{n,\omega}(t,\cdot) - U^\theta_{n,\omega}(t,\cdot)\right\|_{L^2(\mu)} \\ &\leq \mathbb{1}_{\mathbb{N}}(n)\left(\|\widehat{g} - g\|_{L^2(\mu)} + (n-1)\|\mathcal{I} - \mathrm{Id}\|_{C^0}\right) + T\sum_{i=0}^{n-1}\left\|\mathcal{I}^{n-i-1} \circ \widehat{F} \circ \widehat{U}^{(\theta,i,k)}_{i,\omega} - F \circ U^{(\theta,i,k)}_{i,\omega}\right\|_{L^2(\mu)} \\ &\quad + T\sum_{i=0}^{n-1}\mathbb{1}_{\mathbb{N}}(i)\left\|\mathcal{I}^{n-i} \circ \widehat{F} \circ \widehat{U}^{(\theta,-i,k)}_{i-1,\omega} - F \circ U^{(\theta,-i,k)}_{i-1,\omega}\right\|_{L^2(\mu)} \\ &\leq \mathbb{1}_{\mathbb{N}}(n)\left[\|\widehat{g} - g\|_{L^2(\mu)} + 2Tn\left\|\widehat{F} - F\right\|_{C^0(\mathbb{R})} + \left((n-1) + \frac{(n-1)n}{2} + \frac{(n+1)n}{2}\right)\|\mathcal{I} - \mathrm{Id}\|_{C^0}\right] \quad , \\ &\quad + LT\sum_{i=0}^{n-1}\left(\left\|\widehat{U}^{(\theta,i,k)}_i - U^{(\theta,i,k)}_i\right\|_{L^2(\mu)} + \mathbb{1}_{\mathbb{N}}(i)\left\|\widehat{U}^{(\theta,-i,k)}_{i-1} - U^{(\theta,-i,k)}_{i-1}\right\|_{L^2(\mu)}\right) \\ &\leq \mathbb{1}_{\mathbb{N}}(n)2^n\left[\|\widehat{g} - g\|_{L^2(\mu)} + T\left\|\widehat{F} - F\right\|_{C^0(\mathbb{R})} + n\|\mathcal{I} - \mathrm{Id}\|_{C^0}\right] \\ &\quad + \sum_{i=0}^{n-1}(\max\{1, LT\})^{n-i}\left(\left\|\widehat{U}^{(\theta,i,k)}_i - U^{(\theta,i,k)}_i\right\|_{L^2(\mu)} + \mathbb{1}_{\mathbb{N}}(i)\left\|\widehat{U}^{(\theta,-i,k)}_{i-1} - U^{(\theta,-i,k)}_{i-1}\right\|_{L^2(\mu)}\right).\end{aligned} \tag{C.40}$$

Now let us set for every $k \in \mathbb{N}_0$,

$$x_k = \sup_{\theta\in\Theta}\left\|\widehat{U}^\theta_{k,\omega}(t,\cdot) - U^\theta_{k,\omega}(t,\cdot)\right\|_{L^2(\mu)}, \tag{C.41}$$

and in addition we define $\alpha_0 = \|\widehat{g} - g\|_{L^2(\mu)} + T\left\|\widehat{F} - F\right\|_{C^0(\mathbb{R})}$, $\alpha_1 = \|\mathcal{I} - \mathrm{Id}\|_{C^0}$ and $\beta = 2 + LT$. Taking the supremum over all $\theta \in \Theta$ in (C.40) gives us for all $k \in \mathbb{N}_0$ that,

$$x_k \leq \mathbb{1}_{\mathbb{N}}(k)(\alpha_0 + \alpha_1 k)\beta^k + \sum_{i=0}^{k-1}\beta^{k-i}(x_i + x_{\max\{i-1,0\}}). \tag{C.42}$$

Therefore, we can use Lemma C.2 with $\gamma \leftarrow 0$ then gives us that for all $k \in \mathbb{N}_0$ it holds that,

$$\begin{aligned}&\sup_{\theta\in\Theta}\left\|\widehat{U}^\theta_{k,\omega}(t,\cdot) - U^\theta_{k,\omega}(t,\cdot)\right\|_{L^2(\mu)} \\ &\quad \leq \mathbb{1}_{\mathbb{N}}(k)\frac{(1+\sqrt{2})^k}{2}\left(\|\widehat{g} - g\|_{L^2(\mu)} + T\left\|\widehat{F} - F\right\|_{C^0(\mathbb{R})} + \|\mathcal{I} - \mathrm{Id}\|_{C^0}\right)(2 + LT)^k.\end{aligned} \tag{C.43}$$

Next we define

$$C_1 = (T+1)(1 + \mathcal{L}\exp(LT)), \qquad C_2 = 5 + 3LT. \tag{C.44}$$

Combining (C.36) with (C.43) then gives us that,

$$\begin{aligned}&\left\|\widehat{U}^0_{n,\omega}(t,\cdot) - u(t,\cdot)\right\|_{L^2(\mu)} \\ &\quad \leq \left\|\widehat{U}^0_{n,\omega}(t,\cdot) - U^0_{n,\omega}(t,\cdot)\right\|_{L^2(\mu)} + \left\|U^0_{n,\omega}(t,\cdot) - u(t,\cdot)\right\|_{L^2(\mu)} \\ &\quad \leq C_1C_2^n\left(\|\widehat{g} - g\|_{L^2(\mu)} + \left\|\widehat{F} - F\right\|_{C^0(\mathbb{R})} + \|\mathcal{I} - \mathrm{Id}\|_{C^0} + m^{-n/2}\exp\big(m/2\big)\right).\end{aligned} \tag{C.45}$$

For an arbitrary $\sigma > 0$, we choose

$$m = C_2^{2(1+1/\sigma)}, \qquad n = \sigma \log_{C_2}(4C_1 \exp\big(m/2\big)/\varepsilon) \tag{C.46}$$

and if we choose $\widehat{g} = \widehat{g}_\delta$ and $\widehat{F} = \widehat{F}_\delta$ such that,

$$\|\widehat{g} - g\|_{L^2(\mu)} \leq \delta = \frac{\varepsilon}{4C_1 C_2^n} = \frac{\varepsilon^{1+\sigma}}{(4C_1)^{1+\sigma} \exp\big(\sigma m/2\big)}, \tag{C.47}$$

then we obtain that

$$\Big\| \widehat{U}^0_{n,\omega}(t,\cdot) - u(t,\cdot) \Big\|_{L^2(\mu)} \leq \varepsilon. \tag{C.48}$$

Step 3: size estimate. We now provide estimates on the size of the network constructed in Step 1. First of all, it is straightforward to see that the depth of the network can be bounded by

$$\mathcal{L}_\varepsilon(\widehat{U}^0_{n,\omega}) \leq \mathcal{L}_\delta(\widehat{g}) + (n-1)\mathcal{L}_\delta(\widehat{F}) \leq \mathcal{L}_\delta(\widehat{g}) + \log_{C_2}(3C_1 \exp\big(m/2\big)/\varepsilon)\mathcal{L}_\delta(\widehat{F}). \tag{C.49}$$

Next we prove an estimate on the number of needed neurons. For notation, we write $\mathcal{M}_n = \mathcal{M}_\varepsilon(\widehat{U}^0_{n,\omega})$. We find that for all $0 \leq k \leq n$,

$$\begin{aligned} \mathcal{M}_k &\leq \mathbb{1}_{\mathbb{N}}(k) m^k (\mathcal{M}_\delta(\widehat{g}) + (k-1)\mathcal{M}_\delta(\mathcal{I})) \\ &\quad + \sum_{i=0}^{k-1} m^{k-i}(2\mathcal{M}_\delta(\widehat{F}) + (2k-2i-1)\mathcal{M}_\delta(\mathcal{I}) + \mathcal{M}_i + \mathcal{M}_{\max\{i-1,0\}}) \\ &\leq \mathbb{1}_{\mathbb{N}}(k)(\mathcal{M}_\delta(\widehat{g}) + \mathcal{M}_\delta(\widehat{F}) + k\mathcal{M}_\delta(\mathcal{I}))(2m)^k + \sum_{i=0}^{k-1} m^{k-i}(\mathcal{M}_i + \mathcal{M}_{\max\{i-1,0\}}). \end{aligned} \tag{C.50}$$

Applying Lemma C.2 to (C.50) (i.e. $\alpha_0 \leftarrow \mathcal{M}_\delta(\widehat{g}) + \mathcal{M}_\delta(\widehat{F})$, $\alpha_1 \leftarrow \mathcal{M}_\delta(\mathcal{I})$ and $\beta \leftarrow 2m$) then gives us that

$$\mathcal{M}_n \leq \frac{1}{2}(\mathcal{M}_\delta(\widehat{g}) + \mathcal{M}_\delta(\widehat{F}) + \mathcal{M}_\delta(\mathcal{I}))(1+\sqrt{2})^n (2m)^n. \tag{C.51}$$

Observing that $2 + 2\sqrt{2} \leq C_2$ and recalling that $m = C_2^{2(1+1/\sigma)}$ we find that

$$\begin{aligned} \mathcal{M}_n &\leq \frac{1}{2}(\mathcal{M}_\delta(\widehat{g}) + \mathcal{M}_\delta(\widehat{F}) + \mathcal{M}_\delta(\mathcal{I})) C_2^{(3\sigma+2)n/\sigma} \\ &= \frac{1}{2}(\mathcal{M}_\delta(\widehat{g}) + \mathcal{M}_\delta(\widehat{F}) + \mathcal{M}_\delta(\mathcal{I})) \left(\frac{4C_1 \exp\big(m/2\big)}{\varepsilon} \right)^{2+3\sigma}. \end{aligned} \tag{C.52}$$

For the width, we make the estimate $\mathrm{width}_\varepsilon(\widehat{U}^0_{n,\omega}) \leq \mathcal{M}_n$.

□

C.6 PINN approximation of nonlinear parabolic equations

Setting C.12. *Assume Setting C.7, let $\widehat{g} \in C(\mathbb{T}^d, \mathbb{R}) \cap L^2(\mu)$[3] and let $\omega : [0,T] \times \mathbb{N}^2 \to \Omega$ be defined as in* (C.37) *in the proof of Theorem C.11. Let $\widehat{U}^\theta_{n,\omega} \colon [0,T] \times \mathbb{T}^d \times \Omega \to \mathbb{R}$, $n \in \mathbb{Z}$, $\theta \in \Theta$, satisfy for all $n \in \mathbb{N}_0$, $\varepsilon > 0$, $\theta \in \Theta$, $t \in [0,T]$, $x \in \mathbb{T}^d$ that*

$$\begin{aligned} \widehat{U}^\theta_{n,\omega}(t,x) &= \frac{\mathbb{1}_{\mathbb{N}}(n)}{m^n} \left[\sum_{k=1}^{m^n} (\mathcal{I}_\varepsilon^{n-1} \circ \widehat{g})(x + W_{T-t}^{(\theta,0,-k)}(\omega(t,n,m))) \right] \\ &\quad + \sum_{i=0}^{n-1} \frac{(T-t)}{m^{n-i}} \Bigg[\sum_{k=1}^{m^{n-i}} \Big((\mathcal{I}_\varepsilon^{n-i-1} \circ \widehat{F}_\varepsilon)(\widehat{U}^{(\theta,i,k)}_{i,\omega}) \\ &\quad - \mathbb{1}_{\mathbb{N}}(i)(\mathcal{I}_\varepsilon^{n-i} \circ \widehat{F}_\varepsilon)(\widehat{U}^{(\theta,-i,k)}_{i-1,\omega}) \Big) \big(\mathcal{U}_t^{(\theta,i,k)}(\omega(t,n,m)), x + W^{(\theta,i,k)}_{\mathcal{U}_t^{(\theta,i,k)}-t}(\omega(t,n,m)) \big) \Bigg]. \end{aligned} \tag{C.53}$$

[3]The function $\widehat{g}$ can but need not be the same as the function $\widehat{g}_\varepsilon$, for some $\varepsilon > 0$, of Setting C.7.

Lemma C.13. *Assume Setting C.12. Under the assumption that,*

$$\max_{1\le j\le k}\left\|\mathcal{I}^j\right\|_{C^k([-\mathcal{L}-1,\mathcal{L}+1])}\le 2, \tag{C.54}$$

where $\mathcal{I}^j$ denotes j compositions of $\mathcal{I}$, it holds for all $\ell,k\in\mathbb{N}_0$ that,

$$\sup_{\theta\in\Theta}\left\|\widehat{U}^\theta_{k,\omega}\right\|_{C^{(0,\ell)}([0,T]\times\mathbb{T}^d)}\le C_{k,\ell}:=\left[\|\widehat{g}\|_{C^\ell(\mathbb{T}^d)}+2B_\ell(1+\sqrt{2})^k(1+2B_\ell T\left\|\widehat{F}\right\|^\ell_{C^\ell(\mathbb{R})})^k\right]^{2(\ell+1)!}, \tag{C.55}$$

and where B_ℓ denote the ℓ-th Bell number i.e., the number of possible partitions of a set with ℓ elements.

Proof. We prove the claim by induction on ℓ.

Base case. From Definition C.8, we find that for $\ell=0$ and all $k\in\mathbb{N}_0$ it holds that

$$\sup_{\theta\in\Theta}\left\|\widehat{U}^\theta_{k,\omega}\right\|_{C^0([0,T]\times\mathbb{T}^d)}\le\|\widehat{g}\|_{C^0(\mathbb{T}^d)}+2T\left\|\widehat{F}\right\|_{C^0(\mathbb{R})}k. \tag{C.56}$$

Claim (C.55) follows immediately for $\ell=0$.

Induction step. We assume that claim (C.55) holds true for all $0\le\ell^*\le\ell-1$ and $k\in\mathbb{N}_0$. From this assumption, we will deduce that (C.55) holds true for ℓ and all $k\in\mathbb{N}_0$. We first observe that it follows from Lemma C.4, the induction hypothesis and the fact that $(C_{k,\ell})_{\ell\ge0}$ is non-decreasing for any k, that for all $\theta\in\Theta$ and $0\le i,j\le k$ it holds that,

$$\left|(\mathcal{I}^j\circ\widehat{F})(\widehat{U}^\theta_{i,\omega})\right|_{C^\ell([0,T]\times\mathbb{T}^d)}\le\left\|\mathcal{I}^j\circ\widehat{F}\right\|_{C^\ell(\mathbb{R})}\left(B_\ell C^\ell_{k,\ell-1}+\left|\widehat{U}^\theta_{i,\omega}\right|_{C^{(0,\ell)}([0,T]\times\mathbb{T}^d)}\right), \tag{C.57}$$

and where (again using Lemma C.4) it holds that $\left\|\mathcal{I}^j\circ\widehat{F}\right\|_{C^\ell(\mathbb{R})}\le 2B_\ell\left\|\widehat{F}\right\|^\ell_{C^\ell}$.

Using this estimate and the fact that $(C_{k,\ell})_{k\ge0}$ is non-decreasing for any ℓ, we can make the following calculation for every $k\in\mathbb{N}_0$,

$$\begin{aligned}
&\sup_{\theta\in\Theta}\left|\widehat{U}^\theta_{k,\omega}\right|_{C^{(0,\ell)}([0,T]\times\mathbb{T}^d)}\\
&\le\mathbb{1}_{\mathbb{N}}(k)|\widehat{g}|_{C^\ell(\mathbb{T}^d)}+T\sum_{i=0}^{k-1}\sup_{\theta\in\Theta}\left|(\mathcal{I}^{k-i-1}\circ\widehat{F})(\widehat{U}^\theta_{i,\omega})\right|_{C^{(0,\ell)}(([0,T]\times\mathbb{T}^d))}\\
&\quad+T\sum_{i=0}^{k-1}\mathbb{1}_{\mathbb{N}}(i)\sup_{\theta\in\Theta}\left|(\mathcal{I}^{k-i}\circ\widehat{F})(\widehat{U}^\theta_{i-1,\omega})\right|_{C^{(0,\ell)}(([0,T]\times\mathbb{T}^d))}\\
&\le\mathbb{1}_{\mathbb{N}}(k)|\widehat{g}|_{C^\ell(\mathbb{T}^d)}+2B_\ell T\left\|\widehat{F}\right\|^\ell_{C^\ell(\mathbb{R})}\sum_{i=0}^{k-1}\left(B_\ell C^\ell_{k,\ell-1}+\sup_{\theta\in\Theta}\left|\widehat{U}^\theta_{i,\omega}\right|_{C^{(0,\ell)}(([0,T]\times\mathbb{T}^d))}\right)\\
&\quad+2B_\ell T\left\|\widehat{F}\right\|^\ell_{C^\ell(\mathbb{R})}\sum_{i=0}^{k-1}\mathbb{1}_{\mathbb{N}}(i)\left(B_\ell C^\ell_{k,\ell-1}+\sup_{\theta\in\Theta}\left|\widehat{U}^\theta_{i-1,\omega}\right|_{C^{(0,\ell)}(([0,T]\times\mathbb{T}^d))}\right)\\
&\le\mathbb{1}_{\mathbb{N}}(k)(|\widehat{g}|_{C^\ell(\mathbb{T}^d)}+2B_\ell T\left\|\widehat{F}\right\|^\ell_{C^\ell(\mathbb{R})}C^\ell_{k,\ell-1}k)\\
&\quad+\sum_{i=0}^{k-1}2B_\ell T\left\|\widehat{F}\right\|^\ell_{C^\ell(\mathbb{R})}\left(\sup_{\theta\in\Theta}\left|\widehat{U}^\theta_{i,\omega}\right|_{C^{(0,\ell)}(([0,T]\times\mathbb{T}^d))}+\mathbb{1}_{\mathbb{N}}(i)\sup_{\theta\in\Theta}\left|\widehat{U}^\theta_{i-1,\omega}\right|_{C^{(0,\ell)}(([0,T]\times\mathbb{T}^d))}\right)
\end{aligned} \tag{C.58}$$

Application of Lemma C.2 with $\alpha_0 \leftarrow |\widehat{g}|_{C^\ell(\mathbb{T}^d)}$, $\alpha_1 \leftarrow 2B_\ell C^\ell_{k,\ell-1}$, $\beta \leftarrow (1 + 2B_\ell T \left\|\widehat{F}\right\|^\ell_{C^\ell(\mathbb{R})})$ and $\gamma \leftarrow 0$ gives us

$$\begin{aligned} \sup_{\theta\in\Theta} \left\|\widehat{U}^\theta_{k,\omega}\right\|_{C^0([0,T]\times\mathbb{T}^d)} &\leq \frac{|\widehat{g}|_{C^\ell(\mathbb{T}^d)} + 2B_\ell C^\ell_{k,\ell-1}}{2}(1+\sqrt{2})^k(1+2B_\ell T\left\|\widehat{F}\right\|^\ell_{C^\ell(\mathbb{R})})^k \\ &\leq \left[|\widehat{g}|_{C^\ell(\mathbb{T}^d)} + 2B_\ell(1+\sqrt{2})^k(1+2B_\ell T\left\|\widehat{F}\right\|^\ell_{C^\ell(\mathbb{R})})^k\right] C^\ell_{k,\ell-1}. \end{aligned} \tag{C.59}$$

Filling in the definition of $C_{k,\ell-1}$ indeed gives us the formula as stated in (C.55), thereby concluding the proof of the claim. □

Lemma C.14. *Let F be a polynomial. For every $\sigma, \varepsilon > 0$ there is an operator $\mathcal{U}^\varepsilon$ as in Assumption 3.1 such that for every $t \in [0, T]$,*

$$\left\|\mathcal{U}^\varepsilon(u_0,t) - \mathcal{G}(v)(u_0,t)\right\|_{L^2(\mathbb{T}^d)} \leq \varepsilon, \qquad C^B_{\varepsilon,\ell} \leq C(B\varepsilon^{-\sigma})^{2(l+1)!}, \qquad C^\varepsilon_{\text{stab}} \leq C\varepsilon^{-\sigma}. \tag{C.60}$$

Moreover it holds that $\operatorname{depth}(\mathcal{U}^\varepsilon(u_0,t)) \leq \operatorname{depth}(\widehat{u}_0) + C\ln(\varepsilon^{-1})$, $\operatorname{width}(\mathcal{U}^\varepsilon(u_0,t)) \leq \operatorname{width}(\widehat{u}_0)\varepsilon^{-2-\sigma}$ *and* $\operatorname{size}(\mathcal{U}^\varepsilon(u_0,t)) \leq \operatorname{size}(\widehat{u}_0)\varepsilon^{-2-\sigma}$.

Proof. The three bounds are a consequence of, respectively, Theorem C.11 and Lemma C.13 and (C.45). The size estimates follow from Theorem C.11. Note that one might have to rescale the constant $\sigma > 0$. □

D Additional material for Section 4.2

D.1 Errors of DeepONets

In [43], numerous error estimates for DeepONets are proven, with a focus on DeepONets that use the ReLU activation function. In order to quantify this error, the authors fix a probability measure $\mu \in \mathcal{P}(\mathcal{X})$ and define the error as,

$$\widehat{\mathscr{E}} = \left(\int_{\mathcal{X}}\int_{U} \left|\mathcal{G}(u)(y) - \mathcal{G}_\theta(u)(y)\right|^2 \, dy\, d\mu(u)\right)^{1/2}, \tag{D.1}$$

assuming that there exist embeddings $\mathcal{X} \hookrightarrow L^2(D)$ and $\mathcal{Y} \hookrightarrow L^2(U)$. From [43, Lemma 3.4], it then follows that $\widehat{\mathscr{E}}$ (D.1) can be bounded as,

$$\widehat{\mathscr{E}} \leq \operatorname{Lip}_\alpha(\mathcal{G})\operatorname{Lip}(\mathcal{R}\circ\mathcal{P})\,(\widehat{\mathscr{E}}_{\mathcal{E}})^\alpha + \operatorname{Lip}(\mathcal{R})\widehat{\mathscr{E}}_{\mathcal{A}} + \widehat{\mathscr{E}}_{\mathcal{R}}, \tag{D.2}$$

where $\operatorname{Lip}_\alpha(\cdot)$ denotes the α-Hölder coefficient of an operator and where $\widehat{\mathscr{E}}_{\mathcal{E}}$ quantifies the encoding error, where $\widehat{\mathscr{E}}_{\mathcal{A}}$ is the error incurred in approximating the approximator $\mathcal{A}$ and where $\widehat{\mathscr{E}}_{\mathcal{R}}$ quantifies the reconstruction error. Assuming that all Hölder coefficients are finite, one can prove that $\widehat{\mathscr{E}}$ is small if $\widehat{\mathscr{E}}_{\mathcal{E}}$, $\widehat{\mathscr{E}}_{\mathcal{A}}$ and $\widehat{\mathscr{E}}_{\mathcal{R}}$ are all small. We summarize how each of these three errors can be bounded using the results from [43].

- The upper bound on the encoding error $\widehat{\mathscr{E}}_{\mathcal{E}}$ depends on the chosen sensors and the spectral decay rate for the covariance operator associated to the measure μ. Use bespoke sensor points to obtain optimals bounds when possible, otherwise use random sensors to obtain almost optimal bounds. More information can be found in [43, Section 3.5].
- The upper bound on the reconstruction error $\widehat{\mathscr{E}}_{\mathcal{R}}$ depends on the smoothness of the operator and the chosen basis functions $\boldsymbol{\tau}$ i.e., neural networks, for the reconstruction operator $\mathcal{R}$. Following [43, Section 3.4], one first chooses a standard basis $\widetilde{\boldsymbol{\tau}}$ of which the properties are well-known. We denote the corresponding reconstruction by $\widetilde{\mathcal{R}}$ and the corresponding reconstruction error by $\widehat{\mathscr{E}}_{\widetilde{\mathcal{R}}}$. In this work, we focus on Fourier and Legendre basis function,

both of which are introduced in **SM** A. One then proceeds by constructing the neural network basis $\boldsymbol{\tau}$ i.e., the trunk nets, that satisfy for some $\varepsilon > 0$ and $p \geq 1$ the condition

$$\max_{k=1,\dots,p} \|\tau_k - \widetilde{\tau}_k\|_{L^2} \leq \frac{\varepsilon}{p^{3/2}}, \tag{D.3}$$

which is shown to imply that,

$$\widehat{\mathscr{E}}_{\mathcal{R}} \leq \widehat{\mathscr{E}}_{\widetilde{\mathcal{R}}} + C\varepsilon, \tag{D.4}$$

where $C \geq 1$ depends only on $\int_{L^2} \|u\|^2 \, d\mathcal{G}_{\#}\mu(u)$. Using standard approximation theory, one can calculate an upper bound on $\widehat{\mathscr{E}}_{\widetilde{\mathcal{R}}}$ and using neural network theory one can quantify the network size of $\boldsymbol{\tau}$ needed such that (D.3) is satisfied. For the Fourier and Legendre bases such results are presented in Lemma D.1 and Lemma D.2, respectively.

- The upper bound on the approximation error $\widehat{\mathscr{E}}_{\mathcal{A}}$ depends on the regularity of the operator $\mathcal{G}$. We present the tanh counterparts of some results of [43, Section 3.6] in the following sections, with the main result being Theorem D.6.

For bounded linear operators, these calculations are rather straightforward and are presented in [43, **SM** D]. For nonlinear operators, one has to complete all the above steps for each specific case. In [43, Section 4], this has been done for four types of differential equations.

D.2 Auxiliary results for linear operators

Following Section D.1, we need results on the required neural network size to approximate the reconstruction basis to a certain accuracy (D.3). The following lemma provides such a result for the Fourier basis introduced in **SM** A.5.

Lemma D.1. *Let $s, d, p \in \mathbb{N}$. For any $\varepsilon > 0$, there exists a trunk net $\boldsymbol{\tau} : \mathbb{R}^d \to \mathbb{R}^p$ with 2 hidden layers of width $\mathcal{O}(p^{\frac{d+1}{d}} + ps\ln\big(ps\varepsilon^{-1}\big))$ and such that*

$$p^{3/2} \max_{j=1,\dots,p} \|\tau_j - \mathbf{e}_j\|_{C^s([0,2\pi]^d)} \leq \varepsilon, \tag{D.5}$$

*where $\mathbf{e}_1, \dots, \mathbf{e}_p$ denote the first p elements of the Fourier basis, as in **SM** A.5.*

Proof. We note that each element in the (real) trigonometric basis $\mathbf{e}_1, \dots, \mathbf{e}_p$ can be expressed in the form

$$\mathbf{e}_j(x) = \cos(\kappa \cdot x), \quad \text{or} \quad \mathbf{e}_j(x) = \sin(\kappa \cdot x), \tag{D.6}$$

for $\kappa = \kappa(j) \in \mathbb{Z}^d$ with $|\kappa|_\infty \leq N$, where N is chosen as the smallest natural number such that $p \leq (2N+1)^d$. We focus only focus on the first form, as the proof for the second form is entirely similar. Define $f : [0, 2\pi]^d \to \mathbb{R} : x \mapsto \kappa \cdot x$ and $g : [-2\pi dN, 2\pi dN] \to \mathbb{R} : x \mapsto \cos(x)$. As $f([0, 2\pi]^d) \subset [-2\pi dN, 2\pi dN]$, the composition $g \circ f$ is well-defined and one can see that it coincides with a trigonometric basis function $\mathbf{e}_j$. Moreover, the linear map f is a trivial neural network without hidden layers. Approximating $\mathbf{e}_j$ by a neural network τ_j therefore boils down to approximating g by a suitable neural network.

From [15, Theorem 5.1] it follows that the function g there exists an independent constant $R > 0$ such that for large enough $t \in \mathbb{N}$ there is a tanh neural network $\widehat{g}_t$ with two hidden layers and $\mathcal{O}(t + N)$ neurons such that

$$\|g - \widehat{g}_t\|_{C^s([-2\pi dN, 2\pi dN])} \leq 4(8(s+1)^3 R)^s \exp(t - s). \tag{D.7}$$

This can be proven from [15, eq. (74)] by setting $\delta \leftarrow \frac{1}{3}$, $k \leftarrow s$, $s \leftarrow t$, $N \leftarrow 2$ and using $\|g\|_{C^s} = 1$ and Stirling's approximation to obtain

$$\frac{1}{(t-s)!}\left(\frac{3}{2 \cdot 2}\right)^{t-s} \leq \frac{1}{\sqrt{2\pi(t-s)}}\left(\frac{e}{t-s}\right)^{t-s} \leq \exp(s-t) \quad \text{for } t > s + e^2. \tag{D.8}$$

Setting $t = \mathcal{O}(\ln\big(\delta^{-1}\big) + s\ln(s))$ then gives a neural network $\widehat{g}_t$ with $\|g - \widehat{g}_t\|_{C^s} < \eta$. Next, it follows from [15, Lemma A.7] that

$$\begin{aligned} \|g \circ f - \widehat{g}_t \circ f\|_{C^s([0,2\pi]^d)} &\leq 16(e^2 s^4 d^2)^s \|g - \widehat{g}_t\|_{C^s([-2\pi dN, 2\pi dN])} \|f\|^s_{C^s([0,2\pi]^d)} \\ &\leq 16(e^2 s^4 d^2)^s \eta (2\pi dN)^s. \end{aligned} \tag{D.9}$$

From this follows that we can obtain the desired accuracy (D.5) if we set $\tau_j = \widehat{g}_{t(\eta)} \circ f$ with

$$\eta = \frac{\varepsilon p^{-3/2}}{16(2\pi N d^3 e^2 s^4)^s}, \tag{D.10}$$

which amounts to $t = \mathcal{O}(s \ln\big(sN\varepsilon^{-1}\big))$. As a consequence, the tanh neural network τ_j has two hidden layers with $\mathcal{O}(s \ln\big(sN\varepsilon^{-1}\big) + N)$ neurons and therefore, by recalling that $p \sim N^d$, the combined network $\boldsymbol{\tau}$ has two hidden layers with

$$\mathcal{O}(p(s \ln\Big(sN\varepsilon^{-1}\Big) + N)) = \mathcal{O}(ps \ln\Big(ps\varepsilon^{-1}\Big) + p^{\frac{d+1}{d}}) \tag{D.11}$$

neurons. □

D.3 Proof of Theorem 4.6

Proof. Consider the setting of Theorem 4.6. Using [43, Theorem D.3], the reasoning as in [43, Example D.4] and Lemma D.1 we find that there exists a constant $C = C(d, \ell) > 0$, such that for any $m, p, s \in \mathbb{N}$ there exists a DeepONet with trunk net $\boldsymbol{\tau}$ and branch net $\boldsymbol{\beta}$, such that

$$\text{size}(\boldsymbol{\tau}) \leq C(p^{\frac{d+1}{d}} + ps \ln\Big(ps\varepsilon^{-1}\Big)), \quad \text{depth}(\boldsymbol{\tau}) = 3, \tag{D.12}$$

and where

$$\text{size}(\boldsymbol{\beta}) \leq p, \quad \text{depth}(\boldsymbol{\beta}) \leq 1, \tag{D.13}$$

and such that the DeepONet approximation error (D.1) is bounded by

$$\big\|\mathcal{G}(v) - \mathcal{G}_\theta(v)\big\|_{L^2(\mu\times\lambda)} \leq \varepsilon + C\exp\left(-c\,p^{1/d}\right) + C\exp\left(-\frac{c\,m^{1/d}}{\log(m)^{1/d}}\right). \tag{D.14}$$

Moreover, it holds that

$$\big|\mathcal{N}(u)(\cdot)\big|_{C^s} \leq Cp^{s/d}, \tag{D.15}$$

since in this case $\boldsymbol{\tau}$ approximates the Fourier basis (**SM** A.5). From (A.15), one can then deduce the estimate on the C^s-norm of the DeepONet. This proves that (3.7) in Theorem 3.9 holds with $\sigma(s) = s/d$. This concludes the proof. □

D.4 Auxiliary results for nonlinear operators

We provide a neural network approximation result for the Legendre basis from **SM** A.4.

Lemma D.2. *Let $n, p \in \mathbb{N}$. For any $\varepsilon > 0$, there exists a trunk net $\boldsymbol{\tau} : \mathbb{R}^d \to \mathbb{R}^p$ with two hidden layers of width $\mathcal{O}(p)$ such that*

$$p^{3/2} \max_{j=1,\dots,p} \|\tau_j - L_j\|_{C^s([-1,1]^d)} \leq \varepsilon, \tag{D.16}$$

*where $L_1, \dots, L_p$ denote the first p elements of the Legendre basis, as in **SM** A.4.*

Proof. Let $j \in 1, \dots, p$. It holds by definition of Legendre polynomials and the corresponding enumeration (**SM** A.4) that the degree in every variable is at most p. Therefore, L_j is a product of d univariate polynomials of degree at most p. From [15, Lemma 3.2] it follows that one needs a shallow tanh neural network with $O(p)$ neurons to approximate a univariate polynomial to any accuracy. The result from [15, Corollary 3.7] can be used to construct a shallow tanh network that approximates the product of the d univariate polynomials. Note that its size only depends on the dimension d and not on the polynomial degree p or the accuracy. Finally, [15, Lemma A.7] ensures the accuracy of the composition of the two subnetworks. It then follows that there exist a tanh neural network of width $\mathcal{O}(p)$ and two hidden layers that achieves the wanted error estimate. □

In our proofs, we require tanh counterparts to the results for DeepONets with ReLU activation function from [43]. We present these adapted results below for completeness.

The first lemma considers the neural network approximation of the map $\boldsymbol{u} \mapsto \widehat{Y}(\boldsymbol{u})$, as defined in [43, Eq. (3.59)].

Lemma D.3. *Let $N, d \in \mathbb{N}$, and denote $m := (2N+1)^d$. There exists a constant $C > 0$, independent of N, such that for every N there exists a tanh neural network $\Psi : \mathbb{R}^m \to \mathbb{R}^m$, with*

$$\text{size}(\Psi) \leq C(1 + m\log(m)), \quad \text{depth}(\Psi) \leq C(1 + \log(m)), \tag{D.17}$$

and such that $\Psi(\boldsymbol{u}) = (\widehat{Y}_1(\boldsymbol{u}), \dots, \widehat{Y}_m(\boldsymbol{u}))$, for all $\boldsymbol{u} \in \mathbb{R}^m$.

Proof. The proof is identical to that of [43, Lemma 3.28]. □

We can now state the following result [70, Theorem 3.10] which is the counterpart of [43, Theorem 3.32] for tanh neural networks.

Theorem D.4. *Let V be a Banach space and let $\mathcal{J}$ be a countable index set. Let $\mathcal{F} : [-1,1]^{\mathcal{J}} \to V$ be a $(\boldsymbol{b}, \varepsilon, \kappa)$-holomorphic map for some $\boldsymbol{b} \in \ell^q(\mathbb{N})$ and $q \in (0,1)$, and an enumeration $\kappa : \mathbb{N} \to \mathcal{J}$. Then there exists a constant $C > 0$, such that for every $N \in \mathbb{N}$, there exists an index set*

$$\Lambda_N \subset \left\{ \boldsymbol{\nu} = (\nu_1, \nu_2, \dots) \in \textstyle\prod_{j \in \mathcal{J}} \mathbb{N}_0 \mid \nu_j \neq 0 \text{ for finitely many } j \in \mathcal{J} \right\}, \tag{D.18}$$

with $|\Lambda_N| = N$, a finite set of coefficients $\{c_{\boldsymbol{\nu}}\}_{\boldsymbol{\nu} \in \Lambda_N} \subset V$, and a tanh network $\Psi : \mathbb{R}^N \to \mathbb{R}^{\Lambda_N}$, $y \mapsto \{\Psi_{\boldsymbol{\nu}}(y)\}_{\boldsymbol{\nu} \in \Lambda_N}$ with

$$\text{size}(\Psi) \leq C(1 + N\log(N)), \quad \text{depth}(\Psi) \leq C(1 + \log\log(N)), \tag{D.19}$$

and such that

$$\sup_{\boldsymbol{y} \in [-1,1]^{\mathcal{J}}} \left\| \mathcal{F}(\boldsymbol{y}) - \sum_{\boldsymbol{\nu} \in \Lambda_N} c_{\boldsymbol{\nu}} \Psi_{\boldsymbol{\nu}}(y_{\kappa(1)}, \dots, y_{\kappa(N)}) \right\|_V \leq C N^{1-1/q}. \tag{D.20}$$

Using this theorem, we can state the tanh counterpart to [43, Corollary 3.33].

Corollary D.5. *Let V be a Banach space. Let $\mathcal{F} : [-1,1]^{\mathcal{J}} \to V$ be a $(\boldsymbol{b}, \varepsilon, \kappa)$-holomorphic map for some $\boldsymbol{b} \in \ell^q(\mathbb{N})$ and $q \in (0,1)$, where $\kappa : \mathbb{N} \to \mathcal{J}$ is an enumeration of $\mathcal{J}$. In particular, it is assumed that $\{b_j\}_{j \in \mathbb{N}}$ is a monotonically decreasing sequence. If $\mathcal{P} : V \to \mathbb{R}^p$ is a continuous linear mapping, then there exists a constant $C > 0$, such that for every $m \in \mathbb{N}$, there exists a tanh network $\Psi : \mathbb{R}^m \to \mathbb{R}^p$, with*

$$\text{size}(\Psi) \leq C(1 + pm\log(m)), \quad \text{depth}(\Psi) \leq C(1 + \log\log(m)), \tag{D.21}$$

and such that

$$\sup_{\boldsymbol{y} \in [-1,1]^{\mathcal{J}}} \|\mathcal{P} \circ \mathcal{F}(\boldsymbol{y}) - \Psi(y_{\kappa(1)}, \dots, y_{\kappa(m)})\|_{\ell^2(\mathbb{R}^p)} \leq C\|\mathcal{P}\|\, m^{-s}, \tag{D.22}$$

where $s := q^{-1} - 1 > 0$ and $\|\mathcal{P}\| = \|\mathcal{P}\|_{V \to \ell^2}$ denotes the operator norm.

Proof. The proof is identical to the one presented in [43, Appendix C.18]. □

Finally, we use this result to state the counterpart to [43, Theorem 3.34], which considers the approximation of a parametrized version of the operator $\mathcal{G}$, defined as a mapping

$$\mathcal{F} : [-1,1]^{\mathcal{J}} \to L^2(U) : \boldsymbol{y} \mapsto \mathcal{G}(u(\cdot; \boldsymbol{y})). \tag{D.23}$$

A more detailled discussion can be found in [43, Section 3.6.2].

Theorem D.6. *Let $\mathcal{F} : [-1,1]^{\mathcal{J}} \to L^2(U)$ be $(\boldsymbol{b}, \varepsilon, \kappa)$-holomorphic with $\boldsymbol{b} \in \ell^q(\mathbb{N})$ and $\kappa : \mathbb{N} \to \mathcal{J}$ an enumeration, and assume that $\mathcal{F}$ is given by* (D.23)*. Assume that the encoder/decoder pair is constructed as in [43, Section 3.5.3], so that [43, Eq. (3.69)] holds. Given an affine reconstruction $\mathcal{R} : \mathbb{R}^p \to L^2(U)$, let $\mathcal{P} : L^2(U) \to \mathbb{R}^p$ denote the corresponding optimal linear projection [43, Eq. (3.17)]. Then given $k \in \mathbb{N}$, there exists a constant $C_k > 0$, independent of m, p and an approximator $\mathcal{A} : \mathbb{R}^m \to \mathbb{R}^p$ that can be represented by a neural network with*

$$\text{size}(\mathcal{A}) \leq C_k(1 + pm\log(m)), \quad \text{depth}(\mathcal{A}) \leq C_k(1 + \log(m)).$$

and such that the approximation error $\widehat{\mathscr{E}}_{\mathcal{A}}$ can be estimated by

$$\widehat{\mathscr{E}}_{\mathcal{A}} \leq C_k \|\mathcal{P}\|\, m^{-k},$$

where $\|\mathcal{P}\| = \|\mathcal{P}\|_{L^2(U) \to \mathbb{R}^p}$ is the operator norm of $\mathcal{P}$.

Proof. The proof is as in [43, Appendix C.19.1]. □

D.5 Gravity pendulum with external force

Next, we consider the following nonlinear ODE system, already considered in the context of approximation by DeepONets in [49] and [43],

$$\begin{cases} \dfrac{dv_1}{dt} = v_2, \\ \dfrac{dv_2}{dt} = -\gamma \sin(v_1) + u(t). \end{cases} \tag{D.24}$$

with initial condition $v(0) = 0$ and where $\gamma > 0$ is a parameter. Let us denote $v = (v_1, v_2)$ and

$$g(v) := \begin{pmatrix} v_2 \\ -\gamma \sin(v_1) \end{pmatrix}, \quad U(t) := \begin{pmatrix} 0 \\ u(t) \end{pmatrix}, \tag{D.25}$$

so that equation (D.24) can be written in the form

$$\mathcal{L}_u(v) := \frac{dv}{dt} - g(v) + U = 0, \quad v(0) = 0. \tag{D.26}$$

In (D.26), v_1, v_2 are the angle and angular velocity of the pendulum and the constant γ denotes a frequency parameter. The dynamics of the pendulum is driven by an external force u. With the external force u as the input, the output of the system is the solution vector v and the underlying nonlinear operator is given by $\mathcal{G} : L^2([0,T]) \to L^2([0,T]) : u \mapsto \mathcal{G}(u) = v$. Following the discussion in [43], we choose an underlying (parametrized) measure $\mu \in \mathcal{P}(L^2([0,T]))$ as a law of a random field u, that can be expanded in the form

$$u(t;Y) = \sum_{k\in\mathbb{Z}} Y_k \alpha_k \mathbf{e}_k \left(\frac{2\pi t}{T}\right), \quad t \in [0,T], \tag{D.27}$$

where $\mathbf{e}_k(x)$, $k \in \mathbb{Z}$, denotes the one-dimensional standard Fourier basis (A.5) and where the coefficients $\alpha_k \geq 0$ decay to zero as $\alpha_k \leq C_\alpha \exp\big(-|k|\ell\big)$ for some constants $C_\alpha, \ell > 0$. Furthermore, we assume that the $\{Y_k\}_{k\in\mathbb{Z}}$ are iid random variables on $[-1,1]$.

Assuming the described setting, the following lemma gives an error bound of tanh DeepONets in terms of the sizes of the corresponding branch and trunk nets.

Lemma D.7. *Consider the DeepONet approximation problem for the gravity pendulum* (D.24)*, where the forcing $u(t)$ is distributed according to a probability measure $\mu \in \mathcal{P}(L^2([0,T]))$ given as the law of the random field* (D.27)*. For any $k, r \in \mathbb{N}$, there exists a constant $C = C(k,r) > 0$, and a constant $c > 0$, independent of m, p, such that for any m, $p \in \mathbb{N}$, there exists a DeepONet $\mathcal{G}_\theta$ with trunk net $\boldsymbol{\tau}$ and branch net $\boldsymbol{\beta}$, such that*

$$\text{size}(\boldsymbol{\tau}) \leq Cp, \quad \text{depth}(\boldsymbol{\tau}) = 2, \tag{D.28}$$

and

$$\text{size}(\boldsymbol{\beta}) \leq C(1 + pm\log(m)), \quad \text{depth}(\boldsymbol{\beta}) \leq C(1 + \log(m)), \tag{D.29}$$

and such that the DeepONet approximation error (D.1) *is bounded by*

$$\widehat{\mathscr{E}} \leq Ce^{-c\ell m} + Cm^{-k} + Cp^{-r}, \tag{D.30}$$

and that for all $s \in N$,

$$\big|\mathcal{G}_\theta(u)(\cdot)\big|_{C^s} \leq Cp^{d/2+2sd}. \tag{D.31}$$

Proof. The proof of the statement is identical to that of [43, Theorem 4.10], with the only difference that we consider tanh neural networks instead of ReLU neural networks. As a result, the proof comes down to determining the size of the trunk net $\boldsymbol{\tau}$ using Lemma D.2 instead of [64, Proposition 2.10], thereby proving the tanh counterpart of [43, Proposition 4.5], and replacing [43, Proposition 4.9] by Theorem D.6. The C^s-bound of the DeepONet follows from the C^s-bound of Legendre polynomials (A.12) and Lemma D.2. □

We can again follow Theorem 3.9 to obtain error bounds for physics-informed DeepONets. Assumption 3.3 is satisfied for $[0,T]$. As a result, we can apply Theorem 3.9 to obtain the following result.

Theorem D.8. *Consider the setting of Lemma D.7. For every $\beta > 0$, there exists a constant $C > 0$ such that for any $p \in N$, there exists a DeepONet $\mathcal{G}_\theta$ with a trunk net $\boldsymbol{\tau} = (0, \tau_1, \dots, \tau_p)$ with p outputs and branch net $\boldsymbol{\beta} = (0, \beta_1, \dots, \beta_p)$, such that*

$$\operatorname{size}(\boldsymbol{\tau}) \leq Cp, \quad \operatorname{depth}(\boldsymbol{\tau}) = 2, \tag{D.32}$$

and

$$\operatorname{size}(\boldsymbol{\beta}) \leq C(1 + p^2 \log(p)), \quad \operatorname{depth}(\boldsymbol{\beta}) \leq C(1 + \log(p)), \tag{D.33}$$

and such that

$$\left\| \frac{d\mathcal{G}_\theta(u)_1}{dt} - \mathcal{G}_\theta(u)_2 \right\|_{L^2(\mu)} + \left\| \frac{d\mathcal{G}_\theta(u)_2}{dt} + \gamma \sin\big(\mathcal{G}_\theta(u)_1\big) - u(t) \right\|_{L^2(\mu)} \leq Cp^{-\beta}. \tag{D.34}$$

Proof. Lemma D.7 with $s \leftarrow 1$, $k \leftarrow r$ and $m \leftarrow p$ then provides a DeepONet that satisfies the conditions of Theorem 3.9 with $r^* = +\infty$ and equation (3.7) with $\sigma(s) = d/2 + 2sd$. The smoothness of v is guaranteed by [43, Lemma 4.3]. Moreover, it holds that,

$$\left\| \frac{d\mathcal{G}_\theta(u)_1}{dt} - \mathcal{G}_\theta(u)_2 \right\|_{L^2(\mu)} \leq \left\| \frac{d\mathcal{G}_\theta(u)_1}{dt} - \frac{d\mathcal{G}(u)_1}{dt} \right\|_{L^2(\mu)} + \big\| \mathcal{G}(u)_2 - \mathcal{G}_\theta(u)_2 \big\|_{L^2(\mu)}, \tag{D.35}$$

and also that,

$$\begin{aligned} &\left\| \frac{d\mathcal{G}_\theta(u)_2}{dt} + \gamma \sin\big(\mathcal{G}_\theta(u)_1\big) - u(t) \right\|_{L^2(\mu)} \\ &\quad \leq \left\| \frac{d\mathcal{G}_\theta(u)_2}{dt} - \frac{d\mathcal{G}(u)_2}{dt} \right\|_{L^2(\mu)} + \gamma \Big\| \sin\big(\mathcal{G}(u)_1\big) - \sin\big(\mathcal{G}_\theta(u)_1\big) \Big\|_{L^2(\mu)} \\ &\quad \leq \left\| \frac{d\mathcal{G}_\theta(u)_2}{dt} - \frac{d\mathcal{G}(u)_2}{dt} \right\|_{L^2(\mu)} + \gamma \big\| \mathcal{G}(u)_1 - \mathcal{G}_\theta(u)_1 \big\|_{L^2(\mu)}. \end{aligned} \tag{D.36}$$

Combining this estimate with Theorem 3.9 with $k = 2$ then gives the wanted result. □

D.6 An elliptic PDE: Multi-d diffusion with variable coefficients

Next, again following [43], we consider a popular model problem for elliptic PDEs with unknown diffusion coefficients [11] and references therein. For the sake of definiteness and simplicity, we shall assume a periodic domain $D = \mathbb{T}^d$ in the following. For $b \in \mathbb{N}_0$, we consider an elliptic PDE with variable coefficients a,

$$\mathcal{L}_a(u) := \nabla \cdot (a(x) \nabla u(x)) + f(x) = 0, \tag{D.37}$$

for $u \in C^{b+2}(D)$ with suitable boundary conditions, and for fixed $f \in C^b(D)$. Similar to the previous examples, we fix a probability measure μ on the coefficient a by assuming that every a can be written as

$$a(x, Y) = \overline{a}(x) + \sum_{k \in \mathbb{Z}^d} \alpha_k Y_k \mathbf{e}_k(x), \tag{D.38}$$

with notation from **SM** A.5, and where for simplicity $\overline{a}(x) \equiv 1$ is assumed to be constant. Furthermore, we will consider the case of smooth coefficients $x \mapsto a(x; Y)$, which is ensured by requiring that there exist constants $C_\alpha > 0$ and $\ell > 1$, such that $|\alpha_k| \leq C_\alpha \exp\big(-\ell |k|_\infty\big)$ for all $k \in \mathbb{Z}^d$. Still following [43], we define $\boldsymbol{b} = (b_1, b_2, \dots) \in \ell^1(\mathbb{N})$ by

$$b_j := C_\alpha \exp\big(-\ell |\kappa(j)|_\infty\big), \tag{D.39}$$

where $\kappa : \mathbb{N} \to \mathbb{Z}^d$ is the enumeration for the standard Fourier basis, (**SM** A.5). Note that by assumption on the enumeration κ, we have that $\boldsymbol{b}$ is a monotonically decreasing sequence. In the following, we will assume throughout that $\|\boldsymbol{b}\|_{\ell^1} < 1$, ensuring a uniform coercivity condition on all random coefficients $a = a(\,\cdot\,; Y)$ in (D.37). Finally, we assume that the $Y_j \in [-1, 1]$ are centered random variables and we let $\mu \in \mathcal{P}(L^2(\mathbb{T}^d))$ denote the law of the random coefficient (D.38).

The following lemma provides an error estimate for DeepONets approximating the operator $\mathcal{G}$ that maps the input coefficient a into the solution field u of the PDE (D.37).

Lemma D.9. *For any $k, r \in \mathbb{N}$, there exists a constant $C > 0$, such that for any $m, p \in \mathbb{N}$, there exists a DeepONet $\mathcal{G}_\theta = \mathcal{R} \circ \mathcal{A} \circ \mathcal{E}$ with m sensors, a trunk net $\boldsymbol{\tau} = (0, \tau_1, \ldots, \tau_p)$ with p outputs and branch net $\boldsymbol{\beta} = (0, \beta_1, \ldots, \beta_p)$, such that*

$$\text{size}(\boldsymbol{\beta}) \leq C(1 + pm\log(m)), \quad \text{depth}(\boldsymbol{\beta}) \leq C(1 + \log(m)), \tag{D.40}$$

and

$$\text{size}(\boldsymbol{\tau}) \leq Cp^{\frac{d+1}{d}} \quad \text{depth}(\boldsymbol{\tau}) \leq 2 \tag{D.41}$$

such that the DeepONet approximation error (D.1) *satisfies*

$$\widehat{\mathscr{E}} \leq Ce^{-c\ell m^{\frac{1}{d}}} + Cm^{-k} + Cp^{-r}, \tag{D.42}$$

and that for all $s \in N$

$$\left|\mathcal{G}_\theta(u)(\cdot)\right|_{C^s} \leq Cp^{s/d}. \tag{D.43}$$

Proof. This statement is the tanh counterpart of [43, Theorem 4.19], which addresses ReLU DeepONets. We only highlight the differences in the proof. First, one should use Lemma D.1 instead of [43, Lemma 3.13], which then results in different network sizes in [43, Lemma 3.14, Proposition 3.17, Corollary 3.18, Proposition 4.17]. Second, one needs to replace [43, Proposition 4.18] with Theorem D.6.

Moreover, in this case the trunk net $\boldsymbol{\tau}$ approximates the Fourier basis (**SM** A.5). From (A.15), one can then deduce the estimate on the C^s-norm of the DeepONet. □

It is straightforward to verify that the conditions of Theorem 3.9 are satisfied in the current setting. Applying Theorem 3.9 then results in the following theorem on the error of physics-informed DeepONets for (D.37).

Theorem D.10. *Consider the elliptic equation* (D.37) *with $b \geq 1$. For every $\beta > 0$, there exists a constant $C > 0$ such that for any $p \in N$, there exists a DeepONet $\mathcal{G}_\theta$ with a trunk net $\boldsymbol{\tau} = (0, \tau_1, \ldots, \tau_p)$ with p outputs and branch net $\boldsymbol{\beta} = (0, \beta_1, \ldots, \beta_p)$, such that*

$$\text{size}(\boldsymbol{\beta}) \leq C(1 + p^2\log(p)), \quad \text{depth}(\boldsymbol{\beta}) \leq C(1 + \log(p)), \tag{D.44}$$

and

$$\text{size}(\boldsymbol{\tau}) \leq Cp^2 \quad \text{depth}(\boldsymbol{\tau}) \leq 2 \tag{D.45}$$

such that

$$\left\|\nabla \cdot (a(x)\nabla\mathcal{G}_\theta(a)(x)) - f(x)\right\|_{L^2(\mu)} \leq Cp^{-\beta}. \tag{D.46}$$

Proof. We first check the conditions of Theorem 3.9. Lemma D.9 with $s \leftarrow 1$, $k \leftarrow r$ and $m \leftarrow p$ then provides a DeepONet that satisfies the conditions of Theorem 3.9 with $r^* = +\infty$ and equation (3.7) with $\sigma(s) = s/d$. Moreover, the following estimate holds,

$$\begin{aligned}
&\left\|\nabla \cdot (a(x)\nabla\mathcal{G}_\theta(a)(x)) - f(x)\right\|_{L^2(\mu)} \\
&\leq \left\|\nabla \cdot (a(x)\nabla\mathcal{G}_\theta(a)(x)) - \nabla \cdot (a(x)\nabla\mathcal{G}(a)(x))\right\|_{L^2(\mu)} \\
&\leq \sum_{j=1}^{d} \|a\|_{C^0} \left\|\partial_j^2(\mathcal{G}_\theta - \mathcal{G})(a)(x)\right\|_{L^2(\mu)} + \sum_{j=1}^{d} \|a\|_{C^1} \left\|\partial_j(\mathcal{G}_\theta - \mathcal{G})(a)(x)\right\|_{L^2(\mu)}.
\end{aligned} \tag{D.47}$$

Combining this estimate with Theorem 3.9 with $k = 2$ then gives the wanted result. □