A Priori Estimation of the Approximation, Optimization and Generalization Errors of Random Neural Networks for Solving Partial Differential Equations

Xianliang Xu[4] , **Ye Li**[1,2,3*] , **Zhongyi Huang**[4*]
[1]College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics
[2]MIIT Key Laboratory of Pattern Analysis and Machine Intelligence, Nanjing
[3]State Key Laboratory for Novel Software Technology, Nanjing University
[4]Department of Mathematical Sciences, Tsinghua University
xuxl19@mails.tsinghua.edu.cn, yeli20@nuaa.edu.cn, zhongyih@tsinghua.edu.cn

Abstract

In recent years, neural networks have achieved remarkable progress in various fields and have also drawn much attention in applying them on scientific problems. A line of methods involving neural networks for solving partial differential equations (PDEs), such as Physics-Informed Neural Networks (PINNs) and the Deep Ritz Method (DRM), has emerged. Although these methods outperform classical numerical methods in certain cases, the optimization problems involving neural networks are typically non-convex and non-smooth, which can result in unsatisfactory solutions for PDEs. In contrast to deterministic neural networks, the hidden weights of random neural networks are sampled from some prior distribution and only the output weights participate in training. This makes training much simpler, but it remains unclear how to select the prior distribution. In this paper, we focus on Barron type functions and approximate them under Sobolev norms by random neural networks with clear prior distribution. In addition to the approximation error, we also derive bounds for the optimization and generalization errors of random neural networks for solving PDEs when the solutions are Barron type functions.

1 Introduction

As the development of hardcore and algorithms, deep neural networks have made remarkable progress across various fields, including computer vision [He *et al.*, 2016], natural language processing [Devlin, 2018], reinforcement learning [Silver *et al.*, 2016], and others. These successes have inspired researchers to explore the application of neural networks to scientific challenges, particularly in the modeling of physical systems. In the field of scientific computing, a long-standing problem is solving partial differential equations (PDEs) numerically, which can be hindered by the curse of dimensionality when using classical numerical methods. To tackle PDE-related problems, several neural network-based approaches have been introduced. Among these, Physics-Informed Neural Networks (PINNs) and the Deep Ritz Method (DRM) stand out. PINN incorporates prior information from PDEs into the training in solving forward and inverse problems of PDEs. Specifically, it encodes the PDE constraints into the design of the loss function, which restricts the constructed neural network to follow physical law characterized by the PDE. This framework's flexibility stems from its reliance solely on the PDE's form, making it adaptable for a wide range of PDEs. The DRM, on the other hand, incorporates the variational formulation that is an essential tool in traditional methods, into training the neural networks. This approach typically involves lower-order derivatives, potentially offering greater computational efficiency compared to PINNs. However, the DRM's utility is somewhat limited by the fact that not all PDEs possess a variational formulation.

The success of neural networks can be partly attributed to their powerful approximation capabilities. The capacity of neural networks with a variety of activation functions to approximate different types of target functions has been extensively studied. This includes continuous functions [Shen *et al.*, 2022], smooth functions [Lu *et al.*, 2021a], as well as Sobolev functions [Belomestny *et al.*, 2023; Yang *et al.*, 2023b; Yang *et al.*, 2023a; Yarotsky, 2017] and Barron functions [Barron, 1993; Lu *et al.*, 2021b; Siegel and Xu, 2020a], among others. In short, neural networks can approximate the aforementioned function classes with arbitrary precision. Despite the widespread adoption and impressive approximation capabilities of neural networks in scientific computing, their practical application can encounter challenges. A fundamental issue arises in solving the optimization problems involving neural networks, which are typically non-convex and non-smooth. For example, [Krishnapriyan *et al.*, 2021] demonstrates that even for simple problems of convection, reaction, and reaction-diffusion, PINN approach only works for very easy parameter regimes and fails in more challenging physical regimes. By analyzing the loss landscape of PINN, they show that the failure is not due to the limited capacity of the neural network, but rather due to the optimization difficulties caused by the PINN's soft PDE constraint.

Due to the limitations described before, there is a growing interest in the application of random neural networks, whose hidden weights are randomly generated and only the output weights are trainable. Compared to deterministic neu-

*Corresponding author: Ye Li <yeli20@nuaa.edu.cn> and Zhongyi Huang <zhongyih@tsinghua.edu.cn>

ral networks, random neural networks can lead to optimization problems that can be efficiently solved. For instance, employing random neural networks in L^2 regression problems results in least squares problems, which possess closed-form solutions or can be addressed by various optimization algorithms. Because of the favorable properties, random neural networks have been successfully applied not only in traditional machine learning tasks but also in addressing PDE-related problems. In this work, we focus on the utilization of random neural networks in the framework of PINNs. First, we establish the approximation error of neural networks for Barron-type functions under the H^2 norm. Subsequently, for applying random neural networks for solving a certain class of second-order elliptic PDEs with Barron-type solutions, we provide the optimization and generalization error analysis, providing a comprehensive understanding of how random neural networks perform when applied to PDEs.

1.1 Contributions

The contribution of this work can be summarized as follows.

- We provides two approximation results for two Barron-type functions via random neural networks under the Sobolev norms. Unlike previous works that only showed the existence of the prior distributions, we give the concise forms of these distributions.
- When applying random neural networks for solving certain second-order elliptic PDEs whose solutions are Barron-type, we design tailored optimization algorithms for addressing these problems. Subsequently, we perform a full error analysis that rigorously bounds the approximation, optimization, and generalization errors. In deriving the generalization bounds, we also show that projected gradient descent can achieve the faster rate $\mathcal{O}(1/n)$.
- We have conducted numerical experiments to validate our conclusions.

1.2 Related Works

Random Neural Networks in Machine Learning. Random neural networks, also known as Extreme Learning Machines (ELMs) in the field of machine learning, have drawn significant attention due to their special training methodology and efficiency. The concept of ELMs was first introduced in [Huang *et al.*, 2006] as a two-layer random neural network where the hidden weights are randomly assigned, and only the output weights are trained. As shown in [Huang *et al.*, 2006], this algorithm can provide good generalization performance at extremely fast learning speed and it outperforms deterministic neural networks in regression, classification and real-world complex problems. It is also possible to extend ELM to kernel learning [Huang *et al.*, 2011], which shows that ELM can use a wide type of feature mappings, including random hidden nodes and kernels.

Random Neural Networks in Scientific Computing. The potential of random neural networks extends beyond traditional machine learning domains, they have also been effectively utilized in scientific computing, particularly for solving problems related to PDEs. For instance, [Dong and Li, 2021] has developed an efficient method based on domain decomposition and random neural networks to solve different types of PDEs, showing the first time when the neural network-based methods outperform the traditional numerical methods in low dimensions. Later, this method has been extended to high dimensions [Wang and Dong, 2024], producing accurate solutions to high-dimensional PDEs.

[Gonon *et al.*, 2023] has studied approximation based on single-hidden-layer feedforward and recurrent neural networks with randomly generated hidden weights under the L^2 norm. Similar to our work, they are also based on an integral representations of the target functions. However, the prior distribution for the hidden weights remains unclear in their study. In contrast, [Neufeld and Schmocker, 2023] derived approximation rates and an explicit algorithm to learn a deterministic function by a random neural network under certain Sobolev norms, but the conditions for the target functions to be approximated are challenging to verify. The work most closely related to ours is [Gonon, 2023], which provided a full analysis of random neural networks for learning sufficiently non-degenerate Black-Scholes type models. However, their approximation results are presented under the L^∞ norm, and their generalization analysis is grounded in methods for L^2 regression problems. Moreover, their methods cannot be applied to our setting, which involves unsupervised learning as opposed to the supervised problems they consider. To the best of our knowledge, this is the first article that attempts to provide theoretical support for solving PDEs within the framework of PINNs using random neural networks.

1.3 Notations

For $x \in \mathbb{R}^d$, $\|x\|_p$ denotes its p-norm ($1 \leq p \leq \infty$). For the activation functions, we write $\sigma_k(x)$ for the ReLU^k activation function, i.e., $\sigma_k(x) := [\max(0, x)]^k$. For given probability measure P and a sequence of random variables $\{X_i\}_{i=1}^n$ distributed according to P, we denote the empirical measure of P by P_n, i.e. $P_n = \frac{1}{n}\sum_{i=1}^{n} \delta_{X_i}$.

2 Preliminaries

In this section, we provide some preliminaries about random neural networks and Barron functions, which are pivotal to our study. Throughout the paper, we only consider the two-layer random neural networks, i.e., a feedforward neural network with one hidden layer and randomly generated hidden weights. For brevity, we still call them random neural networks.

To make it more precise, for $m \in \mathbb{N}$, let $W_1, \cdots, W_m$ be i.i.d. $\mathbb{R}^d$-valued random vectors and $B_1, \cdots, B_m$ be i.i.d. real-valued random variables, where $W = (W_1, \cdots, W_m)$ and $B = (B_1, \cdots, B_m)$ are independent. Then for any $\mathbb{R}^m$-valued (random) vector $A = (A_1, \cdots, A_m)$, we have the random neural network

$$H_A^{W,B}(x) := \sum_{i=1}^{m} A_i \sigma(W_i \cdot x + B_i), \tag{1}$$

where $\sigma : \mathbb{R} \to \mathbb{R}$ is a fixed activation function.

In addressing the L^2 regression with random neural networks, given training samples $\{(x_i, y_i)\}_{i=1}^n$, where x_i is the input and y_i is the output for $1 \le i \le n$, we have the following objective function to be optimized.

$$\sum_{i=1}^{n} \left(H_A^{W,B}(x_i) - y_i \right)^2. \tag{2}$$

This results in a least squares problem that has a closed-form solution. In contrast, deterministic neural networks also require optimizing the weights of the hidden layers, which leads to a more complex optimization problem.

In this work, we mainly focus on the Barron-type functions, leveraging their integral representations as highlighted in [Siegel and Xu, 2022]. The Barron space with order $s > 0$ is defined as

$$\begin{aligned} &\mathcal{B}^s(\Omega) := \{ f : \Omega \to \mathbb{C} : \\ &\|f\|_{\mathcal{B}^s(\Omega)} := \inf_{f_e|_\Omega = f} \int_{\mathbb{R}^d} (1 + \|w\|_1)^s |\hat{f}_e(w)| dw < \infty, \} \end{aligned} \tag{3}$$

where the infimum is over extensions $f_e \in L^1(\mathbb{R}^d)$ and $\hat{f}_e$ is the Fourier transform of f_e. Barron introduced this class for $s = 1$ and showed that two-layer neural networks with sigmoidal activation function can achieve the approximation rate $\mathcal{O}(1/\sqrt{n})$ in the L^2 norm. Although the convergence rate does not suffer the curse of dimensionality, the related optimization problems are non-convex and challenging to address. Consequently, we shift our focus toward approximating Barron functions employing random neural networks. Note that we choose 1-norm in the definition (3) just for simplicity. In the following, we assume that Ω is a subset of $[0, 1]^d$.

3 Main Results

3.1 Approximation Results

Our approach to approximating Barron functions by random two-layer neural networks leverages the integral representations of these functions, as presented in [Siegel and Xu, 2020b]. Specifically, for $f \in \mathcal{B}^s(\Omega)$, without loss of generality, assume that the infimum is attained at f_e. Then for activation function $\sigma \in L^1(\mathbb{R})$ with $\hat{\sigma}(a) \neq 0$ and some $a \neq 0$, we have

$$\begin{aligned} f(x) &= \int_{\mathbb{R}^d} e^{i\omega \cdot x} \hat{f}_e(\omega) d\omega \\ &= \int_{\mathbb{R}^d} \int_{\mathbb{R}} \frac{1}{2\pi\hat{\sigma}(a)} \sigma(\frac{\omega}{a} \cdot x + b) \hat{f}_e(\omega) e^{-iab} db d\omega \\ &= \int_{\mathbb{R}^d} \int_{\mathbb{R}} \frac{1}{2\pi\hat{\sigma}(a)} \sigma(\frac{\omega}{a} \cdot x + b) |\hat{f}_e(\omega)| \cos(\theta(\omega) - ab) db d\omega, \end{aligned} \tag{4}$$

where $\hat{f}_e(\omega) = e^{i\theta(\omega)} |\hat{f}_e(\omega)|$.

In the paper, we consider the activation function $\sigma(t) = \sum_{i=0}^{4} (-1)^i C_4^i \sigma_3(t + 2 - i)$, which is compactly supported on $[-2, 2]$. The conclusions can be naturally extended to sigmoidal activation functions and tanh activation functions, which are the commonly used smooth activation functions for PINNs. Specific details can be found in Remark 2. The approximation results rely on the polynomial decay condition of the activation function, i.e., $|\sigma^{(k)}(t)| \le C_p(1 + |t|)^{-p}$ for $0 \le k \le 2$ for some $p > 1$. This condition also appears in [Siegel and Xu, 2020b] for deriving the approximation rate for functions in $\mathcal{B}^s(\Omega)$ using two-layer deterministic neural networks.

Theorem 1. *Let P_1 be the uniform distribution of the domain $\{\omega \in \mathbb{R}^d : \|\omega\|_1 \le M\}$ and P_2 be the uniform distribution of the domain $\{b \in \mathbb{R} : |b| \le 2M\}$ with a constant $M \ge 2$. Let $W_1, \cdots, W_m \sim P_1$ and $B_1, \cdots, B_m \sim P_2$, then there exists a $\mathbb{R}^m$-valued, $\sigma(W, B)$-measurable vector $A = (A_1, \cdots, A_m)$ such that for $3 < s < 6$,*

$$\begin{aligned} &\mathbb{E}\left[\|f(x) - f_m(x)\|_{H^2(\Omega)}^2 \right] \lesssim \\ &\frac{M^{d+7-s}}{m} \|f\|_{\mathcal{B}^s(\Omega)} \|\hat{f}_e\|_{L^\infty(\mathbb{R}^d)} + \left(\frac{1}{M^{s-3}} \|f\|_{\mathcal{B}^s(\Omega)} \right)^2. \end{aligned} \tag{5}$$

and for $s \ge 6$,

$$\begin{aligned} &\mathbb{E}\left[\|f(x) - f_m(x)\|_{H^2(\Omega)}^2 \right] \lesssim \\ &\frac{M^{d+1}}{m} \|f\|_{\mathcal{B}^s(\Omega)} \|\hat{f}_e\|_{L^\infty(\mathbb{R}^d)} + \left(\frac{1}{M^{s-3}} \|f\|_{\mathcal{B}^s(\Omega)} \right)^2. \end{aligned} \tag{6}$$

where

$$f_m(x) = \frac{1}{m} \sum_{i=1}^{m} A_i \sigma(W_i \cdot x + B_i) \tag{7}$$

and

$$|A_i| \le C \|\hat{f}_e\|_{L^\infty(\mathbb{R}^d)} M^{d+1}, \tag{8}$$

$\lesssim$ indicates that a universal multiplicative constant is omitted.

Remark 1. *By appropriately selecting the value of M, we can obtain that*

$$\mathbb{E}\left[\|f(x) - f_m(x)\|_{H^2(\Omega)}^2 \right] \lesssim \begin{cases} m^{-\frac{2s-6}{d+s+1}}, & 3 < s < 6, \\ m^{-\frac{2s-6}{d+2s-5}}, & s \ge 6. \end{cases} \tag{9}$$

Remark 2. *Extending our discussion beyond the specific activation function we have chosen, we explore other common activation functions, such as the Sigmoidal function Sig(x) defined as $(1 + e^{-x})^{-1}$ and the Hyperbolic tangent function Tanh(x), expressed as $\frac{e^x - e^{-x}}{e^x + e^{-x}}$. Then we can construct new activation functions $\sigma(x) = Sig(x + 1) - Sig(x)$ and $\sigma(x) = Tanh(x + 1) - Tanh(x)$, which satisfy that*

$$|\sigma^{(k)}(t)| \le Ce^{-|t|}$$

for $0 \le k \le 2$, where C is a universal constant.

With this exponential decay condition, we can deduce from the proof of Theorem 1 that for $3 < s < 6$,

$$\begin{aligned} &\mathbb{E}\left[\|f(x) - f_m(x)\|_{H^2(\Omega)}^2 \right] \lesssim \\ &\frac{M^{d+7-s}}{m} \|f\|_{\mathcal{B}^s(\Omega)} \|\hat{f}_e\|_{L^\infty(\mathbb{R}^d)} + \left(\frac{1}{M^{s-3}} + \frac{1}{e^M} \right)^2 \|f\|_{\mathcal{B}^s(\Omega)}^2. \end{aligned} \tag{10}$$

for $s \geq 6$,

$$\mathbb{E}\left[\|f(x) - f_m(x)\|^2_{H^2(\Omega)}\right] \lesssim \frac{M^{d+1}}{m}\|f\|_{\mathcal{B}^s(\Omega)}\|\hat{f}_e\|_{L^\infty(\mathbb{R}^d)} + \left(\frac{1}{M^{s-3}} + \frac{1}{e^M}\right)^2 \|f\|^2_{\mathcal{B}^s(\Omega)}. \tag{11}$$

Besides these activation functions, Theorem 3 in [Siegel and Xu, 2020b] implies that, when $\sigma \in W^{m,\infty}(\mathbb{R})$ is a non-constant periodic function, f has a similar integral representation for σ. Moreover, in this case, we do not need to truncate b, which may yields a better approximation rate.

Proof Sketch: In the integral representations of functions within $\mathcal{B}^s(\Omega)$, the integral can be partitioned into two components, corresponding to bounded and unbounded regions, respectively. Specifically,

$$\begin{aligned} f(x) &= \int_{\mathbb{R}^d}\int_{\mathbb{R}} \frac{\sigma(\omega\cdot x+b)}{2\pi\hat{\sigma}(1)}|\hat{f}(\omega)|\cos(\theta(\omega)-b)dbd\omega \\ &= \int_A \frac{\sigma(\omega\cdot x+b)}{2\pi\hat{\sigma}(1)}|\hat{f}(\omega)|\cos(\theta(\omega)-b)dbd\omega \\ &\quad + \int_{A^c} \frac{\sigma(\omega\cdot x+b)}{2\pi\hat{\sigma}(1)}|\hat{f}(\omega)|\cos(\theta(\omega)-b)dbd\omega \\ &:= f_1(x) + f_2(x), \end{aligned} \tag{12}$$

where the domain $A := \{\|\omega\|_1 \leq M, |b| \leq 2M\}$ and we write $\hat{f}$ for $\hat{f}_e$ for simplicity. For the first part $f_1(x)$, it can be approximated by a random neural network. The second part, $f_2(x)$, can be further decomposed into two terms: one corresponding to the integral over $\{\|\omega\|_1 > M, b \in \mathbb{R}\}$ and another to the integral over $\{\|\omega\|_1 \leq M, |b| > 2M\}$. We denote these parts as $f_{21}(x)$ and $f_{22}(x)$, respectively. Given that the activation function σ is compactly supported in $[-2, 2]$, we can deduce that $\sigma(\omega\cdot x+b) = 0$ in $\{\|\omega\|_1 \leq M, |b| > 2M\}$, since

$$|\omega\cdot x+b| \geq |b| - |\omega\cdot x| \geq |b| - \|\omega\|_1\|x\|_\infty > M \geq 2,$$

which implies $f_{22}(x) = 0$.

Therefore, to ensure that $f(x)$ can be well-approximated well by the random neural network designed to approximate $f_1(x)$, it suffices to guarantee that $f_{21}(x)$ has a small enough H^2 norm. It is for this reason that we impose the polynomial decay condition on σ. With this condition, we can establish that

$$\int_{\mathbb{R}} |\sigma^{(k)}(\omega\cdot x+b)|^s db \lesssim 1 + \|\omega\|_1, \tag{13}$$

which is crucial to achieve our goal. The full proof can be found in the appendix of [Xu *et al.*, 2024].

Remark 3. *In [Gonon, 2023], the authors also considered employing a random neural network to approximate the function f belonging to $\mathcal{B}^s(\Omega)$, with W_i having a strictly positive Lebesgue-density π_w on $\mathbb{R}^d$ and B_i having a strictly positive Lebesgue-density π_b on $\mathbb{R}$. Despite achieving a dimension-independent approximation rate of $\mathcal{O}(1/\sqrt{m})$, the densities π_w and π_b are dependent on the unknown decay of the Fourier transform of f, leaving the prior distributions for W_i and B_i unclear. For general s, there may not be densities π_w and π_b that ensure the constants in the approximation rate are finite. Nonetheless, the approximation results in [Gonon, 2023] remain valid, because the target function they consider possesses a Fourier transform that decays exponentially. Moreover, the method employed in [Gonon, 2023] is specifically tailored for the ReLU activation function and the approximation results are given in the L^∞ norm, which renders it inapplicable to the context of PINNs.*

For the L^2 regression problems, as shown in [Braun et al.*, 2021], random neural networks can achieve a rate of $\mathcal{O}(1/\sqrt{n})$ for the final prediction error, where n is the sample size. This study assumes that the regression function f satisfies the condition*

$$|\hat{f}(\omega)| \leq \frac{c_1}{\|\omega\|_2^{d+1}(\log\|\omega\|_2)^2}, \tag{14}$$

for all $\omega \in \mathbb{R}^d$ with $\|\omega\|_2 \geq 2$ and some constant $c_1 > 0$. It is evident that this condition implies that

$$\int_{\mathbb{R}^d}(1+\|\omega\|_2)|\hat{f}(\omega)|d\omega < \infty. \tag{15}$$

Thus, the assumption made in [Braun et al.*, 2021] is stronger than the one in our study and may be more challenging to verify.*

Clearly, the approximation rates presented in Theorem 1 are affected by the curse of dimensionality, which is attributed to the unknown decay of the Fourier transform of the target function. When the smoothness parameter s is sufficiently large, we can choose not to truncate ω and b, which may lead to better results. Specifically, assume that $s > \alpha+\beta+5$ with $\alpha > d, \beta > 1$, then we can derive the following representation for $f \in \mathcal{B}^s(\Omega)$.

$$\begin{aligned} f(x) &= \int_{\mathbb{R}^d}\int_{\mathbb{R}} \frac{\sigma(\omega\cdot x+b)}{2\pi\hat{\sigma}(1)}|\hat{f}(\omega)|\cos(\theta(\omega)-b)dbd\omega \\ &= \int_{\mathbb{R}^d}\int_{\mathbb{R}} \frac{\sigma(\omega\cdot x+b)}{2\pi\hat{\sigma}(1)}|\hat{f}(\omega)|\cos(\theta(\omega)-b) \\ &\qquad \cdot \frac{(1+\|\omega\|_2)^\alpha(1+|b|)^\beta}{C_\alpha C_\beta}\frac{C_\alpha C_\beta}{(1+\|\omega\|_2)^\alpha(1+|b|)^\beta}dbd\omega \\ &= \mathbb{E}_{P_1(\omega),P_2(b)}\left[\frac{1}{2\pi\hat{\sigma}(1)}\frac{\sigma(\omega\cdot x+b)|\hat{f}(\omega)|\cos(\theta(\omega)-b)}{p_1(\omega)p_2(b)}\right], \end{aligned} \tag{16}$$

where $P_1(\omega) = p_1(\omega)d\omega, P_2(b) = p_2(b)db$ are probability measures with density functions $p_1(\omega)$ and $p_2(b)$ in $\mathbb{R}^d$ and $\mathbb{R}$, respectively, defined as $p_1(\omega) := \frac{C_\alpha}{(1+\|\omega\|_2)^\alpha}$ and $p_2(b) := \frac{C_\beta}{(1+|b|)^\beta}$.

Building upon our previous discussion, we now consider a scenario where the target function f belongs to a smaller function space. Specifically, we assume that f satisfies the following condition.

Condition 1. *Given function f, there is a function $f_e \in L^1(\mathbb{R}^d)$ such that $f_e|_\Omega = f$ and*

$$\int_{\mathbb{R}^d}(1+\|\omega\|_2)^s|\hat{f}_e(\omega)|^2 d\omega < \infty \tag{17}$$

with $s > \alpha+\beta+5$ and $\alpha > d+4, 1 < \beta < 5$.

Given the integral representation of the function f that satisfies Condition 1, i.e., (17), we can derive the following approximation results for f.

Theorem 2. *Let $W_1, \cdots, W_m \sim P_1$ and $B_1, \cdots, B_m \sim P_2$, then there exists a $\mathbb{R}^m$-valued, $\sigma(W,B)$-measurable vector $A = (A_1, \cdots, A_m)$ such that*

$$\mathbb{E}\left[\|f(x) - f_m(x)\|^2_{H^2(\Omega)}\right] \lesssim \frac{1}{m}\frac{d^{\beta+1}}{C_\alpha}, \tag{18}$$

where

$$f_m(x) = \frac{1}{m}\sum_{i=1}^{m} A_i \sigma(W_i \cdot x + B_i) \tag{19}$$

and

$$A_i = \frac{1}{2\pi\hat{\sigma}(1)}\frac{|\hat{f}(W_i)|\cos(\theta(W_i) - B_i)}{p_1(W_i)p_2(B_i)} I_{\{|B_i| \le Cd(1+\|W_i\|_2)\}} \tag{20}$$

for $1 \le i \le m$.

Remark 4. *Note that $\frac{1}{C_\alpha} \sim \frac{1}{\alpha - 1}\mathcal{H}^{d-1}(S^{d-1})$, i.e., the hypervolume of the $(d-1)$-dimensional unit sphere. Therefore, when d is sufficiently large, this term becomes small, and $\frac{d^{\beta+1}}{C_\alpha}$ does too.*

Remark 5. *Here, we aim to discuss the relationship between the class of functions we are considering and the Sobolev class of functions. Lemma 2.1 in [Xu, 2020] shows that for function u and any $\epsilon > 0$,*

$$\|u\|_{H^s(\Omega)} \lesssim \|u\|_{\mathcal{B}^s(\Omega)} \lesssim \|u\|_{H^{s+\frac{d}{2}+\epsilon}(\Omega)}.$$

This demonstrates that Sobolev spaces of sufficiently high order can be embedded into Barron spaces. Furthermore, the form of the definition for the functions considered in Condition 1 can be viewed as an equivalent definition of functions in $H^{s/2}(\Omega)$. Therefore, overall, these two cases are not very strict assumptions.

3.2 Optimization and Generalization Results

Within the framework of Physics-Informed Neural Networks (PINNs), we begin with the following elliptic partial differential equation, endowed with Dirichlet boundary conditions.

$$\begin{cases} -\Delta u(x) + V(x)u(x) = f(x), & in\ \Omega, \\ u(y) = g(y), & on\ \partial\Omega, \end{cases} \tag{21}$$

where $V, f, g \in L^\infty(\Omega)$.

Let u^* be the solution of the given equation (21), we consider two scenarios:

(1) u^* belongs to the Barron space $\mathcal{B}^s(\Omega)$;

(2) u^* satisfies Condition 1.

In the first scenario, for brevity and readability, we focus on the situation where s is sufficiently large. Specifically, we take s such that $2(s-3) \ge d+1$, i.e., $s \ge \frac{d+7}{2}$. For other values of s, as can be seen from the proof in the appendix of [Xu *et al.*, 2024], the process does not undergo significant changes; the only adjustment required is in the final step concerning the selection of values for m and λ.

According to Theorem 1, we can formulate a random neural network u given by the expression:

$$u(x) = \sum_{i=1}^{m} a_i \sigma(\omega_i \cdot x + b_i), \tag{22}$$

where $\omega_1, \cdots, \omega_m \sim P_1$, $b_1, \cdots, b_m \sim P_2$ (as defined in Theorem 1) and $a = (a_1, \cdots, a_m) \in \mathbb{R}^m$ is to be determined. For $1 \le i \le m$, we denote the k-th component of ω_i by $\omega_{i,k}$.

Given the form of the random neural network u, the loss function of PINNs can be written as

$$\begin{aligned} L(a) &:= \int_\Omega (-\Delta u + Vu - f)^2 dx + \int_{\partial\Omega}(u-g)^2 dy \\ &= \int_\Omega \Big(\sum_{i=1}^{m} a_i \Big(-\sum_{k}^{d} \omega_{i,k}^2 \sigma''(\omega_i \cdot x + b_i) + V(x)\sigma(\omega_i \cdot x + b_i)\Big) \\ &\quad - f(x)\Big)^2 dx + \int_{\partial\Omega}\Big(\sum_{i=1}^{m} a_i \sigma(\omega_i \cdot y + b_i) - g(y)\Big)^2 dy \\ &= \int_\Omega (a \cdot f_1(x) + g_1(x))^2 dx + \int_{\partial\Omega}(a \cdot f_2(y) + g_2(y))^2 dy, \end{aligned} \tag{23}$$

where $g_1(x) = -f(x), g_2(y) = -g(y)$ and the i-th components of $\mathbb{R}^m$-valued functions $f_1(x)$ and $f_2(y)$ are defined as

$$\begin{aligned} f_1^i(x) &= -\sum_{k=1}^{d} \omega_{i,k}^2 \sigma''(\omega_i \cdot x + b_i) + V(x)\sigma(\omega_i \cdot x + b_i) \\ &= |\omega_i|^2 \sigma''(\omega_i \cdot x + b_i) + V(x)\sigma(\omega_i \cdot x + b_i) \end{aligned} \tag{24}$$

and $f_2^i(y) = \sigma(\omega_i \cdot y + b_i)$, respectively.

To solve for a, we introduce a regularization term to the loss function $L(\cdot)$, resulting in the following objective function:

$$\begin{aligned} F(a) &:= L(a) + \lambda\|a\|_2^2 \\ &= \int_\Omega (a \cdot f_1(x) + g_1(x))^2 dx + \int_{\partial\Omega}(a \cdot f_2(y) + g_2(y))^2 dy \\ &\quad + \lambda\|a\|_2^2, \end{aligned} \tag{25}$$

where λ is a hyperparameter that controls the regularization strength and is to be determined later.

For simplicity, assuming an equal number of samples from the interior and the boundary, the empirical version of $F(a)$ can be expressed as

$$F_n(a) := \frac{1}{n}\sum_{i=1}^{n} l(a; x_i, y_i) + \lambda\|a\|_2^2, \tag{26}$$

where

$$\begin{aligned} &l(a; x_i, y_i) := \\ &\quad |\Omega|(a \cdot f_1(x_i) + g_1(x_i))^2 + |\partial\Omega|(a \cdot f_2(y_i) + g_2(y_i))^2. \end{aligned} \tag{27}$$

In the first case, according to Theorem 1, there exists a random neural network $u_m = \sum_{i=1}^{m} \bar{a}_i \sigma(\omega_i \cdot x + b_i)$ such that

$$\mathbb{E}\left[\|u(x) - u_m(x)\|^2_{H^2(\Omega)}\right] \lesssim \frac{1}{\sqrt{m}}, \tag{28}$$

where

$$\bar{a} = (\bar{a}_1, \cdots, \bar{a}_m),\ |\bar{a}_i| \lesssim 1/\sqrt{m} \tag{29}$$

for $1 \leq i \leq m$.

Note that $\|\bar{a}\|_2 = \sqrt{\sum_{i=1}^m \bar{a}_i^2} = \mathcal{O}(1)$, thus we can employ the projected gradient descent method for

$$a \in \mathcal{A} := \{a \in \mathbb{R}^m : \|a\|_2 \leq C\}, \tag{30}$$

where C is a constant ensuring that $\bar{a} \in \mathcal{A}$. Specifically, the method of projected gradient descent consists of iteration of the following update rule for $t = 1, \cdots, T$,

$$\begin{cases} y_t = a_t - \nu_t g_t, \quad where\ g_t \in \partial F_n(a_t) \\ a_{t+1} = P_{\mathcal{A}}(y_t), \end{cases} \tag{31}$$

where T is the total number of iterations, a_1 is an initial approximation and $P_{\mathcal{A}}$ is the projection operator onto the convex closed set $\mathcal{A}$. The step size ν_t is a positive scalar chosen according to a specific schedule or rule.

From the specific form of $F_n(\cdot)$, we can see that it is strongly convex and smooth on $\mathcal{A}$. In convex optimization, it is well-known that for a function f that is α-strongly convex and β-smooth on $\mathcal{A}$, the projected gradient descent with $\nu_t = 1/\beta$ ensures that for all non-negative integers t, the following inequality holds for the sequence $\{y_t\}$ generated by (31):

$$\|y_{t+1} - y^*\|_2^2 \leq \exp\left(-\frac{t}{\kappa}\right)\|y_1 - y^*\|_2^2, \tag{32}$$

where $\kappa = \beta/\alpha$.

The strong convexity of $F_n(\cdot)$ is crucial for the optimization process, ensuring that the algorithm converges to the optimal solution efficiently. Moreover, this property contributes to the generalization performance of the random neural networks, which can achieve the fast rate $\mathcal{O}(1/n)$.

Theorem 3. *Assume that $u^* \in \mathcal{B}^s(\Omega)$ with $s \geq \frac{d+7}{2}$, by applying the projected gradient descent in (31) with $\nu_t = \mathcal{O}(m^{-\frac{d+3}{d+1}})$ to the empirical loss function $F_n(\cdot)$, we have that*

$$\begin{aligned} &\mathbb{E}_{W,B}\mathbb{E}_{X,Y}[L(a_T; f_1, f_2, g_1, g_2)] \\ &\lesssim \frac{1}{\sqrt{m}} + \lambda + \frac{m^{3+\frac{6}{d+1}}}{\lambda^2} e^{-\frac{\lambda T}{m^{1+2/(d+1)}}} + \frac{m^{2+\frac{4}{d+1}}}{\lambda n}\log\log(mn), \end{aligned} \tag{33}$$

where

$$\begin{aligned} &L(a_T; f_1, f_2, g_1, g_2) := \\ &\int_\Omega (a_T \cdot f_1(x) + g_1(x))^2 dx + \int_{\partial\Omega} (a_T \cdot f_2(y) + g_2(y))^2 dy \end{aligned} \tag{34}$$

and a_T is obtained from (31) after T iterations.

Taking $\lambda = \frac{1}{\sqrt{m}}$, $m = n^{\frac{d+1}{3d+7}}$ and $T = \mathcal{O}(\sqrt{n}\log(n))$, we have

$$\mathbb{E}_{W,B}\mathbb{E}_{X,Y}[L(a_T; f_1, f_2, g_1, g_2)] = \mathcal{O}(n^{-\frac{d+1}{2(3d+7)}}\log\log(n)). \tag{35}$$

The first term represents the approximation error. The second term arises from the addition of the regularization term, which imposes strong convexity to the empirical loss function. This modification allows for the application of the projected gradient descent algorithm, ensuring rapid convergence. The third term corresponds to the optimization error, while the fourth term is the generalization error. The regularization term also contributes to the generalization error, leading to the fast rate $\mathcal{O}(1/n)$, which is attributed to the strong convexity of the empirical loss function and the utilization of the localization technique.

Remark 6. *In Theorem 3, we have not show the dependency on the dimension, which is an important factor. In fact, from the proof, the final bound provided in Theorem 1 depends quadratically on $|\Omega|$ and $|\partial\Omega|$. Furthermore, the bound is linear with respect to the constant involved in the Sobolev trace theorem, which is utilized for estimating the approximation error. The generalization error can also be quantified by an appropriate Sobolev norm. Specifically, as demonstrated in [Agmon* et al.*, 1959], we have*

$$\|u - u^*\|^2_{H^{\frac{1}{2}}(\Omega)} \lesssim \|-\Delta u + Vu - f\|^2_{L^2(\Omega)} + \|u - g\|^2_{L^2(\partial\Omega)}, \tag{36}$$

where u^* is the solution of equation (21).

Remark 7. *In [Klochkov and Zhivotovskiy, 2021], algorithm stability has been utilized to derive the excess risk for the projected gradient descent method applied to a loss function that is λ-strongly convex and L-Lipschitz continuous. It has been demonstrated that $T = \mathcal{O}(n^2)$ iterations are required to attain the excess risk $\tilde{\mathcal{O}}(\frac{L^2}{\lambda n})$. It is important to note that our approach to deriving the generalization bound relies solely on the strong convexity and Lipschitz continuity of the loss function. Under the same conditions as in [Klochkov and Zhivotovskiy, 2021], by employing the projected gradient descent method with $\nu_t = \frac{2}{\lambda(t+1)}$ and $a_T = \frac{2}{T(T+1)}\sum_{t=1}^T t a_t$, our method can achieve the same excess risk within $T = \mathcal{O}(n)$ steps. Additionally, because the loss function in our setting is also smooth, $T = \mathcal{O}(\sqrt{n}\log(n))$ is sufficient to reach the same excess risk.*

In the second case, because the randomly sampled hidden weights are no longer bounded, the loss function can become unbounded. Consequently, projected gradient descent is not suitable for this optimization problem, as the output weights of the random neural networks need to be bounded to achieve the approximation results stated in Theorem 2. Moreover, standard gradient descent is also unsuitable. Although the empirical loss function is convex, it lacks Lipschitz continuity, which means there is no guarantee of convergence for gradient descent. Instead, we can directly apply the closed form of ridge regression. According to the following theorem, we can choose $m = n^{1/4}$, which provides an acceptable level of computational complexity for employing closed-form solutions.

Theorem 4. *Assume that u^* satisfies Condition 1, and let $a_n = \arg\min_{a \in \mathbb{R}^m} F_n(a)$, then we can deduce that*

$$\mathbb{E}_{W,B}\mathbb{E}_{X,Y}[L(a_n; f_1, f_2, g_1, g_2)] \lesssim \frac{1}{m} + \frac{\lambda}{m} + \frac{m}{\lambda\sqrt{n}}, \tag{37}$$

where

$$L(a_n; f_1, f_2, g_1, g_2) := \int_\Omega (a_n \cdot f_1(x) + g_1(x))^2 dx + \int_{\partial\Omega} (a_n \cdot f_2(y) + g_2(y))^2 dy. \tag{38}$$

By taking $\lambda = \mathcal{O}(1), m = n^{1/4}$*, we have*

$$\mathbb{E}_{W,B}\mathbb{E}_{X,Y}[L(a_n; f_1, f_2, g_1, g_2)] = \mathcal{O}(n^{-\frac{1}{4}}). \tag{39}$$

Moreover, a_n *can be solved by using the closed form of the ridge regression and the complexity is* $\mathcal{O}(m^2 n) = \mathcal{O}(n^{\frac{3}{2}})$.

Remark 8. *The error components are distinctly categorized: the first represents the approximation error, the second is attributed to regularization, and the third captures the generalization error. In Theorem 4, an optimization error is notably absent, contrasting with Theorem 3, due to the implementation of the closed-form solution from ridge regression. Furthermore, the unbounded nature of the loss function makes it impossible to to obtain a fast rate for generalization error.*

4 Experimental Results

4.1 Verification of Theorem

Theorem 4 shows if $m = n^{\frac{1}{4}}$, where m is the hidden width and n is the collocation points number, then the testing loss is expected to be $\mathcal{O}(\frac{1}{m})$. We demonstrate this result from a simple 1D Poisson equation.

The hidden layer's width ranges from 50 to 500, with $tanh$ activations. In Figure 1, we can see that the estimated physics-informed loss decays as the width m increases. The decay trend is consistent with the theoretical decay $\mathcal{O}(\frac{1}{m})$, and the fluctuations may come from the different random initialization of the first layer.

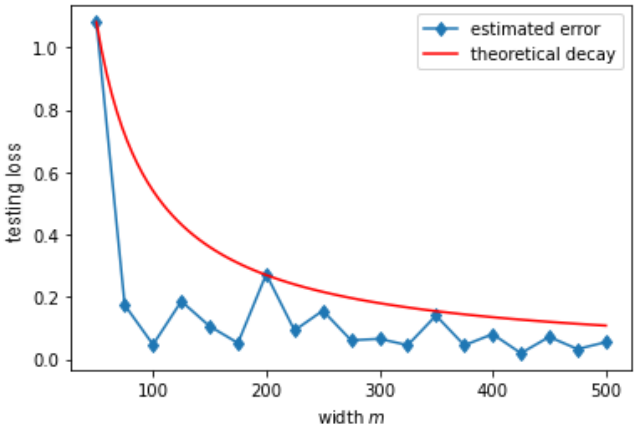

Figure 1: Testing loss for the Poisson equation. A random neural network with m hidden nodes (m ranges from 50 to 500) is used to solve 1D Poisson equation with PINN method. The dots show the estimated loss $L(m)$, the line shows the decay implied by the theoretical results $\frac{50L(50)}{m}$.

4.2 Application

We apply the random neural network to solve the Reaction-Diffusion (RD) equation, which is significant to nonlinear physics, chemistry, and developmental biology. We consider the spatial-temporal Reaction-Diffusion equation with the following form:

$$\begin{cases} u_t = d_1 u_{xx} + d_2 u^2, \quad t \in [0,1], x \in [-1,1], \\ u(0,x) = \sin(2\pi x)(1+\cos(2\pi x)), \\ u(t,-1) = u(t,1) = 0, \end{cases}$$

where $d_1 = d_2 = 0.01$.

We choose $N_b = 300$ randomly sampled points on the initial domain and the boundary domain, and $N_f = 5,000$ randomly sampled points within the domain $\Omega = [0,1] \times [-1,1]$. A random neural network with 4 hidden layers, each containing 128 units with $\tanh$ activation functions, is used for all computations. The training process runs for $10,000$ epochs with a learning rate of $lr = 1e-3$. The relative L_2 error is 0.13%. As shown in Figure 2, the random neural network method achieves high accuracy. The training time for conventional PINN method is 1397s in [Li *et al.*, 2024], while the training time for random neural network is 3.2s. This highlights the high efficiency and precision of our approach.

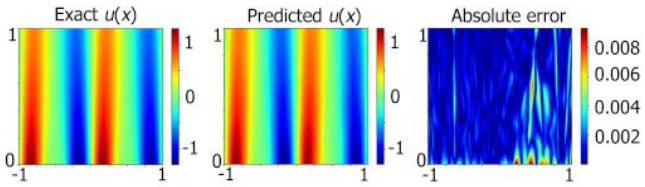

Figure 2: Comparison between the reference and predicted solutions for the Reaction-Diffusion equation.

5 Conclusion and Discussion

In this paper, we have established approximation results for Barron-type functions using random neural networks. Our approach diverges from prior studies by providing precise forms for the sampling distributions of hidden weights, rather than relying on empirical selection. Within the PINNs framework, applying random neural networks to solve PDEs translates into formulating regularized least square problems. We address these problems by employing projected gradient descent for one case and leveraging the closed-form solutions for regularized least squares in the other. Combining with the approximation results, we present a comprehensive error analysis. Nevertheless, this work has its limitations. For example, the approximation rate presented in Theorem 1 for functions in $\mathcal{B}^s(\Omega)$ suffers the curse of dimensionality. Moreover, both approximation results in Theorem 1 and Theorem 2 are only valid for relatively small function spaces. Expanding the approximation capabilities of random neural networks to more general functions presents an interesting direction for future research. Furthermore, the methods we employed for deriving the optimization and generalization errors could be extended to other methods involving random neural networks for solving PDEs, such as the Deep Ritz Method.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (No. 12025104, No. 62106103,), and the basic research project (ILF240021A24). This work is also partially supported by High Performance Computing Platform of Nanjing University of Aeronautics and Astronautics.

References

[Agmon *et al.*, 1959] Shmuel Agmon, Avron Douglis, and Louis Nirenberg. Estimates near the boundary for solutions of elliptic partial differential equations satisfying general boundary conditions. i. *Communications on pure and applied mathematics*, 12(4):623–727, 1959.

[Barron, 1993] Andrew R Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information theory*, 39(3):930–945, 1993.

[Belomestny *et al.*, 2023] Denis Belomestny, Alexey Naumov, Nikita Puchkin, and Sergey Samsonov. Simultaneous approximation of a smooth function and its derivatives by deep neural networks with piecewise-polynomial activations. *Neural Networks*, 161:242–253, 2023.

[Braun *et al.*, 2021] Alina Braun, Michael Kohler, Sophie Langer, and Harro Walk. The smoking gun: Statistical theory improves neural network estimates. 2021.

[Devlin, 2018] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[Dong and Li, 2021] Suchuan Dong and Zongwei Li. Local extreme learning machines and domain decomposition for solving linear and nonlinear partial differential equations. *Computer Methods in Applied Mechanics and Engineering*, 387:114129, 2021.

[Gonon *et al.*, 2023] Lukas Gonon, Lyudmila Grigoryeva, and Juan-Pablo Ortega. Approximation bounds for random neural networks and reservoir systems. *The Annals of Applied Probability*, 33(1):28–69, 2023.

[Gonon, 2023] Lukas Gonon. Random feature neural networks learn black-scholes type pdes without curse of dimensionality. *Journal of Machine Learning Research*, 24(189):1–51, 2023.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[Huang *et al.*, 2006] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: theory and applications. *Neurocomputing*, 70(1-3):489–501, 2006.

[Huang *et al.*, 2011] Guang-Bin Huang, Hongming Zhou, Xiaojian Ding, and Rui Zhang. Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(2):513–529, 2011.

[Klochkov and Zhivotovskiy, 2021] Yegor Klochkov and Nikita Zhivotovskiy. Stability and deviation optimal risk bounds with convergence rate $o(1/n)$. *Advances in Neural Information Processing Systems*, 34:5065–5076, 2021.

[Krishnapriyan *et al.*, 2021] Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael W Mahoney. Characterizing possible failure modes in physics-informed neural networks. *Advances in Neural Information Processing Systems*, 34:26548–26560, 2021.

[Li *et al.*, 2024] Ye Li, Siqi Chen, Bin Shan, and Sheng-Jun Huang. Causality-enhanced discreted physics-informed neural networks for predicting evolutionary equations. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 4497–4505, 2024.

[Lu *et al.*, 2021a] Jianfeng Lu, Zuowei Shen, Haizhao Yang, and Shijun Zhang. Deep network approximation for smooth functions. *SIAM Journal on Mathematical Analysis*, 53(5):5465–5506, 2021.

[Lu *et al.*, 2021b] Yulong Lu, Jianfeng Lu, and Min 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.

[Neufeld and Schmocker, 2023] Ariel Neufeld and Philipp Schmocker. Universal approximation property of random neural networks. *arXiv preprint arXiv:2312.08410*, 2023.

[Shen *et al.*, 2022] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Optimal approximation rate of relu networks in terms of width and depth. *Journal de Mathématiques Pures et Appliquées*, 157:101–135, 2022.

[Siegel and Xu, 2020a] Jonathan W Siegel and Jinchao Xu. Approximation rates for neural networks with general activation functions. *Neural Networks*, 128:313–321, 2020.

[Siegel and Xu, 2020b] Jonathan W Siegel and Jinchao Xu. Approximation rates for neural networks with general activation functions. *Neural Networks*, 128:313–321, 2020.

[Siegel and Xu, 2022] Jonathan W Siegel and Jinchao Xu. High-order approximation rates for shallow neural networks with cosine and reluk activation functions. *Applied and Computational Harmonic Analysis*, 58:1–26, 2022.

[Silver *et al.*, 2016] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[Wang and Dong, 2024] Yiran Wang and Suchuan Dong. An extreme learning machine-based method for computational pdes in higher dimensions. *Computer Methods in Applied Mechanics and Engineering*, 418:116578, 2024.

[Xu *et al.*, 2024] Xianliang Xu, Ye Li, and Zhongyi Huang. A priori estimation of the approximation, optimization and generalization errors of random neural networks for solving partial differential equations. *arXiv preprint arXiv:2406.03080*, 2024.

[Xu, 2020] Jinchao Xu. The finite neuron method and convergence analysis. *arXiv preprint arXiv:2010.01458*, 2020.

[Yang *et al.*, 2023a] Yahong Yang, Yue Wu, Haizhao Yang, and Yang Xiang. Nearly optimal approximation rates for deep super relu networks on sobolev spaces. *arXiv preprint arXiv:2310.10766*, 2023.

[Yang *et al.*, 2023b] Yahong Yang, Haizhao Yang, and Yang Xiang. Nearly optimal vc-dimension and pseudo-dimension bounds for deep neural network derivatives. *Advances in Neural Information Processing Systems*, 36:21721–21756, 2023.

[Yarotsky, 2017] Dmitry Yarotsky. Error bounds for approximations with deep relu networks. *Neural networks*, 94:103–114, 2017.