Commun. Comput. Phys.
doi: 10.4208/cicp.OA-2023-0253

Vol. **35**, No. 3, pp. 761-815
March 2024

Generalization Error in the Deep Ritz Method with Smooth Activation Functions

Janne Siipola*

Department of Mathematics and Statistics, University of Helsinki, Finland.

Received 27 September 2023; Accepted (in revised version) 12 March 2024

Abstract. Deep Ritz method is a deep learning paradigm to solve partial differential equations. In this article we study the generalization error of the Deep Ritz method. We focus on the quintessential problem which is the Poisson's equation. We show that generalization error of the Deep Ritz method converges to zero with rate $\frac{C}{\sqrt{n}}$, and we discuss about the constant C. Results are obtained for shallow and residual neural networks with smooth activation functions.

AMS subject classifications: 35A35, 65C05, 92B20, 68T10

Key words: Deep learning, Deep Ritz method, Poisson's equation, residual neural networks, shallow neural networks, generalization.

1 Introduction

We are studying the Deep Ritz method [1], in particular the problem of approximating the solution of the Poisson's equation in a ball with deep neural networks [2, 3]; the results given in this article can easily be generalized for more complex domains, what is important is that the domain is bounded. Let us give a few definitions and remarks to frame what we are talking about here.

1.1 Deep Ritz Method

Let $\Omega \subset \mathbb{R}^d$ be a bounded domain with a smooth boundary, where $d \in \mathbb{N}$ is the dimension. Let's consider the Poisson's equation

$$-\Delta u = f,$$

*Corresponding author. *Email address:* janne.siipola@helsinki.fi (J. Siipola)

where f is some function, possibly $f \in L^2(\Omega)$, sometimes referred as the source term. We can consider this equation with Dirichlet boundary condition as

$$\begin{cases} -\Delta u = f & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega. \end{cases} \tag{1.1}$$

The Poisson's equation can also be stated in a variational form as

$$\min_{u \in H_0^1(\mathbb{R}^d)} \int_\Omega \frac{1}{2}|\nabla u|^2 - fu, \tag{1.2}$$

meaning that a minimizer of the integral in (1.2), among a set of admissible functions $H_0^1(\mathbb{R}^d)$, is the solution of the Poisson's equation. Here a natural choice for the set of admissible functions is the Sobolev space $H_0^1(\Omega)$. This variational form is what was proposed for a loss function in [1], known as the Deep Ritz method (DRM). However, in DRM we cannot minimize the integral over some Sobolev space – the set of admissible functions will be some set of neural networks.

When the set of admissible functions is some set of neural networks, then the zero boundary condition might not be met, and one must add a penalty term in order to force the solution to meet the boundary condition. Therefore the loss function for the Deep Ritz method will be

$$\min_{u \in \mathcal{H}} \int_\Omega \left(\frac{1}{2}|\nabla u|^2 - fu \right) + \lambda \int_{\partial\Omega} u^2, \tag{1.3}$$

where the penalty term $\lambda > 0$ is a constant. The minimal solution for (1.3) must take into account the boundary values as well, and so this is the actual theoretical loss function for the DRM. The latter variational form is related to another boundary value problem for the Poisson's equation, namely the Robin boundary value problem

$$\begin{cases} -\Delta u = f & \text{in } \Omega, \\ u + \frac{1}{2\lambda}\partial_\nu u = 0 & \text{on } \partial\Omega, \end{cases} \tag{1.4}$$

where $\partial_\nu u$ is the derivative of u into direction of the normal of the boundary $\partial\Omega$. These two boundary value problems are connected with the parameter λ, and it is a known result that if we let $\lambda \to \infty$ then the solution of the Robin problem converges to the solution of the Dirichlet problem [4].

The theoretical loss function (1.3) can be represented as

$$\begin{aligned} \mathcal{L}_\lambda(u) = |\Omega| \mathbb{E}_{X \sim U(\Omega)} \left[\frac{\|\nabla u(X)\|_2^2}{2} - u(X) f(X) \right] \\ + \frac{\lambda}{2} |\partial\Omega| \mathbb{E}_{Y \sim U(\partial\Omega)} [T(u(Y)^2)], \end{aligned} \tag{1.5}$$

where $U(\Omega)$ and $U(\partial\Omega)$ are uniform distributions on the respective sets, $|\cdot|$ denotes the Hausdorff measure of the respective sets, and T is the trace operator defined on the boundary of Ω. An actual loss function, which can be computed, is a discretized version of it

$$\widehat{\mathcal{L}}_\lambda(u,n,m) = \frac{|\Omega|}{n}\sum_{i=1}^{n}\left[\frac{\|\nabla u(X_i)\|_2^2}{2} - u(X_i)f(X_i)\right] + \frac{\lambda}{2}\frac{|\partial\Omega|}{m}\sum_{j=1}^{m}[T(u(Y_j)^2)], \tag{1.6}$$

where $X_i \sim U(\Omega)$ for all $i \in \{1,\cdots,n\}$, and $Y_j \sim U(\partial\Omega)$ for all $j \in \{1,\cdots,m\}$. Therefore one can use the loss function (1.6) to optimize a neural network to approximate the solution of the Poisson's equation. In the error analysis, we simplify the setting by assuming that the source function f is constant 1.

1.2 Error analysis

Let the solution of the Poisson's equation be u and let $\widehat{u}$ be a neural network approximation for the Poisson's equation as the loss function $\widehat{\mathcal{L}}_\lambda(u,n,m)$, see equation (1.6), is minimized by some stochastic gradient decent (SGD) algorithm; the question becomes, how the following difference behaves

$$\|u-\widehat{u}\|, \tag{1.7}$$

with some suitable norm; the norm is of course another problem, the Sobolev norms [5] and the Barron norm [6] are among possible options. Heuristically speaking, we can split this error into four different terms: penalization error, approximation error, statistical error, and optimization error. The statistical error and the optimization error forming together the generalization error [7], which does not stand out from the four error terms.

Anyway, here is a short idea behind the four error terms: the penalization error is due to the fact that we are aiming for the solution of (1.2) with $\mathcal{H} = H_0^1(\mathbb{R})$, but instead we are using penalized version (1.3) of the loss function [4]. Convergence estimate for the boundary penalty is well known, see for example [8, 9], or [10] where it is shown in Theorem 4 that with respect to the Sobolev norm H^1 the error converges in $\mathcal{O}(\lambda^{-1})$ as $\lambda \to \infty$.

The approximation error is due to the fact that the set of admissible functions, which is some set of neural nets, might not contain the solution to the loss function, and this causes an error between the target function and the approximation we get by using some neural network architecture. There is a long tradition on studies in this field [11]. We show in the Appendix D that in our problem the solution belongs to the Barron space. This space is suitable to approximate with shallow and residual neural networks as it is

shown that Barron functions can be approximated with error estimate

$$\mathbb{E}_x[(f^*(x)-f(x;\theta))^2] \leq \frac{3\gamma_2^2(f^*)}{m},$$
$$\|\theta\|_{\mathcal{P}} \leq 2\gamma_2(f^*),$$

where $f^*(x)$ is the target function and $f(x,\theta)$ is a shallow neural network and m is the width of the shallow network. For more information see for example [12].

The statistical error measures the difference between solutions of (1.5) and (1.6), and so it is a measure of the difference between the solutions of the mean of the loss function and its sample mean with limited amount of observations. Sometimes the statistical error is also referred to as the generalization error, it is correct in a sense that it measures the ability to model data which the algorithm has not seen while training, i.e. how well it learns the underlying distribution, but it is not the same as the observed generalization error of the optimized neural network as the optimization algorithm also affects the generalization error.

In this article we give an estimate for the statistical error

$$|\mathcal{L}_\lambda(u)-\widehat{\mathcal{L}}_\lambda(u,n,m)|,$$

where u is either a shallow or a residual neural network, see Theorems 2.4, and 2.4. The main theorem implies that the statistical error converges to zero in $\mathcal{O}(\frac{1}{\sqrt{n}})$, where n is the number of points used in the Monte Carlo integration. All the constants in the estimate can be calculated. It gives information about the ability of shallow and residual networks to generalize in this PDE problem. We give these results for networks with smoothed rectified linear unit (Relu) activation functions which can be used in practice to minimize the loss function.

For example [10, 13–15] have studied the generalization error for Deep Ritz method, but failed to show the optimal $\frac{C}{\sqrt{n}}$ bound. In article [6] the optimal convergence rate is found for two-layer networks, with Relu and softplus activation functions, in the context of Barron space. Our result is different in that in applies also to residual neural networks and for wide range of activation functions.

In [16] the optimal convergence rate is found for PINNs [17] (or residual minimization). For PINNs, there are more related studies in for example [18–21].

Also [22], and [23] has studied generalization error within the context of PDEs. In [23] the authors study a cumulative loss function and enable take SGD into account to the generalization error, they obtain the optimal convergence rate.

We present the theory and our results in Section 2, Results. More about related results of the statistical error in Section 3, Literature and Discussion.

2 Results

In this article a letter[†] $\mathcal{F}$ has in some sense double meaning (which in the end is the same meaning), it refers to the architecture of a neural net, and by fixing the weights and the biases of the neural net architecture we define a function, mapping possibly from some vector space to another, and so as we have just defined the architecture and not the weights, we have in fact a collection of functions; there are the two meanings. Therefore we may refer to $\mathcal{F}$ as an architecture, but also as a family of neural nets (functions). Then again, if we are talking about minimizing a loss function, then $\mathcal{F}$ might be referred to a set of admissible functions. The architectures which are in our primary interest are residual neural networks (resnet) [24, 25], and two-layer neural networks, also known as shallow neural networks.

Outline of the proof of the main theorem is as follows. We bound the difference between the two loss functions with the help of the Rademacher complexity, and then use some of the latest results on bounding the Rademacher complexity of neural networks, to arrive to the result we have. Many results have been shown in the same fashion in machine learning[‡] and deep learning, two excellent books to mention here are [26, 27].

Notation used in this article: $\Omega := B(0,1)$ is the unit ball. $|\Omega|$ is the Lebesgue measure of the domain, and $|\partial\Omega|$ is the Hausdorff measure of the boundary. σ denotes an activation function. $\gamma(\sigma)$ is defined in Definition 2.1. Curly letter $\mathcal{F}$, $\mathcal{H}$ denote a set of functions and/or a neural network architecture, and with subindex Q as $\mathcal{F}_Q$ we denote a set of neural networks with architecture $\mathcal{F}$ with path norm bounded by Q. Curly $\mathcal{P}$ is used to denote the path norms as $\|\cdot\|_{\mathcal{P}}$. With $\widehat{\mathscr{R}}(\mathcal{F}_Q)$ and $\widehat{\mathscr{R}}_S(\mathcal{F}_Q,n)$ we denote empirical Rademacher complexity. With symbol $x\in\Omega$ we denote the space variable, and θ denotes the weights of a neural network. A neural network is usually denoted by letter u, $u(x,\theta)$. With $\vee$ we denote the maximum: $x\vee y := \max(x,y)$.

2.1 Activation functions

In the case of the loss function (1.6), a trial function should have differentiability at least for 2nd order, as the loss function consumes one derivative and the optimization algorithm second. The Relu activation function does not have even the first order derivative in the classical sense, as it is not differentiable at the origin. This lack of differentiability has not blocked the use of the Relu in applications and it has become a popular choice in many neural net architectures. We assume that one reason behind this is the fact that in those applications a loss function does not use the gradient of the neural net, or higher order derivatives, but just the training algorithm takes one order of differentiability and therefore the problems of the lack of differentiability do not cause too much problems.

[†]Some times some curly version of H might be used maybe because it refers to *hypothesis class*, or letter N because it refers to a term *neural network*.

[‡]Also referred to as statistical learning theory or neural network learning, with its own special meanings of course.

DOI https://doi.org/10.4208/cicp.OA-2023-0253 | Generated on 2024-12-19 11:09:00

Also Relu might have some other positive benefits, for example the fact that in the positive side Relu gives always constant amount of derivative, which does not flatten like in the tanh activation function.

This is possibly the reason for the fact that in our numerical experiments we were not able to train successfully a neural network with the Relu activation function for the loss function (1.6). But when using residual neural networks, it is better to use an activation function which is close to Relu. So, an activation function should be smooth and similar to ReLu. With smooth activation functions we were able to minimize the loss function and approximate the target function. We have moved the discussion of activation functions into the Appendix C.

For these reasons, we are aiming for theoretical bounds for neural network architectures with smoothed Relu activation functions. There is some theoretical tools existing for neural networks with Relu activation function, and sometimes these results can be proved to work also with the architectures with smoothed Relus. In [28] the authors state that under certain conditions, an activation function can be approximated by shallow networks with the Relu activation functions with bounded path norm. Let's group the conditions under one definition.

Definition 2.1. Let $\sigma:\mathbb{R}\to\mathbb{R}$ be an activation function. Let's define γ_0, g, and γ:

$$\gamma_0(\sigma)=\int_{\mathbb{R}}|\sigma''(x)|(|x|+1)dx, \tag{2.1}$$

$$g(x)=|\sigma(x)|+(|x|+2)|\sigma'(x)|, \tag{2.2}$$

$$\gamma(\sigma)=\gamma_0(\sigma)+\inf_{x\in\mathbb{R}}g(x). \tag{2.3}$$

Let the function σ satisfy the following conditions.

1. σ is continuous.
2. σ is twice weakly differentiable on $\mathbb{R}$.
3. The second derivative σ'' is locally Riemann integrable on $\mathbb{R}$.
4. $\gamma(\sigma)<\infty$.

If such conditions are met, we call the activation function *approachable*.

Next theorem justifies what was previously said about the approximation ability by the neural networks with the Relu activation function [28].

Theorem 2.1. *If an activation function σ is approachable, meaning it satisfies the conditions of the Definition* 2.1*, then for any $\delta>0$, there exists a two-layer neural network with the Relu activation function $u_m(\cdot,\theta)$ of width $m<\infty$ such that*

$$\sup_{x\in\mathbb{R}}|\sigma(x)-u_m(x,\theta)|\leq\delta, \tag{2.4}$$

$$\|\theta\|_{\mathcal{P}}\leq\gamma(\sigma)+\delta. \tag{2.5}$$

Proof. See the proof in [28]. □

The Theorem above means that an activation function which satisfies the conditions of the Definition 2.1 can be approximated by a two-layer neural network with Relu activation function with any precision, in such a way that the path-norm of the approximating neural network is bounded, and the bound is connected to the activation function itself by the value of $\gamma(\sigma)$. For the proof of the theorem see the source [28].

2.2 Architectures and path norms

Shallow, or two-layer, neural networks are the following functions.

Definition 2.2. Let $\sigma:\mathbb{R}\to\mathbb{R}$ be a nonlinear activation function. Let $m\in\mathbb{N}$. Let $x\in\mathbb{R}^d$ be the input layer, $\omega_i\in\mathbb{R}^d$, and $a_i\in\mathbb{R}$, $i=1,\cdots,m$ are the weights. A shallow neural network is the function

$$f(x;\theta)=\sum_{i=1}^{m}a_i\sigma(\langle\omega_i,x\rangle),$$

where θ denotes the collection of parameters, the weights, $\theta=\{a_i,\omega_i:i=1,\cdots,m\}$.

Remark 2.1. If the activation function is specified, then we can add this information into the name of the architecture: for example, if the activation function is the Relu, the architecture is called a shallow neural network with the Relu activation function.

In the Definition above at first glance it seems that it does not include the bias terms, but they are included in this Definition if we let the input layer contain one 'additional' dimension, and the value of the input vector's last coordinate is always 1.

For shallow networks with the Relu activation function we have a path norm [12,29]. It measures how much different paths of the information flow can affect the output of a shallow neural network.

Definition 2.3 (Path norm)**.** Let $u(x,\theta)$ be a shallow neural net with width m and the parameters $\theta=\{a_i,\omega_i:i=1,\cdots,m\}$, and with the Relu activation function. A path norm of the shallow neural network is

$$\|\theta\|_{\mathcal{P}}=\sum_{i=1}^{m}|a_i|\|\omega_i\|_1.$$

For smoothed versions of the Relu activation function we need a little bit different version of the path norm [28].

Definition 2.4. [Another path norm] Let σ be an approachable activation function, and let $u(x,\theta)$ be a shallow neural network with the activation function σ. Then the path norm $\|\cdot\|_{\tilde{\mathcal{P}}}$ is

$$\|\theta\|_{\tilde{\mathcal{P}}}:=\sum_{k=1}^{m}|a_k|\big(\|b_k\|_1+|c_k|+1\big),\tag{2.6}$$

where c_k is the bias term, and a_k, b_k are the weights of the neural network $u(x,\theta)$. The norm $\|\cdot\|_1$ is the l_1-norm on the vector space $\mathbb{R}^d$.

The latter path norm bounds the former path norm.

Let the structure of the residual neural net (resnet) be the following.

Definition 2.5 (Resnet)**.** Let the hidden layers h_l and g_l, and the output layer $u(x;\theta)$ have the following relation with a skip connection:

$$\begin{cases} h_0 = Vx, \\ g_l = \sigma(W_l h_{l-1}), \\ h_l = h_{l-1} + U_l g_l, \\ f(x;\theta) = a^T h_L, \end{cases} \tag{2.7}$$

where $l=1,\cdots,L$, σ is an activation function, $V\in\mathbb{R}^{D\times d}$, $W_l\in\mathbb{R}^{m\times D}$ and $U_l\in\mathbb{R}^{D\times m}$, $a\in\mathbb{R}^D$. θ denotes the set of parameters, $\theta=\{u,V,W_l,U_l : l=1,\cdots,L\}$. $u(x,\theta)\in\mathbb{R}$ is the output (layer) of the neural network.

We have a path norm also for residual neural networks with the Relu activation function [29].

Definition 2.6 (Weighted path norm)**.** Given a residual neural network of $u(x;\theta)$, define the weighted path norm of u as

$$\|\theta\|_P = \big\| |a|^T (I+2|U_L||W_L|)\cdots(I+2|U_1||W_1|)2|V| \big\|_1, \tag{2.8}$$

where $|A|$, with A being a vector or a matrix, means taking the absolute values of all the entries of the vector or the matrix.

In [28] authors came up with an alternative path norm for residual neural networks with approachable activation functions.

Definition 2.7. For any residual neural network $u(x,\theta)$ with an approachable activation function defined as in Definition 2.5, let its path norm be

$$\begin{aligned} \|\theta\|_{\tilde{\mathcal{P}}} = \sum_{i=0}^{L} \big\| |a|^T (I+c_\sigma|U_L||W_L|)(I+c_\sigma|U_{L-1}||W_{L-1}|) \\ \cdots (I+c_\sigma|U_{i+1}||W_{i+1}|)|U_i| \big\|_1. \end{aligned}$$

Here $U_0=V$, and $c_\sigma>4\gamma(\sigma)+1$ is a constant only related to the activation function σ.

Remark 2.2. We denote with $\mathcal{F}_Q$ a collection of neural networks with a bounded path norm, where the path norm is bounded by a constant $Q>0$.

Lemma 2.1. *Let $u(x,\theta)\in\mathcal{F}_Q$ be a shallow, or residual neural net, let $\|\theta\|_{\tilde{\mathcal{P}}}<Q$, where $Q>0$ is some positive real number. Then we have $\|\theta\|_{\mathcal{P}}<\|\theta\|_{\tilde{\mathcal{P}}}$, and in particular $\|\theta\|_{\mathcal{P}}<Q$.*

Proof. See the proof of Lemmas A.1, and A.2 in the appendix. □

2.3 Rademacher complexity

Now let's leave path norms for a while and talk about the Rademacher complexity a little bit. If $\mathcal{H}$ is a set of functions, and S is a set of points, such that functions in $\mathcal{H}$ map from S to $\mathbb{R}$, then let $\mathcal{H}\circ S$ be the set of all possible evaluations a function $h\in\mathcal{H}$ can have on a sample S, namely,

$$\mathcal{H}\circ S=\{h(z):h\in\mathcal{H},\, z\in S\}.$$

Rademacher complexity is defined as follows.

Definition 2.8. (Empirical Rademacher complexity) Let $\mathcal{F}$ be a collection of maps from a domain $\Omega\subset\mathbb{R}^d$ to $\mathbb{R}$, $\mathcal{F}:=\{f|f:\Omega\to\mathbb{R}\}$. Let $S=(z_1,\cdots,z_m)$ be a sample of points such that $z_i\in\Omega$ for all i. Let ξ_i be an independent uniform random variable taking value in $\{-1,1\}$. The empirical Rademacher complexity of the function class $\mathcal{F}$ is

$$\widehat{\mathcal{R}}_S(\mathcal{F},m)=\frac{1}{m}\mathbb{E}_\xi\Big[\sup_{f\in\mathcal{F}}\sum_{i=1}^m\xi_i f(z_i)\Big].$$

Remark 2.3. In the literature, Rademacher complexity might refer to the expectation of the empirical Rademacher complexity, but in this article we are using only empirical Rademacher complexity. For now on, if we say Rademacher complexity we mean the empirical Rademacher complexity.

Remark 2.4. Instead of the notation $\widehat{\mathcal{R}}_S(\mathcal{F},m)$ we might denote the Rademacher complexity just by $\widehat{\mathcal{R}}(\mathcal{F})$, omitting the letters that indicate the fact that Rademacher complexity depends on the sample.

Next lemma is from [30], with this lemma we are going to bound the difference of the two loss functions.

Lemma 2.2. *Let $\mathcal{H}$ be a set of functions (hypothesis class). Let $h\in\mathcal{H}$, and let $z\in\Omega\subset\mathbb{R}^d$. Let l be a function depending on the function h and the variable z, let's denote a collection of maps as*

$$l\circ\mathcal{H}:=\{z\mapsto l(h,z):h\in\mathcal{H}\},$$

where the mappings map from Ω to $\mathbb{R}$. Let $f\in l\circ\mathcal{H}$. The expectation of a function f with respect to distribution $\mathcal{D}$ is

$$L_{\mathcal{D}}(f)=\mathop{\mathbb{E}}_{z\sim\mathcal{D}}[f(z)],$$

and the empirical expectation is

$$L_S(f)=\frac{1}{n}\sum_{j=1}^n f(z_j).$$

Let $C>0$ and let's assume that for all $z\in\Omega$ and all $h\in\mathcal{H}$ we have $|l(h,z)|\le C$. Let $\delta>0$. Then with probability at least $1-\delta$ for all $f\in l\circ\mathcal{H}$

$$L_D(f)-L_S(f)\le 2\widehat{\mathcal{R}}_S(l\circ\mathcal{H},m)+3C\sqrt{\frac{2\ln(4/\delta)}{m}}.$$

DOI https://doi.org/10.4208/cicp.OA-2023-0253 | Generated on 2024-12-19 11:09:00

Proof. See [30] theorem 26.5. □

Next we present a lemma which claims that the Rademacher complexity of a composition function of a Lipschitz function and a set of functions is bounded by the Lipschitz constant times the Rademacher complexity of the set of functions.

Lemma 2.3. *(Talagrand's contraction lemma) Let $\phi:\mathbb{R}\to\mathbb{R}$ be a ρ-Lipschitz function. Let $\mathcal{H}:=\{h|h:\Omega\to\mathbb{R}\}$. Let $\phi\circ\mathcal{H}=\{z\mapsto\phi(h(z)):h\in\mathcal{H}\}$ is a collection of maps from Ω to $\mathbb{R}$. Then we have*

$$\widehat{\mathcal{R}}(\phi\circ\mathcal{H})\leq\rho\widehat{\mathcal{R}}(\mathcal{H}). \tag{2.9}$$

Proof. See [30]. □

In [31] the following bound for the Rademacher complexity of two layer neural networks with the Relu activation function was presented

$$\widehat{\mathcal{R}}(\mathcal{F}_Q)\leq 2Q\sqrt{\frac{2\ln(2d+2)}{n}}.$$

Also [12,29] provides a good review on the theme. One limitation of this result is that it is given for neural networks with the Relu activation function. [28] extended these results to neural networks with approachable activation functions; a bound for Rademacher complexity of two-layer neural networks and another similar result for residual neural networks.

Theorem 2.2. *Assume that an activation function σ satisfies the conditions of the Definition* 2.1. *Let $\mathcal{F}_Q=\{f(x,\theta):\|\theta\|_{\tilde{\mathcal{P}}}\leq Q,x\in\Omega\}$ be a two-layer neural net with the activation function σ and weights bounded by $\|\theta\|_{\tilde{\mathcal{P}}}\leq Q$. Then we have*

$$\widehat{\mathcal{R}}(\mathcal{F}_Q)\leq 2\gamma(\sigma)Q\sqrt{\frac{2\ln(2d+2)}{n}}, \tag{2.10}$$

where $\gamma(\sigma)$ depends only on σ and is defined in the Definition 2.1.

Proof. See article [28]. □

And a similar theorem for residual neural networks.

Theorem 2.3. *Assume that an activation function σ satisfies the conditions of the Definition* 2.1. *Let $\mathcal{F}_Q=\{f(x,\theta):\|\theta\|_{\tilde{\mathcal{P}}}\leq Q,x\in\Omega\}$ be a residual neural network with the activation function σ and weights bounded by $\|\theta\|_{\tilde{\mathcal{P}}}\leq Q$. Then we have*

$$\widehat{\mathcal{R}}(\mathcal{F}_Q)\leq c_\sigma Q\sqrt{\frac{2\ln(2d+2)}{n}},$$

where $c_\sigma=4\gamma(\sigma)+1$ is a constant only related to the activation function σ, $\gamma(\sigma)$ is defined in Definition 2.1.

Proof. See article [28]. □

2.4 Bound for the integrand with different architectures

In Lemma 2.2, we notice that in order to use the Lemma the function l should be bounded. For this reason we present the following Lemma where we show that the integrand in the DRM loss function is bounded when defining it with shallow or residual neural networks.

Lemma 2.4 (Bounds for shallow neural networks). *Let $u \in \mathcal{F}_Q := \{f(x,\theta) : \|\theta\|_{\mathcal{P}} \leq Q,\ x \in \Omega\}$ be a shallow network with bounded path norm $\|\theta\|_{\mathcal{P}} < Q$. Let the activation function of the network be σ, and let's assume σ is M-Lipschitz, and σ' is K-Lipschitz function, with some $M > 0$, and $K > 0$. Then with this architecture, all neural networks and their gradients are bounded in the following way:*

$$|u(x,\theta)| < MQ,$$
$$|u(x,\theta)|^2 < M^2Q^2,$$

and

$$\|\nabla_x u(x,\theta)\|_2^2 < K^2Q^2.$$

If instead of K-Lip property of σ' we have that σ' is bounded, $|\sigma'(z)| < C_\sigma$ for all $z \in \mathbb{R}$ for some $C_\sigma > 0$, then we have

$$\|\nabla_x u(x,\theta)\|_2^2 < C_\sigma^2 Q^2.$$

Proof. In the appendix, see the proof of Lemma A.3. □

The same holds for the $\tilde{\mathcal{P}}$-path norm.

Corollary 2.1. *Let $u \in \mathcal{F}_Q := \{u(x,\theta) : \|\theta\|_{\tilde{\mathcal{P}}} \leq Q,\ x \in \Omega\}$ be a shallow neural network with the bounded path norm $\|\theta\|_{\tilde{\mathcal{P}}} < Q$. Let the activation function of the network be σ, and let's assume σ is M-Lipschitz, and σ' is K-Lipschitz function, with some $M > 0$, and $K > 0$. Then with this architecture, all neural networks and their gradients are bounded in the following way:*

$$|u(x,\theta)| < MQ,$$
$$|u(x,\theta)|^2 < M^2Q^2,$$

and

$$\|\nabla_x u(x,\theta)\|_2^2 < K^2Q^2.$$

If instead of K-Lip property of σ' we have that σ' is bounded, $|\sigma'(z)| < C_\sigma$ for all $z \in \mathbb{R}$ for some $C_\sigma > 0$, then we have

$$\|\nabla_x u(x,\theta)\|_2^2 < C_\sigma^2 Q^2.$$

Proof. The proof follows immediately from Lemmas 2.1 and 2.4. □

DOI https://doi.org/10.4208/cicp.OA-2023-0253 | Generated on 2024-12-19 11:09:00

Corollary 2.2. *Let $\mathcal{F}_Q$ be a shallow neural network with a bounded path norm $\|\theta\|_{\tilde{\mathcal{P}}} < Q$. Let the activation function σ of the neural network be M-Lipschitz, and let its first order derivative σ' be K-Lipschitz, with some $M > 0$ and $K > 0$. Then the integrand of the Deep Ritz method's loss function is bounded*

$$\left|\frac{1}{2}|\nabla u(x,\theta)|^2 - |u(x,\theta)|\right| < \frac{1}{2}K^2Q^2 + MQ, \tag{2.11}$$

for all $\theta \in \mathcal{F}_Q$ and $x \in B(0,1)$. Similar argument holds for the boundary term: $|u(x,\theta)|^2 < M^2Q^2$ for all for all $\theta \in \mathcal{F}_Q$ and $x \in \partial B(0,1)$.

If instead of K-Lip property of σ' we have that σ' is bounded, i.e. there exists $C_\sigma > 0$ such that $|\sigma'(z)| < C_\sigma$ for all $z \in \mathbb{R}$, then we have

$$\left|\frac{1}{2}|\nabla u(x,\theta)|^2 - |u(x,\theta)|\right| < \frac{1}{2}C_\sigma^2Q^2 + MQ. \tag{2.12}$$

Proof. The proof follows immediately from Lemma 2.4 and Corollary 2.1. □

Lemma 2.5 (Bounds for residual neural networks)**.** *Let $u \in \mathcal{F}_Q$ be a residual neural network with a bounded path norm $\|\theta\|_{\tilde{\mathcal{P}}} < Q$. Let the activation function of the network be σ, and let σ be K-Lipschitz, and let σ' be bounded by some $C_\sigma > 0$, $|\sigma'(z)| < C_\sigma$ for all $z \in \mathbb{R}$. Let $L \in \mathbb{N}$ be the number of residual blocks. Then with this architecture all neural networks and their gradients are bounded in the following way:*

$$\begin{aligned}
&|u(x,\theta)| < (K \vee 1)^L Q,\\
&|u(x,\theta)|^2 < (K \vee 1)^{2L} Q^2,\\
&\|\nabla_x u(x,\theta)\|_2^2 < (C_\sigma \vee 1)^{2L} Q^2.
\end{aligned}$$

Without any information about the magnitude of K and C_σ, we can say

$$\begin{aligned}
&|u(x,\theta)| < Q(K^L + \cdots K + 1),\\
&|u(x,\theta)|^2 < Q^2(K^L + \cdots K + 1)^2,\\
&\|\nabla_x u(x,\theta)\|_2^2 < Q^2(C_\sigma^L + \cdots + C_\sigma + 1)^2.
\end{aligned}$$

Proof. Proof is given in the appendix. □

Corollary 2.3. *Let $u \in \mathcal{F}_Q$ be a residual neural network with a bounded path norm $\|\theta\|_{\tilde{\mathcal{P}}} < Q$. Let the activation function σ of the neural network be M-Lipschitz with some $M > 0$, and let its first order derivative σ' be bounded $|\sigma'(z)| < C_\sigma$ for all $z \in \mathbb{R}$. Then the integrand of the Deep Ritz method is bounded*

$$\left|\frac{1}{2}|\nabla u(x,\theta)|^2 - |u(x,\theta)|\right| < \frac{1}{2}(C_\sigma \vee 1)^{2L}Q^2 + (K \vee 1)^L Q, \tag{2.13}$$

for all $\theta \in \mathcal{F}_Q$ and $x \in B(0,1)$. Similar argument holds for the boundary term:

$$|u(x,\theta)|^2 < (K \vee 1)^{2L} Q^2, \tag{2.14}$$

for all $\theta \in \mathcal{F}_Q$ and $x \in \partial B(0,1)$.

2.5 Statistical errors

With all the tools we have presented here, we can proceed to prove the main theorems of this article. Here is the bound for the statistical error of the Deep Ritz method for shallow, and residual neural networks with approachable activation functions.

Theorem 2.4 (Statistical error of DRM for shallow neural networks)**.** *Let $u(x,\theta)\in\mathcal{F}_Q$, be a shallow neural network with a bounded path norm $\|\theta\|_{\tilde{\mathcal{P}}}<Q$, with an approachable activation function σ. Let σ be M-Lipschitz. We have two cases, we assume either σ' is K-Lipschitz or $|\sigma'|<C_\sigma$, with some $C_\sigma>0$. Let $\delta>0$ be a confidence parameter. With probability $1-\delta$, the statistical error of the Deep Ritz method is bounded as*

$$|\mathcal{L}(u)-\widehat{\mathcal{L}}(u,n,m)|\leq C|\Omega|\left(4\gamma(\sigma)Q\sqrt{\frac{2\ln(2d+2)}{n}}+3\left(\frac{1}{2}AQ^2+Q\right)\sqrt{\frac{2\ln(4/\delta)}{n}}\right)$$
$$+C\lambda|\partial\Omega|\left(\gamma(\sigma)Q\sqrt{\frac{2\ln(2d+2)}{m}}+\frac{3}{2}Q^2\sqrt{\frac{2\ln(4/\delta)}{m}}\right),$$

where n is the number of sample points in Ω, $C>0$ is a maximum of Lipschitz constant for the loss function inside the domain and on the boundary, and m is the number of sample points on the boundary $\partial\Omega$. The constant A is

$$\begin{cases} A=K^2 & \text{if } \sigma' \text{ is } K\text{-Lip}, \\ A=C_\sigma^2 & \text{if } |\sigma'|<C_\sigma. \end{cases} \tag{2.15}$$

Proof. Proof given in the appendix. See Appendix A.1. □

Theorem 2.5 (Statistical error of the DRM for residual neural networks)**.** *Let $u(x,\theta)\in\mathcal{F}_Q$ be a residual neural network with $L\in\mathbb{N}$ skip connections, with bounded path norm $\|\theta\|_{\tilde{\mathcal{P}}}<Q$, $Q>0$, and with an approachable activation function σ. Let σ be M-Lipschitz, and let σ' be bounded by $|\sigma'(z)|<C_\sigma$ for all $z\in\mathbb{R}$. Let $\delta>0$ be a confidence parameter. With probability $1-\delta$, the statistical error of the Deep Ritz method is bounded as*

$$|\mathcal{L}(u)-\widehat{\mathcal{L}}(u,n,m)|\leq C|\Omega|\left(4c_\sigma Q\sqrt{\frac{2\ln(2d+2)}{n}}+3\left(\frac{1}{2}(C_\sigma\vee 1)^{2L}Q^2+(K\vee 1)^L Q\right)\sqrt{\frac{2\ln(4/\delta)}{n}}\right)$$
$$+C\lambda|\partial\Omega|\left(c_\sigma Q\sqrt{\frac{2\ln(2d+2)}{m}}+\frac{3}{2}(K\vee 1)^{2L}Q^2\sqrt{\frac{2\ln(4/\delta)}{m}}\right),$$

where n is the number of sample points in the domain Ω, $C>0$ is a maximum of Lipschitz constant for the loss function inside the domain and on the boundary, m is the number of sample points on the boundary $\partial\Omega$, and $c_\sigma=4\gamma(\sigma)+1$.

Proof. Given in the appendix. See Appendix A.2. □

3 Literature and discussion

This work is one in the line of research about the complexity measures for different neural network structures, [12,28,29,31], showing that with path norms like the ones used here we can have meaningful error bounds, also some other complexity measures have been proposed [32,33]. In this article we have extended the study of the complexity measures to the study of Deep Ritz method. To achieve a path-norm bound in a realistic setting, some form of regularization [34] is needed to force the weights not to blow up.

In recent years there has been a lot of research for Deep Ritz Method [5,6,10,13–15, 35,36], PINNs [17,37], and other methods [38–41] to solve PDEs with neural networks. Like said, some try to approximate the solutions in Sobolev spaces [10,13,35,36] some in Barron space [6,12].

The main results of this article are related to generalization error of Deep Ritz method. Also other research groups have considered these issues. The main difference between this study and other studies is that we consider residual neural networks, most of previous studies have consider only feed forward neural networks, also our results apply for shallow networks. Feed forward neural networks work well with Deep Ritz method, and as feed forward neural networks usually are rather small size they are fast to train. But there is a problem that feed forward neural networks suffer of the vanishing/exploding gradient problem and therefore they cannot be made as deep as one would like; residual neural networks in contrast can be made as deep as one likes and they are still able to train. Another related difference is the activation function, in the former studies authors have considered architectures with activation functions like Relu^2, or sigmoidal activation functions, but in this study we have decided to focus on general smooth activation functions: our aim is to provide a theory that applies to activations which are smooth versions of the Relu activation function. The reason for this is the fact that residual neural networks have stable initialization with Relu activation function, and also smooth versions also provide stable initialization. These smooth versions of Relu are included in the definition of general activations so called approachable activation functions.

We think that residual networks have practical and theoretical use in the study of Deep Ritz method and other deep learning methods to solve PDEs. One aspect is the fact that in PDE problem which we consider the solution belongs to the Barron space and residual neural networks can approximate these functions in a natural way, this kind of a theory for feed forward neural networks still waits to be discovered.

One important difference to the previous studies is that we do not assume that the weights of the network architecture are bounded, instead we consider neural networks with bounded path-norm, to our knowledge other groups have not considered these kind of network architectures in the Deep Ritz method studies.

The main theorems show that the statistical error of the Deep Ritz method is $\mathcal{O}(\frac{1}{\sqrt{n}},\frac{1}{\sqrt{m}})$, meaning that it converges to zero polynomially with rate $\frac{1}{\sqrt{n}}$ as n goes to infinity. Property has been verified for neural networks also by other researchers [6], [16], and [23]. This is what we should expect from the [42] monte carlo integration of a smooth

function in a bounded domain. The main Theorems 2.4, and 2.5 also give an indication how the network structure affects the generalization error: interestingly the width of shallow neural networks does not really affect the generalization error, when we look at the Theorem 2.4, whereas with residual neural networks the number of skip connections causes the error to grow; reason for this is in activation functions. It is not actually the number of skip connections, but the number of the activation functions in the residual network that affects the term $\frac{1}{2}(C_\sigma \vee 1)^{2L} Q^2 + (K \vee 1)^L Q$ in the error estimation, and similarly the only activation function affects shallow networks' estimate.

For comparison, we collect here some generalization error estimations from other publications. In [10] following bound was given

$$\begin{aligned}\mathcal{E}_{sta} \leq & Cd(\mathcal{D}+3)(\mathcal{D}+2)\mathcal{W}\sqrt{\mathcal{D}+3+\log(d((\mathcal{D}+2)\mathcal{W}))}\Big(\frac{\log(n)}{n}\Big)^{1/2} \\ & + Cd\mathcal{D}\mathcal{W}\sqrt{\mathcal{D}+\log(\mathcal{W})}\Big(\frac{\log(n)}{n}\Big)^{1/2}\lambda.\end{aligned}$$

Bound is given for a feed forward neural network with depth $\mathcal{D}$, and width $\mathcal{W}$. It is not totally clear if they have considered the boundary term of the loss function here as well, or if the boundary has same amount of data points (n) in the empirical risk calculation. d is the dimension of the problem as they do not restrict the study in a plane as we do. λ is the boundary penalty parameter.

In [13] a following error bound is given

$$\mathbb{E}_{\{X_i\}_{i=1}^N, \{Y_i\}_{i=1}^M} \sup_{u \in \mathcal{P}} \pm \Big[\mathcal{L}(u) - \widehat{\mathcal{L}}(u)\Big] \leq \frac{C_{\Omega, coe, \alpha}}{\beta} \frac{d\sqrt{\mathcal{D}}\eta_{\mathcal{D}}^{2\mathcal{D}} B_\theta^{2\mathcal{D}}}{\sqrt{N}} \sqrt{\log(d\mathcal{D}\eta_{\mathcal{D}} B_\theta N)}.$$

We cannot fully explain here all the terms in the bound, but let's say something. $\mathcal{D}$ is the depth of a feed forward neural network, and $\eta_{\mathcal{D}}$ is the number of parameters in the neural network. β is the boundary penalty parameter. Bound is given so that the number of data points on the boundary and inside the domain is equal $N = M$.

In [14] authors arrive in error bound

$$\mathcal{E}_{sta} \leq C\left[d(\mathcal{D}+3)(\mathcal{D}+2)\mathcal{W}\sqrt{\frac{\mathcal{D}+3+\log(d(\mathcal{D}+2)\mathcal{W})}{n}}\right]^{1-\nu}.$$

Similarly to the previous bounds, $\mathcal{D}$ is the depth of the neural network architecture and $\mathcal{W}$ is the width, d is the dimension of the input layer, and n is the number of data points (probably the number of data points inside the domain and on the boundary is the same). It's a bit unclear to us how they get rid of the $log(n)$ term, but this is how they present the result.

Biggest differences to our study is that the previously mentioned studies [10, 13, 14] focused on feed forward neural networks. At least in [14] the neural network weights are bounded, whereas in our analysis the bound comes from the path-norm bound.

In [6] authors studied the Deep Ritz method with shallow neural networks and obtained.

For shallow we showed an error estimate

$$|\mathcal{L}(u)-\widehat{\mathcal{L}}(u,n,m)|\leq C|\Omega|\left(4\gamma(\sigma)Q\sqrt{\frac{2\ln(6)}{n}}+3\Big(\frac{1}{2}AQ^2+Q\Big)\sqrt{\frac{2\ln(4/\delta)}{n}}\right)$$
$$+C\lambda|\partial\Omega|\left(\gamma(\sigma)Q\sqrt{\frac{2\ln(6)}{m}}+\frac{3}{2}Q^2\sqrt{\frac{2\ln(4/\delta)}{m}}\right). \tag{3.1}$$

For the shallow networks the bound works well in the way that there is not much in addition to the pathnorm bound Q, we mean there is no terms from the width of the shallow network architecture. We can have as wide network as want as long as the path norm stays under the bound Q. For example in it is shown that Barron functions can be approximated with error estimate

$$\mathbb{E}_x(f^*(x)-f(x;\theta))\leq\frac{3\gamma_2^2(f^*)}{m}, \tag{3.2}$$
$$\|\theta\|_{\mathcal{P}}\leq 2\gamma_2(f^*), \tag{3.3}$$

where $f^*(x)$ is the target function and $f(x,\theta)$ is a shallow neural network. The above error estimate means that one can go to any desired accuracy as long as the path norm stays under the bound $2\gamma_2(f^*)$, where $\gamma_2(f^*)$ is the so-called spectral norm [12]. We have show that the solution to the Poisson's equation belongs to the Barron space and so these kinds of error estimations hold.

For residual neural network we presented a following error bound:

$$|\mathcal{L}(u)-\widehat{\mathcal{L}}(u,n,m)|\leq C|\Omega|\left(4c_\sigma Q\sqrt{\frac{2\ln(2d+2)}{n}}\right.$$
$$\left.+3\Big(\frac{1}{2}(C_\sigma\vee 1)^{2L}Q^2+(K\vee 1)^L Q\Big)\sqrt{\frac{2\ln(4/\delta)}{n}}\right)$$
$$+C\lambda|\partial\Omega|\left(c_\sigma Q\sqrt{\frac{2\ln(2d+2)}{m}}\right.$$
$$\left.+\frac{3}{2}(K\vee 1)^{2L}Q^2\sqrt{\frac{2\ln(4/\delta)}{m}}\right). \tag{3.4}$$

The error bound is similar to the one obtained for the shallow networks, except the $(C_\sigma\vee 1)^{2L}Q^2+(K\vee 1)^L Q)$ and $(K\vee 1)^{2L}$ terms which grow exponentially with the number of skip connections L.

Similarly to shallow network, there is a Barron space error estimate for the residual neural networks. In article [29] (theorem 2.5) the authors showed that residual networks

can approximate the Barron functions in a following way: Let f^* be the Barron function. There exists a residual network $f(\cdot,\theta)$ with depth L and width m such that

$$\|f(x,\theta)-f^*\| \leq \frac{3\|f^*\|_{\mathcal{B}}^2}{Lm}, \tag{3.5}$$

$$\|\theta\|_P \leq 4\|f^*\|_{\mathcal{B}}, \tag{3.6}$$

where $\|f^*\|_{\mathcal{B}}$ is a Barron norm.

Also [6] studied Deep Ritz method with focus on the Barron space estimation, they studied it with shallow neural networks with softplus activation function. Their network architecture is a bit different as they consider architecture with bounded weights and in we consider bounded path norm.

The error bounds (3.1), and (3.4) are of course better the smaller path norm bound Q is chosen. We cannot though push Q to zero as this would severely weaken the approximative capability of the neural network architecture: Q can be chosen with respect to the Barron space estimates. For shallow networks following error estimate (3.2) Q could be $2\gamma_2(u)$, where u is the solution for the Poisson's equation and $\gamma(u)$ is the spectral norm of u. We cannot give exact number for this spectral norm, but it can be estimated from above. Similarly for residual neural network Q could be chosen to be $4\gamma(u)$ with respect to Barron space approximation capability (3.5).

In both of these architectures the bound converges to zero with rate $\mathcal{O}(\frac{1}{\sqrt{n}},\frac{1}{\sqrt{m}})$ with respect to the number of data points inside the domain and on the boundary. Approachable activation functions which we have in the mind are closely 1-Lipschitz (for example Gelu, and Swish), and then the constant K is nearly one in the error bounds. Quality of an activation function plays a crucial role in this bound, and it affects many coefficients. $\gamma(\sigma)$ is a coefficient that is set solely based on the activation function. In [28] authors have computed γ values for a group of activation functions: for the Gelu it is approximatively 2,5 and for the Swish approximatively $\frac{1,8}{\alpha}+1,4$. In the appendix we have present an activation function for which the γ value is $\frac{3}{8}\epsilon+\frac{3}{2}$, where $\epsilon>0$ is a small number, and so the value approaches 1,5. This gives an idea at what magnitude the $\gamma(\sigma)$ is going to be. For our purposes, an activation function should be similar to Relu, and this rules out activation functions like tanh and sigmoid, also Relu itself is out of question due its poor differentiability.

In our setting, a neural network with bounded path norm in a bounded domain, the function defined by the network is bounded, but to have a bounded gradient, as we show in Lemmas 2.4, and 2.5, the Relu activation function is out of question; and so it is critically important to be able to use smooth version of the Relu activation function. Another reason to have the Lemmas 2.4, and 2.5, is to get concrete bounds for the main Theorems.

Some researchers have raised criticism for the existing tools to study generalization [43]. They have a good point, generalization of neural network remains a challenging problem as the theoretical error estimates are significantly bigger than the experimental findings of the generalization error. But the existing state-of-the-art theoretical

error bounds should not be totally disregarded as they still give us information about the problem, as discussed earlier in this article. One way to try to surpass this problem is of course to look at the optimization algorithm, thing which we do not consider in this article. The error estimates presented in this article are in a sense global, they give quarantee for the whole space of neural networks, but stochastic gradient decent is a local process, happening in some small region of the parameter space. Some research have been done for example within implicit regularization [44], where they study the regularization properties of optimization algorithms. And the SGD algorithms have been modeled as stochastic differential equations [45].

Acknowledgments

We thank Academy of Finland for funding this research. Thanks for anonymous peer reviewers for their valuable comments. Also thanks for professor Tuomo Kuusi for all the support and discussions over time.

Appendices

A Missing proofs

Lemma A.1. *Let $u(x,\theta)\in\mathcal{F}_Q$ be a shallow neural network of width m, and let $\|\theta\|_{\tilde{\mathcal{P}}}<Q$, where $Q>0$ is some positive real number. Then we have $\|\theta\|_{\mathcal{P}}<\|\theta\|_{\tilde{\mathcal{P}}}$, and in particular $\|\theta\|_{\mathcal{P}}<Q$.*

Proof. Let $x\in\mathbb{R}^d$. The path norm, according to Definition 2.4, of u is

$$\|\theta\|_{\tilde{\mathcal{P}}}=\sum_{i=1}^{m}|a_i|\big(\|b_i\|_1+|c_i|+1\big).$$

Let $h(x,\theta)$ be another neural net with $x\in\mathbb{R}^{d+1}$, such that $x=(x_d,1)$, where $x_d\in\mathbb{R}^d$, and parameters θ such that

$$\omega_i=(b_i,c_i),$$

and weights a_i are the same as in the neural network $u(x,\theta)$. If we now overload this notation so that with x we mean $(x,1)$ when talking about the neural network h, and x when talking about the neural network u, then with this definition the neural networks $h(x,\theta)$ and $u(x,\theta)$ output the same value for all $x\in\mathbb{R}^d$. So, they are the same neural net, only difference is how bias terms are defined; and after all, the bias terms are the same. Also, the actual input of the neural net is the same, it is just that h contains an "extra" dimension, but on this dimension the input is always 1, in fact the neural network u would have the same input as well, it is just that this input is hidden into the different

way of expressing the bias term. Anyway, with all this we can say $h(x,\theta)=u(x,\theta)$. Now let's compare the two norms

$$\begin{aligned}
\|\theta\|_{\tilde{\mathcal{P}}} &= \sum_{i=1}^{m} |a_i| \left(\|b_i\|_1 + |c_i| + 1 \right) \\
&= \sum_{i=1}^{m} |a_i| \left(\sum_{j=1}^{d} |b_j^i| + |c_i| + 1 \right) \\
&= \sum_{i=1}^{m} |a_i| \left(|b_1^i| + \cdots + |b_d^i| + |c_i| + 1 \right) \\
&= \sum_{i=1}^{m} |a_i| \left(\sum_{j=1}^{d+1} |\omega_j^i| + 1 \right) \\
&= \sum_{i=1}^{m} |a_i| \left(\|\omega_i\|_1 + 1 \right) \\
&> \sum_{i=1}^{m} |a_i| \left(\|\omega_i\|_1 \right) \\
&= \|\theta\|_{\mathcal{P}}.
\end{aligned}$$

This completes the proof. □

Lemma A.2. *Let $u(x,\theta)\in\mathcal{F}_Q$ be a residual neural network, and let $\|\theta\|_{\tilde{\mathcal{P}}}<Q$, where $Q>0$ is some positive real number. Then we have $\|\theta\|_{\mathcal{P}}<\|\theta\|_{\tilde{\mathcal{P}}}$, and in particular $\|\theta\|_{\mathcal{P}}<Q$.*

Proof. In [28] in Lemma 3 in it is proven that $\|\theta\|_{\tilde{\mathcal{P}}}=\|\theta\|_{\mathcal{P}}+r$, where $r>0$ is a positive number. □

Lemma A.3 (Bounds for a shallow net)**.** *Let $u\in\mathcal{F}_Q$ be a shallow neural network with a bounded path norm $\|\theta\|_{\mathcal{P}}<Q$. Let the activation function of the network be σ, and let's assume σ is M-Lipschitz, and σ' is K-Lipschitz function, with some $K>0$. Let $x\in\Omega=B(0,1)$. Then neural networks with this architecture are bounded, and their gradients are bounded in the following way:*

$$\begin{aligned}
|u(x,\theta)| &< MQ, \\
|u(x,\theta)|^2 &< M^2Q^2,
\end{aligned}$$

and

$$\|\nabla_x u(x,\theta)\|_2^2 < K^2Q^2.$$

Proof. As $u\in\mathcal{F}_Q$, we know that $\|\theta\|_P<Q$. From $|u(x,\theta)|\leq Q$ we obviously get $|u(x,\theta)^2|\leq$

DOI https://doi.org/10.4208/cicp.OA-2023-0253 | Generated on 2024-12-19 11:09:00

Q^2. Let's prove that $|u(x,\theta)| \leq Q$

$$\begin{aligned}
|u(x,\theta)| &= \Big|\sum_{k=1}^{m} a_k \sigma(\langle \omega_k, x\rangle)\Big| \leq \sum_{k=1}^{m} |a_k||\sigma(\langle \omega_k, x\rangle)| \\
&\overset{(*)}{\leq} M \sum_{k=1}^{m} |a_k||\langle \omega_k, x\rangle| \\
&\leq M \sum_{k=1}^{m} |a_k| \|\omega\|_1 \|x\|_\infty \\
&\overset{(**)}{\leq} M \sum_{k=1}^{m} |a_k| \|\omega\|_1 \\
&= M\|\theta\|_P \\
&\leq MQ.
\end{aligned}$$

At (*) we use the M-Lip property of σ. At (**) we used the boundedness of Ω.

For the gradient, let's calculate

$$\begin{aligned}
\|\nabla_x u(x,\theta)\|_2^2 &= \|\sum_{k=1}^{m} a_k \sigma'(\langle \omega_k, x\rangle)\omega_k\|_{l^2}^2 \\
&= \sum_{j=1}^{L} \Big|\sum_{k=1}^{N} a_k \big(\sigma'(\langle \omega_k, x\rangle)\big)\omega_k^j\Big|^2 \\
&\leq \sum_{j=1}^{L} \Big(\sum_{k=1}^{N} |a_k|^2 |\sigma'(\langle \omega_k, x\rangle)|^2 |\omega_k^j|^2 \\
&\qquad + \sum_{\substack{k,l=1 \\ k\neq l}}^{N} |a_k a_l| |\sigma'(\langle \omega_k, x\rangle)| |\sigma'(\langle \omega_l, x\rangle)| |\omega_k^j||\omega_l^j| \Big) \\
&\leq K^2 \sum_{j=1}^{L}\sum_{k=1}^{N} |a_k|^2 |\omega_k^j|^2 + K^2 \sum_{j=1}^{L} \sum_{\substack{k,l=1 \\ k\neq l}}^{N} |a_k a_l||\omega_k^j||\omega_l^j| \\
&\leq K^2 \sum_{k=1}^{N} |a_k|^2 \sum_{j=1}^{L} |\omega_k^j|^2 + K^2 \sum_{\substack{k,l=1 \\ k\neq l}}^{N} |a_k a_l| \sum_{j=1}^{L} |\omega_k^j||\omega_l^j| \\
&\leq K^2 \sum_{k=1}^{N} |a_k|^2 \|\omega_k\|_{l^2}^2 + K^2 \sum_{\substack{k,l=1 \\ k\neq l}}^{N} |a_k a_l| \sum_{j=1}^{L} |\omega_k^j||\omega_l^j|. \qquad \text{(A.1)}
\end{aligned}$$

Let's pause here for a moment, and look at $\|\theta\|_P^2$:

$$\|\theta\|_P^2 = \Big(\sum_{k=1}^{N} |a_k| \|\omega_k\|_{l^1}\Big)^2 = \sum_{k=1}^{N} |a_k|^2 \|\omega_k\|_{l^1}^2 + \sum_{\substack{k,l=1 \\ k\neq l}}^{N} |a_k||a_l| \|\omega_k\|_{l^1} \|\omega_l\|_{l^1}.$$

Therefore if $\|\theta\|_P < Q$, then we have

$$\sum_{k=1}^{N} |a_k|^2 \|\omega_k\|_{l^1}^2 + \sum_{\substack{k,l=1 \\ k\neq l}}^{N} |a_k||a_l| \|\omega_k\|_{l^1} \|\omega_l\|_{l^1} < Q^2. \tag{A.2}$$

We know that the l^2-norm of any vector is bounded by its l^1 norm, also by calculating

$$\begin{aligned} \|\omega_k\|_1 \|\omega_l\|_1 &= (|\omega_k^1| + \cdots + |\omega_k^L|)(|\omega_k^1| + \cdots + |\omega_k^L|) \\ &= |\omega_k^1||\omega_l^1| + \cdots + |\omega_k^L||\omega_l^L| + R(\omega_k, \omega_l), \end{aligned}$$

where $R(\omega_k, \omega_l) > 0$ contains the remainder terms, we can deduce that

$$|\omega_k^1||\omega_l^1| + \cdots + |\omega_k^L||\omega_l^L| \leq \|\omega_k\|_1 \|\omega_l\|_1. \tag{A.3}$$

With these we can continue from (A.1) and conclude that

$$\|\nabla_x u(x,\theta)\|_2^2 \leq K^2 Q^2. \tag{A.4}$$

This completes the proof. □

Lemma A.4 (Bounds for residual neural networks)**.** *Let $u \in \mathcal{F}_Q$ be a residual neural network with a bounded path norm $\|\theta\|_{\mathcal{P}} < Q$. Let the activation function of the network be σ, and let's assume σ is K-Lipschitz, and σ' is bounded function, with some $C_\sigma > 0$, $|\sigma'(x)| < C_\sigma$ for all $x \in \Omega$. Let $x \in \Omega = B(0,1)$. Let $L \in \mathbb{N}$ be the depth of the neural net, meaning the number of residual blocks. Then neural networks with this architecture are bounded, and their gradients are bounded in the following way:*

$$\begin{aligned} &|u(x,\theta)| < (K \vee 1)^L Q, \\ &|u(x,\theta)|^2 < (K \vee 1)^{2L} Q^2, \\ &\|\nabla_x u(x,\theta)\|_2^2 < (C_\sigma \vee 1)^{2L} Q^2. \end{aligned}$$

Without any information about the magnitude of K and C_σ, we can say

$$\begin{aligned} &|u(x,\theta)| < Q(K^L + \cdots K + 1), \\ &|u(x,\theta)|^2 < Q^2 (K^L + \cdots K + 1)^2, \\ &\|\nabla_x u(x,\theta)\|_2^2 < Q^2 (C_\sigma^L + \cdots + C_\sigma + 1)^2. \end{aligned}$$

Proof. If $L = 0$ then the mapping is not a residual neural network, so $L = 1$ is the lowest depth we can have. Let $L = 1$. Then $u(x,\theta)$ is

$$u(x,\theta) = a^T \Big(U_1 \sigma(W_1 V x) + V x \Big),$$

or in another way

$$u(x,\theta)=\sum_{m,k=1}^{n}a_m u_{mk}\sigma\Big(\sum_{i=1}^{n}\sum_{j=1}^{d}w_{ki}v_{ij}x_j\Big)+\sum_{m}^{n}\sum_{j=1}^{d}a_m v_{mj}x_j.$$

And so the absolute value is bounded by

$$\begin{aligned}|u(x,\theta)|&\leq\sum_{m,k=1}^{n}|a_m||u_{mk}|K\sum_{i=1}^{n}\sum_{j=1}^{d}|w_{ki}||v_{ij}|+\sum_{m}^{n}\sum_{j=1}^{d}|a_m||v_{mj}|\\&\leq K\sum_{m,k=1}^{n}|a_m||u_{mk}|\sum_{i=1}^{n}\sum_{j=1}^{d}|w_{ki}||v_{ij}|+\sum_{m}^{n}\sum_{j=1}^{d}|a_m||v_{mj}|.\end{aligned}$$

Now there is three ways how we can proceed from here. One is the following, as

$$\sum_{m,k=1}^{n}|a_m||u_{mk}|\sum_{i=1}^{n}\sum_{j=1}^{d}|w_{ki}||v_{ij}|\leq\|\theta\|_{\mathcal{P}}<Q$$

and

$$\sum_{m}^{n}\sum_{j=1}^{d}|a_m||v_{mj}|\leq\|\theta\|_{\mathcal{P}}<Q,$$

we can say

$$|u(x,\theta)|\leq KQ+Q.$$

But on the other hand, if $K<1$ we can say

$$\begin{aligned}|u(x,\theta)|&\leq K\sum_{m,k=1}^{n}|a_m||u_{mk}|\sum_{i=1}^{n}\sum_{j=1}^{d}|w_{ki}||v_{ij}|+\sum_{m}^{n}\sum_{j=1}^{d}|a_m||v_{mj}|\\&\leq\sum_{m,k=1}^{n}|a_m||u_{mk}|\sum_{i=1}^{n}\sum_{j=1}^{d}|w_{ki}||v_{ij}|+\sum_{m}^{n}\sum_{j=1}^{d}|a_m||v_{mj}|\\&\leq Q.\end{aligned}$$

And if $K\geq 1$ then we have

$$|u(x,\theta)|\leq KQ.$$

Let $L=2$. Then the neural net is

$$u(x,\theta)=a^T\Big[U_2\sigma\Big(W_2\big(U_1\sigma(W_1Vx)+Vx\big)\Big)+U_1\sigma(W_1Vx)+Vx\Big],$$

and the same by using the hidden layer notation

$$\begin{aligned}
&h_0' = h_0,\\
&h_1' = U_1\sigma(W_1 h_0),\\
&h_2' = U_2\sigma\Big(W_2\big(h_0 + U_1\sigma(W_1 h_0)\big)\Big),\\
&u(x,\theta) = a^T\big(h_2' + h_1' + h_0'\big),\\
&|u(x,\theta)| \leq |a^T h_2'| + |a^T h_1'| + |a^T h_0'|.
\end{aligned}$$

For h_1', and h_0' we have already calculated

$$|a^T h_0'| \leq \sum |a_j||v_{i,j}|,$$

and

$$|a^T h_1'| \leq K \sum_{m,k=1}^{n} |a_m||u_{mk}| \sum_{i=1}^{n}\sum_{j=1}^{d} |w_{ki}||v_{ij}|.$$

In the following the style of notation, w_{ij}^2 denotes the element of the ith row and jth column of the matrix W_2, and so the number two does not indicate a power of two but refers the index of the matrix, which in this case is 2. Similarly we have for example u_{ij}^1 for U_1. In the following we do not have any powers, it's all superscripts, except for the constants K and C_σ.

$$\begin{aligned}
a^T h_2'(x,\theta) &= a^T U_2\sigma\left(W_2\Big(U_1\sigma\big(W_1 V x\big) + V x\Big)\right)\\
&= \sum_{c,\xi,t,k} a_c u_{c\xi}^2 \sigma\Big(w_{\xi t}^2 u_{tk}^1 \sigma\big(\sum_{i,j} w_{ki}^1 v_{ij} x_j\big) + \sum_{i,j} w_{\xi i}^2 v_{ij} x_j\Big),
\end{aligned}$$

and so we have

$$|a^T h_2'(x,\theta)| \leq K^2 \sum_{c,\xi,t,k,i,j} |a_c||u_{c\xi}^2||w_{\xi t}^2||u_{tk}^1||w_{ki}^1||v_{ij}| + K \sum_{c,\xi,i,j} |a_c||u_{c\xi}^2||w_{\xi i}^2||v_{ij}|.$$

Again we have three possibilities: give a bound regardless of K, or depending if $K<1$ or $K\geq 1$. We can bound both summation terms by $\|\theta\|_{\mathcal{P}}$ and get

$$|h_2'(x,\theta)| \leq (K^2+K)Q.$$

If $K<1$, we have

$$\begin{aligned}
|a^T h_2'(x,\theta)| &\leq \sum_{c,\xi,t,k,i,j} |a_c||u_{c\xi}^2||w_{\xi t}^2||u_{tk}^1||w_{ki}^1||v_{ij}| + \sum_{c,\xi,i,j} |a_c||u_{c\xi}^2||w_{\xi i}^2||v_{ij}|\\
&\leq Q.
\end{aligned}$$

DOI https://doi.org/10.4208/cicp.OA-2023-0253 | Generated on 2024-12-19 11:09:00

If $K \geq 1$, we have

$$\begin{aligned}|a^T h_2'(x,\theta)| &\leq K^2 \sum_{c,\xi,t,k,i,j} |a_c||u^2_{c\xi}||w^2_{\xi t}||u^1_{tk}||w^1_{ki}||v_{ij}| + K^2 \sum_{c,\xi,i,j} |a_c||u^2_{c\xi}||w^2_{\xi i}||v_{ij}| \\ &\leq K^2 Q.\end{aligned}$$

In this proof, with ${}^*h^k_c$ we mean the vector which is the input for the kth hidden vector, input which comes as output when multiplied with matrix U_k, it is the same vector as g_l, for some reason we have used two notations. In the following we use the Einstein notation, $\sum_a w_{ia}v_{an} = w_{ia}v_{an}$, so that we sum over the index that appears twice. Another notation which we use here is $P_1^{x\to {}^*h^2_c}$, it denotes a collection of the paths from x to ${}^*h^2_c$

$$\begin{aligned}h_2 &= U_2\sigma(W_2({}^*h_1 + {}^*h_0)) \\ &= u^2_{c\xi}\sigma(w^2_{\xi j}{}^*h^1_j + w^2_{\xi j}{}^*h^0_j)\end{aligned}$$

from which we get

$$\begin{aligned}|{}^*h^2_c| &\leq K|u^2_{c\xi}||w^2_{\xi j}||{}^*h^1_j| + K|u^2_{c\xi}||w^2_{\xi j}||{}^*h^0_j| \\ &= K|u^2_{c\xi}||w^2_{\xi j}||u^1_{jf}\sigma(w^1_{f\phi}v_{\phi a}x_a)| + K|u^2_{c\xi}||w^2_{\xi j}||v_{ja}x_a| \\ &\leq K^2|u^2_{c\xi}||w^2_{\xi j}||u^1_{jf}||w^1_{f\phi}||v_{\phi a}| + K|u^2_{c\xi}||w^2_{\xi j}||v_{ja}| \\ &= K^2 \sum_{\theta_i \in P_2^{x\to {}^*h^2_c}} \|\theta_i\|_{\mathcal{P}} + K \sum_{\theta_i \in P_1^{x\to {}^*h^2_c}} \|\theta_i\|_{\mathcal{P}}.\end{aligned}$$

The initial case of the induction is proved. Let's move on.

Let's make the induction assumption and assume that

$$|{}^*h^L_c| \leq K^L \sum_{\theta_i \in P_L^{x\to {}^*h^L_c}} \|\theta_i\|_{\mathcal{P}} + \cdots + K \sum_{\theta_i \in P_1^{x\to {}^*h^L_c}} \|\theta_i\|_{\mathcal{P}}. \tag{A.5}$$

Now let's take the induction step and prove that the same holds for $L+1$:

$$\begin{aligned}|{}^*h^{L+1}_c| &= |u^{L+1}_{c\xi}\sigma(w^{L+1}_{\xi j}({}^*h^L_j + \cdots + {}^*h^0_j))| \\ &\leq |u^{L+1}_{c\xi}|K|w^{L+1}_{\xi j}|({}^*|h^L_j| + \cdots + |h^0_j|) \\ &\leq K|u^{L+1}_{c\xi}||w^{L+1}_{\xi j}|K^L \sum_{\theta_i \in P_L^{x\to {}^*h^L_j}} \|\theta_i\|_{\mathcal{P}} + \cdots \\ &\qquad + K|u^{L+1}_{c\xi}||w^{L+1}_{\xi j}|K \sum_{\theta_i \in P_1^{x\to {}^*h^L_j}} \|\theta_i\|_{\mathcal{P}} + K|u^{L+1}_{c\xi}||w^{L+1}_{\xi j}||v_j| \\ &\leq K^{L+1} \sum_{\theta_i \in P_{L+1}^{x\to {}^*h^{L+1}_c}} \|\theta_i\|_{\mathcal{P}} + \cdots + K \sum_{\theta_i \in P_1^{x\to {}^*h^{L+1}_c}} \|\theta_i\|_{\mathcal{P}}.\end{aligned}$$

With this result we can say

$$\begin{aligned}|u(x,\theta)| &= |a^T({}^*h_L+\cdots+{}^*h_0)| \\ &\leq \sum_c |a_c|\Big(K^L \sum_{\theta_i\in P_L^{x\to {}^*h_c^L}} \|\theta_i\|_{\mathcal{P}}+\cdots+K \sum_{\theta_i\in P_1^{x\to {}^*h_c^L}} \|\theta_i\|_{\mathcal{P}}\Big)+\cdots \\ &\quad +\sum_c |a_c|\Big(K^2 \sum_{\theta_i\in P_2^{x\to {}^*h_c^2}} \|\theta_i\|_{\mathcal{P}}+K \sum_{\theta_i\in P_1^{x\to {}^*h_c^2}} \|\theta_i\|_{\mathcal{P}}\Big) \\ &\quad +\sum_c |a_c|\Big(K \sum_{\theta_i\in P_1^{x\to {}^*h_c^1}} \|\theta_i\|_{\mathcal{P}}\Big)+\sum_{c,a} |a_c||v_{ca}|.\end{aligned}$$

If $K\geq 1$ then

$$\begin{aligned}|u(x,\theta)| &\leq K^L\Bigg[\sum_c |a_c|\Big(\sum_{\theta_i\in P_L^{x\to {}^*h_c^L}} \|\theta_i\|_{\mathcal{P}}+\cdots+\sum_{\theta_i\in P_1^{x\to {}^*h_c^L}} \|\theta_i\|_{\mathcal{P}}\Big)+\cdots \\ &\quad +\sum_c |a_c|\Big(K^2 \sum_{\theta_i\in P_2^{x\to {}^*h_c^2}} \|\theta_i\|_{\mathcal{P}}+\sum_{\theta_i\in P_1^{x\to {}^*h_c^2}} \|\theta_i\|_{\mathcal{P}}\Big) \\ &\quad +\sum_c |a_c|\Big(\sum_{\theta_i\in P_1^{x\to {}^*h_c^1}} \|\theta_i\|_{\mathcal{P}}\Big)+\sum_{c,a} |a_c||v_{ca}|\Bigg] \\ &\leq K^L\|\theta\|_{\mathcal{P}} \\ &\leq K^L Q.\end{aligned}$$

If $K<1$ then similarly we get

$$|u(x,\theta)|\leq \|\theta\|_{\mathcal{P}}\leq Q.$$

And if we do not have information about the K then

$$\begin{aligned}|u(x,\theta)| &\leq K^L \sum_{\theta_i\in P_L^{x\to u}} \|\theta_i\|_{\mathcal{P}}+\cdots+K \sum_{\theta_i\in P_1^{x\to u}} \|\theta_i\|_{\mathcal{P}}+\sum_{\theta_i\in P_0^{x\to u}} \|\theta_i\|_{\mathcal{P}} \\ &\leq K^L Q+\cdots+KQ+Q \\ &\leq Q(K^L+\cdots K+1).\end{aligned}$$

For the gradient, in the following the gradient is taken with respect to variable x, so $\nabla=\nabla_x$, we omit the subindex. Let's prove first the initial case.

$$u(x,\theta)=a^T{}^*h_2+a^T{}^*h_1+a^T{}^*h_0$$

DOI https://doi.org/10.4208/cicp.OA-2023-0253 | Generated on 2024-12-19 11:09:00

and therefore

$$\begin{aligned}\nabla u(x,\theta) &= a^T\nabla^* h_2 + a^T\nabla^* h_1 + a^T\nabla^* h_0,\\ \nabla^* h_0 &= \nabla V x = V,\\ \nabla^* h_1 &= \nabla U_1\sigma(W_1Vx)\\ &= U_1\mathrm{diag}\sigma'(W_1Vx)W_1V.\end{aligned}$$

And

$$\begin{aligned}\nabla^* h_2 &= \nabla U_2\sigma\Big(W_2\big(U_1\sigma(W_1Vx)+Vx\big)\Big)\\ &= U_2\mathrm{diag}\sigma'\Big(W_2\big(U_1\sigma(W_1Vx)+Vx\big)\Big)\Big[W_2U_1\mathrm{diag}\sigma'(W_1Vx)W_1V+W_2V\Big].\end{aligned}$$

Let's now calculate the partial derivative ∂_{x_i} for one element of ${}^*h_2$, let it be c'th element ${}^*h_c^2$.

$$\partial_{x_i}{}^*h_c^2 = u_{ct}^2\sigma'(w_{t\phi}^2\cdots)\Big[w_{t\phi}^2u_{\phi\xi}^1\sigma'(w_{\xi a}^1v_{ab}x_b)w_{\xi a}^1v_{ai}+w_{t\phi}^2v_{\phi i}\Big],$$

which is essentially ic element of the matrix above. Observe that again we are using Eistein notation, and summation is done over all other indeces except c and i. From this we deduce

$$\begin{aligned}|\partial_{x_i}{}^*h_c^2| &\le |u_{ct}^2|C_\sigma\Big[|w_{t\phi}^2||u_{\phi\xi}^1|C_\sigma|w_{\xi a}^1||v_{ai}|+|w_{t\phi}^2||v_{\phi i}|\Big],\\ &= C_\sigma^2|u_{ct}^2||w_{t\phi}^2||u_{\phi\xi}^1||w_{\xi a}^1||v_{ai}|+C_\sigma|u_{ct}^2||w_{t\phi}^2||v_{\phi i}|\\ &= (C_\sigma)^2\sum_{\theta_k\in P_2^{v_i\to{}^*h_c^2}}\|\theta_k\|_{\mathcal{P}}+C_\sigma\sum_{\theta_k\in P_1^{v_i\to{}^*h_c^2}}\|\theta_k\|_{\mathcal{P}}.\end{aligned}$$

The initial step is proved. As an induction assumption, let's assume

$$|\partial^*h_c^L|\le(C_\sigma)^L\sum_{\theta_k\in P_L^{v_i\to{}^*h_c^L}}\|\theta_k\|_{\mathcal{P}}+\cdots+C_\sigma\sum_{\theta_k\in P_1^{v_i\to{}^*h_c^L}}\|\theta_k\|_{\mathcal{P}}.$$

Now let's take the induction step and prove that the same holds for $L+1$:

$$\begin{aligned}|\partial_{x_i}{}^*h_c^{L+1}| &= |u_{c\xi}^{L+1}\sigma'(w_{\xi j}^{L+1}(\partial_{x_i}{}^*h_j^L+\cdots+\partial_{x_i}{}^*h_j^0))|\\ &\le |u_{c\xi}^{L+1}|C_\sigma|w_{\xi j}^{L+1}|(|\partial_{x_i}{}^*h_j^L|+\cdots+|\partial_{x_i}h_j^0|)\\ &\le C_\sigma|u_{c\xi}^{L+1}||w_{\xi j}^{L+1}|(C_\sigma)^L\sum_{\theta_i\in P_L^{v_i\to{}^*h_j^L}}\|\theta_k\|_{\mathcal{P}}+\cdots\\ &\qquad+C_\sigma|u_{c\xi}^{L+1}||w_{\xi j}^{L+1}|C_\sigma\sum_{\theta_k\in P_1^{v_i\to{}^*h_j^L}}\|\theta_k\|_{\mathcal{P}}+C_\sigma|u_{c\xi}^{L+1}||w_{\xi j}^{L+1}||v_j|\\ &\le (C_\sigma)^{L+1}\sum_{\theta_k\in P_{L+1}^{v_i\to{}^*h_c^{L+1}}}\|\theta_k\|_{\mathcal{P}}+\cdots+C_\sigma\sum_{\theta_k\in P_1^{v_i\to{}^*h_c^{L+1}}}\|\theta_k\|_{\mathcal{P}}.\end{aligned}$$

With this information, we can calculate the ℓ^1-norm of ∇u. Remember that

$$\|\nabla u(x,\theta)\|_1 = \sum_i |\partial_{x_i} u(x,\theta)|,$$
$$\partial_{x_i} u(x,\theta) = a^T(\partial_{x_i}{}^* h_L + \cdots + \partial_{x_i}{}^* h_0).$$

From which we get

$$\begin{aligned}
|\partial_{x_i} u(x,\theta)| &\le |a^T|(|\partial_{x_i}{}^* h_L| + \cdots + |\partial_{x_i}{}^* h_0|) \\
&\le \sum_c \Big(|a_c||\partial_{x_i}{}^* h_c^L|\Big) + \cdots + \sum_c \Big(|a_c||\partial_{x_i}{}^* h_c^0|\Big) \\
&\le \sum_c |a_c| \Big((C_\sigma)^L \sum_{\theta_k \in P_L^{v_i \to {}^* h_c^L}} \|\theta_k\|_{\mathcal{P}} + \cdots + C_\sigma \sum_{\theta_k \in P_1^{v_i \to {}^* h_c^L}} \|\theta_k\|_{\mathcal{P}}\Big) + \cdots \\
&\quad + \sum_c |a_c| C_\sigma \sum_{\theta_k \in P_1^{v_i \to {}^* h_c^1}} \|\theta_k\|_{\mathcal{P}} + \sum_c |a_c||v_{ci}|.
\end{aligned}$$

And so we have

$$\begin{aligned}
\|\nabla u(x,\theta)\|_1 \le \sum_i \Big[\sum_c |a_c| \Big((C_\sigma)^L \sum_{\theta_k \in P_L^{v_i \to {}^* h_c^L}} \|\theta_k\|_{\mathcal{P}} + \cdots \\
+ C_\sigma \sum_{\theta_k \in P_1^{v_i \to {}^* h_c^L}} \|\theta_k\|_{\mathcal{P}} \Big) + \cdots \\
+ \sum_c |a_c| C_\sigma \sum_{\theta_k \in P_1^{v_i \to {}^* h_c^1}} \|\theta_k\|_{\mathcal{P}} + \sum_c |a_c||v_{ci}| \Big].
\end{aligned}$$

If $C_\sigma \ge 1$ then we have

$$\begin{aligned}
\|\nabla u(x,\theta)\|_1 &\le (C_\sigma)^L \sum_i \Big[\sum_c |a_c| \Big(\sum_{\theta_k \in P_L^{v_i \to {}^* h_c^L}} \|\theta_k\|_{\mathcal{P}} + \cdots \\
&\qquad + \sum_{\theta_k \in P_1^{v_i \to {}^* h_c^L}} \|\theta_k\|_{\mathcal{P}} \Big) + \cdots \\
&\qquad + \sum_c |a_c| \sum_{\theta_k \in P_1^{v_i \to {}^* h_c^1}} \|\theta_k\|_{\mathcal{P}} + \sum_c |a_c||v_{ci}| \Big] \\
&= (C_\sigma)^L \|\theta\|_{\mathcal{P}} \\
&< (C_\sigma)^L Q.
\end{aligned}$$

If $C_\sigma < 1$ then with similar deduction we get

$$\|\nabla u(x,\theta)\|_1 < Q.$$

DOI https://doi.org/10.4208/cicp.OA-2023-0253 | Generated on 2024-12-19 11:09:00

Without the information of the magnitude of C_σ we can say

$$\|\nabla u(x,\theta)\|_1 < Q((C_\sigma)^L + \cdots C_\sigma + 1).$$

As we know $\|\cdot\|_2 \leq \|\cdot\|_1$, the claim for the ℓ^2-norm $\|\nabla u(x,\theta)\|_2^2$ follows. □

Theorem A.1 (Statistical error of DRM for shallow neural networks)**.** *Let $u : \Omega \to \mathbb{R}$, $B(0,1) \subset \mathbb{R}^d$, be a shallow neural network $u(x,\theta) \in \mathcal{F}_Q$ with bounded path norm $\|\theta\|_{\hat{\mathcal{P}}} < Q$, $Q > 0$, with an approachable activation function σ. Let σ be M-Lipschitz, and let σ' be K-Lipschitz, with some $M > 0$, and $K > 0$. Let $\delta > 0$ be a confidence parameter. With probability $1-\delta$, the statistical error of the Deep Ritz method is bounded as*

$$\begin{aligned}
&|\mathcal{L}(u) - \widehat{\mathcal{L}}(u,n,m)| \\
\leq &|\Omega| \left(4\gamma(\sigma) Q \sqrt{\frac{2\ln(2d+2)}{n}} + 3\left(\frac{1}{2}K^2Q^2 + MQ\right)\sqrt{\frac{2\ln(4/\delta)}{n}} \right) \\
&+ \frac{\lambda}{2}|\partial\Omega| \left(4\gamma(\sigma) Q \sqrt{\frac{2\ln(2d+2)}{m}} + 3M^2Q^2 \sqrt{\frac{2\ln(4/\delta)}{m}} \right).
\end{aligned}$$

where $\gamma(\sigma)$ is a constant depending only in σ, defined in Definition 2.1*, n is the number of sample points in the domain denoted as $\Omega := B(0,1)$, m is the number of sample points on the boundary $\partial\Omega$, $|\Omega|$ is the Lebesgue measure of the domain, and $|\partial\Omega|$ is the Hausdorff measure of the boundary.*

Proof. Let X be an uniform random variable in the domain Ω and let Y be an uniform random variable on the boundary $\partial\Omega$. Let X_i, $i = 1,\cdots,n$; be a sample of the random variable of X, and let Y_i, $i = 1,\cdots,n$; be a sample of random variable Y.

The statistical error can be divided into two terms, due to integration over the domain and over the boundary. According to (1.5) and (1.6) we get

$$\begin{aligned}
&|\mathcal{L}(u) - \widehat{\mathcal{L}}(u,n,m)| \\
\leq &\left| |\Omega| \mathbb{E}_X \left[\frac{\|\nabla u(X)\|_2^2}{2} - u(X) \right] - \frac{|\Omega|}{n} \sum_{i=1}^{n} \left[\frac{\|\nabla u(X_i)\|_2^2}{2} - u(X_i) \right] \right| \\
&+ \left| \frac{\lambda}{2} |\partial\Omega| \mathbb{E}_Y[(u(Y)^2)] - \frac{\lambda}{2} \frac{|\partial\Omega|}{M} \sum_{j=1}^{m} [(u(Y_j)^2)] \right|.
\end{aligned}$$

We dropped the trace operator away as the function u is continuous on the boundary, trace operator would be just an identity. From here we continue by bounding the two terms.

In the spirit of Lemma 2.2, the function integrated in the Deep Ritz method's loss function is

$$\ell(x,u) = \frac{1}{2}\|\nabla u(x,\theta)\|_2^2 - u(x,\theta). \tag{A.6}$$

And by Corollary 2.2, the function $l(x,u)$ is bounded by

$$|\ell(x,u)| = \left|\frac{1}{2}\|\nabla u(x,\theta)\|_2^2 - |u(x,\theta)|\right| < \frac{1}{2}K^2Q^2 + MQ. \tag{A.7}$$

Therefore, we can use Lemma 2.2. The first term is bounded with probability $1-\delta$ as

$$\begin{aligned}
&\left||\Omega|\mathbb{E}_X\left[\frac{\|\nabla u(X)\|_2^2}{2} - u(X)\right] - \frac{|\Omega|}{n}\sum_{i=1}^{n}\left[\frac{\|\nabla u(X_i)\|_2^2}{2} - u(x_i)\right]\right| \\
&= |\Omega|\left|\mathbb{E}_X\left[\frac{\|\nabla u(X)\|_2^2}{2} - u(X)\right] - \frac{1}{n}\sum_{i=1}^{n}\left[\frac{\|\nabla u(X_i)\|_2^2}{2} - u(x_i)\right]\right| \\
&\leq |\Omega|\left(2\widehat{\mathscr{R}}(\ell\circ\mathcal{F}_Q) + 3\left(\frac{1}{2}K^2Q^2 + MQ\right)\sqrt{\frac{2\ln(4/\delta)}{n}}\right).
\end{aligned}$$

Similarly we can bound the boundary term, using Corollary 2.2 (or just Lemma A.3) we can apply Lemma 2.2

$$\begin{aligned}
&\left|\frac{\lambda}{2}|\partial\Omega|\mathbb{E}_Y[(u(Y)^2)] - \frac{\lambda}{2}\frac{|\partial\Omega|}{m}\sum_{j=1}^{m}[(u(y_j)^2)]\right| \\
&= \frac{\lambda}{2}|\partial\Omega|\left|\mathbb{E}_Y[(u(Y)^2)] - \frac{1}{m}\sum_{j=1}^{m}[(u(y_j)^2)]\right| \\
&\leq \frac{\lambda}{2}|\partial\Omega|\left(2\widehat{\mathscr{R}}(\ell\circ\mathcal{F}_Q) + 3M^2Q^2\sqrt{\frac{2\ln(4/\delta)}{m}}\right),
\end{aligned}$$

where $|\partial\Omega|$ is the Hausdorff measure of the boundary.

In Appendix C we show that the loss function is Lipschitz, Theorem C.2. We do not specify the Lipschitz constant, but we can say that there exists some $C>0$ which works as a Lipschitz constant for the ℓ. From Theorem 2.2 we know that $\widehat{\mathscr{R}}(\mathcal{F}_Q) \leq 2\gamma(\sigma)Q\sqrt{\frac{2\ln(2d+2)}{n}}$ for shallow neural networks and with an approachable activation with path norm bounded by $Q>0$. Therefore with Lemma 2.3 and Theorem 2.2 we get

$$\begin{aligned}
&\left||\Omega|\mathbb{E}_X\left[\frac{\|\nabla u(X)\|_2^2}{2} - u(X)\right] - \frac{|\Omega|}{n}\sum_{i=1}^{n}\left[\frac{\|\nabla u(X_i)\|_2^2}{2} - u(x_i)\right]\right| \\
&\leq C|\Omega|\left(4\gamma(\sigma)Q\sqrt{\frac{2\ln(2d+2)}{n}} + 3\left(\frac{1}{2}K^2Q^2 + MQ\right)\sqrt{\frac{2\ln(4/\delta)}{n}}\right).
\end{aligned}$$

On the boundary the loss function is just a polynomial, it is clearly Lipschitz as the values of the neural network is bounded above. Therefore there exists a Lipschitz constant $C>0$ for the loss on the boundary. We do not specify the constant exactly, and we use the same

letter as what was used for the case inside-the-domain (let C be the one which has greater Lipshitz constant). Using the Lemma 2.3 and Theorem 2.2 we get we get a bound for the boundary term

$$\begin{aligned}&\left|\frac{\lambda}{2}|\partial\Omega|\mathbb{E}_Y[(u(Y)^2)]-\frac{\lambda}{2}\frac{|\partial\Omega|}{m}\sum_{j=1}^{m}[(u(y_j)^2)]\right|\\ \leq& C\frac{\lambda}{2}|\partial\Omega|\left(4\gamma(\sigma)Q\sqrt{\frac{2\ln(2d+2)}{m}}+3M^2Q^2\sqrt{\frac{2\ln(4/\delta)}{m}}\right),\end{aligned}$$

where $|\partial\Omega|$ is the Hausdorff measure of the boundary, $\gamma(\sigma)$ is defined in the Definition 2.1, m is the number of sample points, and δ is a confidence parameter. Together these two bounds give the statistical error of Deep Ritz method, namely, with probability $1-\delta$ we have

$$\begin{aligned}&|\mathcal{L}(u)-\widehat{\mathcal{L}}(u,n,m)|\\ \leq& C|\Omega|\left(4\gamma(\sigma)Q\sqrt{\frac{2\ln(2d+2)}{n}}+3\left(\frac{1}{2}K^2Q^2+MQ\right)\sqrt{\frac{2\ln(4/\delta)}{n}}\right)\\ &+C\frac{\lambda}{2}|\partial\Omega|\left(4\gamma(\sigma)Q\sqrt{\frac{2\ln(2d+2)}{m}}+3M^2Q^2\sqrt{\frac{2\ln(4/\delta)}{m}}\right).\end{aligned}$$

This completes the proof. □

Theorem A.2 (Statistical error of the DRM for residual neural networks)**.** *Let $u_\theta:\Omega\to\mathbb{R}$ be a residual neural network $u(x,\theta)\in\mathcal{F}_Q$, with $L\in\mathbb{N}$ skip connections, with a bounded path norm $\|\theta\|_{\tilde{\mathcal{P}}}<Q$, $Q>0$, and with an approachable activation function σ. Let σ be M-Lipschitz, $M>0$. Let σ' be bounded by $|\sigma'(z)|<C_\sigma$, $C_\sigma>0$, for all $z\in\mathbb{R}$. Let $\delta>0$ be a confidence parameter. With probability $1-\delta$, the statistical error of the Deep Ritz method is bounded as*

$$\begin{aligned}&|\mathcal{L}(u)-\widehat{\mathcal{L}}(u,n,m)|\\ \leq& |\Omega|\left(4c_\sigma Q\sqrt{\frac{2\ln(2d+2)}{n}}+3\left(\frac{1}{2}(C_\sigma\vee 1)^{2L}Q^2+(K\vee 1)^L Q\right)\sqrt{\frac{2\ln(4/\delta)}{n}}\right)\\ &+\frac{\lambda}{2}|\partial\Omega|\left(2c_\sigma Q\sqrt{\frac{2\ln(2d+2)}{m}}+3(K\vee 1)^{2L}Q^2\sqrt{\frac{2\ln(4/\delta)}{m}}\right),\end{aligned}$$

where n is the number of sample points in the domain Ω, m is the number of sample points on the boundary $\partial\Omega$, c_σ is defined in Definition 2.7.

Proof. The proof will is similar to the proof of Theorem A.1. Let X be an uniform random variable in the domain Ω, and let Y be an uniform random variable on the boundary $\partial\Omega$.

Let X_i, $i=1,\cdots,n$; be a sample of the random variable of X and let Y_i, $i=1,\cdots,n$; be a sample of the random variable Y.

The statistical error can be divided into two terms, due to integration over the domain and over the boundary. According to (1.5) and (1.6) we get

$$\begin{aligned}
&|\mathcal{L}(u)-\widehat{\mathcal{L}}(u,n,m)|\\
\leq&\left||\Omega|\mathbb{E}_X\left[\frac{\|\nabla u(X)\|_2^2}{2}-u(X)\right]-\frac{|\Omega|}{n}\sum_{i=1}^n\left[\frac{\|\nabla u(X_i)\|_2^2}{2}-u(x_i)\right]\right|\\
&+\left|\frac{\lambda}{2}|\partial\Omega|\mathbb{E}_Y[(u(Y)^2)]-\frac{\lambda}{2}\frac{|\partial\Omega|}{M}\sum_{j=1}^m[(u(Y_j)^2)]\right|.
\end{aligned}$$

We dropped the trace operator away as the function is continuous on the boundary, trace operator would be just an identity. From here we continue by bounding the two terms. In the spirit of Lemma 2.2, the function integrated in the Deep Ritz method's loss function is

$$\ell(x,u)=\frac{1}{2}\|\nabla u(x,\theta)\|_2^2-u(x,\theta). \tag{A.8}$$

And by Corollary 2.3, the function $\ell(x,u)$ is bounded by

$$|\ell(x,u)|=\left|\frac{1}{2}\|\nabla u(x,\theta)\|^2-u(x,\theta)\right|<\frac{1}{2}(C_\sigma\vee 1)^{2L}Q^2+(K\vee 1)^LQ. \tag{A.9}$$

Therefore, we can use Lemma 2.2. With probability $1-\delta$, the first term is bounded by

$$\begin{aligned}
&\left||\Omega|\mathbb{E}_X\left[\frac{\|\nabla u(X)\|_2^2}{2}-u(X)\right]-\frac{|\Omega|}{n}\sum_{i=1}^n\left[\frac{\|\nabla u(X_i)\|_2^2}{2}-u(x_i)\right]\right|\\
=&|\Omega|\left|\mathbb{E}_X\left[\frac{\|\nabla u(X)\|_2^2}{2}-u(X)\right]-\frac{1}{n}\sum_{i=1}^n\left[\frac{\|\nabla u(X_i)\|_2^2}{2}-u(x_i)\right]\right|\\
\leq&|\Omega|\left(2\widehat{\mathscr{R}}(\ell\circ\mathcal{F}_Q)+3\left(\frac{1}{2}(C_\sigma\vee 1)^{2L}Q^2+(K\vee 1)^LQ\right)\sqrt{\frac{2\ln(4/\delta)}{n}}\right).
\end{aligned}$$

Similarly we can bound the boundary term, using Corollary 2.3 (or just Lemma A.4) we know

$$|u(Y,\theta)|^2<(K\vee 1)^{2L}Q^2,$$

on the boundary $\partial\Omega$. Therefore we can apply Lemma 2.2

$$\begin{aligned}
&\left|\frac{\lambda}{2}|\partial\Omega|\mathbb{E}_Y[(u(Y)^2)]-\frac{\lambda}{2}\frac{|\partial\Omega|}{m}\sum_{j=1}^{m}[(u(y_j)^2)]\right| \\
=&\frac{\lambda}{2}|\partial\Omega|\left|\mathbb{E}_Y[(u(Y)^2)]-\frac{1}{m}\sum_{j=1}^{m}[(u(y_j)^2)]\right| \\
\leq&\frac{\lambda}{2}|\partial\Omega|\left(\widehat{\mathscr{R}}(\ell\circ\mathcal{F}_Q)+3(K\vee 1)^{2L}Q^2\sqrt{\frac{2\ln(4/\delta)}{m}}\right),
\end{aligned}$$

where $|\partial\Omega|$ is the Hausdorff measure of the boundary.

In Appendix C we have shown in Theorem C.3 that the loss function is Lipshitz in the case of residual neural networks with bounded path norm. We have not specified the exact value of a Lipschitz constant, but we know that such a constant $C>0$ exists. Similarly on the boundary, as we know that the neural network is bounded, and therefore we can say that the function $u(x\theta)^2$ on boundary is Lipschitz. Let the Lipschitz constant $C>0$ be the one that is a maximum of the two Lipschitz constants. From Theorem 2.3 we know that $\widehat{\mathscr{R}}(\mathcal{F}_Q)\leq c_\sigma Q\sqrt{\frac{2\ln(2d+2)}{n}}$ for residual neural networks with an approachable activation, and bounded path norm. Therefore with Lemma 2.3 and Theorem 2.3 we get

$$\begin{aligned}
&\left||\Omega|\mathbb{E}_X\left[\frac{\|\nabla u(X)\|_2^2}{2}-u(X)\right]-\frac{|\Omega|}{n}\sum_{i=1}^{n}\left[\frac{\|\nabla u(X_i)\|_2^2}{2}-u(x_i)\right]\right| \\
\leq&C|\Omega|\left(2c_\sigma Q\sqrt{\frac{2\ln(2d+2)}{n}}+3\left(\frac{1}{2}(C_\sigma\vee 1)^{2L}Q^2+(K\vee 1)^L Q\right)\sqrt{\frac{2\ln(4/\delta)}{n}}\right).
\end{aligned}$$

Using the same deduction we get a bound for the boundary term

$$\begin{aligned}
&\left|\frac{\lambda}{2}|\partial\Omega|\mathbb{E}_Y[(u(Y)^2)]-\frac{\lambda}{2}\frac{|\partial\Omega|}{m}\sum_{j=1}^{m}[(u(y_j)^2)]\right| \\
\leq&C\frac{\lambda}{2}|\partial\Omega|\left(c_\sigma Q\sqrt{\frac{2\ln(2d+2)}{m}}+3(K\vee 1)^{2L}Q^2\sqrt{\frac{2\ln(4/\delta)}{m}}\right),
\end{aligned}$$

where $|\partial\Omega|$ is the Hausdorff measure of the boundary, $\gamma(\sigma)$ is defined in Definition 2.1, m is the number of sample points, and δ is a confidence parameter. Together these two bounds give the statistical error of Deep Ritz method, namely, with probability $1-\delta$ we

have

$$\begin{aligned}&|\mathcal{L}(u)-\widehat{\mathcal{L}}(u,n,m)|\\&\leq C|\Omega|\left(2c_\sigma Q\sqrt{\frac{2\ln(2d+2)}{n}}+3\left(\frac{1}{2}(C_\sigma\vee 1)^{2L}Q^2+(K\vee 1)^L Q\right)\sqrt{\frac{2\ln(4/\delta)}{n}}\right)\\&\quad+C\frac{\lambda}{2}|\partial\Omega|\left(c_\sigma Q\sqrt{\frac{2\ln(2d+2)}{m}}+3(K\vee 1)^{2L}Q^2\sqrt{\frac{2\ln(4/\delta)}{m}}\right).\end{aligned}$$

This completes the proof. □

B Activation functions

Instead with smooth activation functions like tanh, or some smooth version of the Relu, we were able to minimize the loss function and approximate the target function. More about activation functions in the appendix. With Relu^2 we mean the activation function

$$\sigma(x)=\max\{x,0\}^2, \tag{B.1}$$

for example in [1] the authors used a similar smoothed version of the Relu activation function. The polynomially smoothed Relu is a following activation function: $\sigma:\mathbb{R}\to\mathbb{R}$, such that

$$\sigma(x)=\begin{cases}0 & \text{when } x<-\epsilon,\\ x(\frac{x}{\epsilon}+1)^3 & \text{when } -\epsilon\leq x\leq 0,\\ x & \text{when } x>0,\end{cases}$$

more about this activation function in the appendix. Disadvantage of the activation function Relu^2 is that it is difficult, if not possible, according to our experiments, to build a deep residual neural net with Relu^2 as the activation function. If one were to train a deep residual neural net then it is better to use a Relu-a-like activation functions like the above mentioned polynomially smoothed Relu, Gelu [46], or the Swish activation function [47].

We present an activation function of the following form: let $\epsilon>0$ be a small number, $\sigma:\mathbb{R}\to\mathbb{R}$,

$$\sigma(x)=\begin{cases}0 & \text{when } x<-\epsilon,\\ x(\frac{x}{\epsilon}+1)^3 & \text{when } -\epsilon\leq x\leq 0,\\ x & \text{when } x>0.\end{cases}$$

The idea of the activation function is that we would like to have an activation function similar to the ReLu [48], [49], but without the problems at the origin $x=0$. Namely, the ReLu activation function does not have a derivative at $x=0$. The activation function presented here is exactly the same as the ReLu when $x\geq 0$, and $x\leq-\epsilon$, the difference is

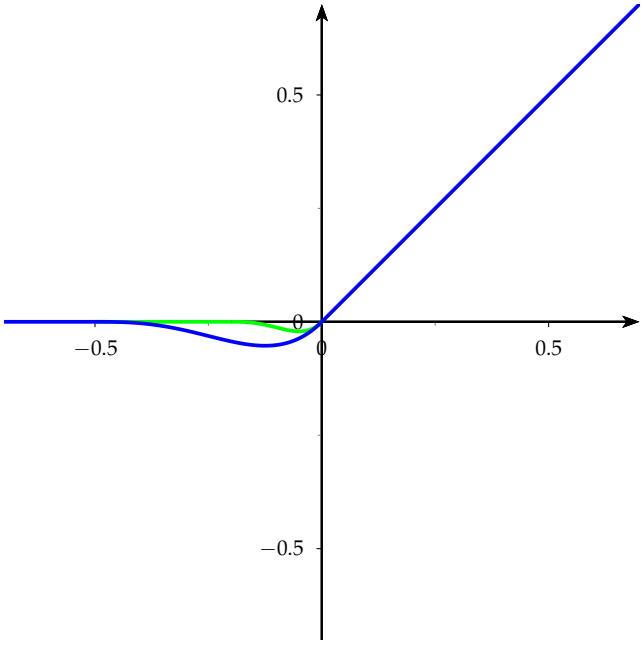

Figure 1: The activation function with $\epsilon=0{,}2$ and $\epsilon=0{,}5$.

in $-\epsilon<x<0$ where σ is polynomial. The derivative of the activation function is

$$\sigma'(x)=\begin{cases}0 & \text{when } x<-\epsilon,\\ (\frac{x}{\epsilon}+1)^3+\frac{3x}{\epsilon}(\frac{x}{\epsilon}+1) & \text{when } -\epsilon\le x\le 0,\\ 1 & \text{when } x>0.\end{cases}$$

The derivative is well defined for all $x\in\mathbb{R}$ and the derivative is continuous.

Second order derivative of σ is the following function.

$$\sigma''(x)=\begin{cases}0 & \text{when } x<-\epsilon,\\ \frac{6}{\epsilon}(\frac{x}{\epsilon}+1)^2+\frac{6x}{\epsilon}(\frac{x}{3}+1) & \text{when } -\epsilon\le x<0,\\ 0 & \text{when } x>0.\end{cases}\tag{B.2}$$

The ReLu function, $\mathrm{ReLu}:\mathbb{R}\to\mathbb{R}$, can be expressed as $\mathrm{ReLu}(x)=\max\{x,0\}$. Similarly the here proposed activation function can be represented as $\sigma(x)=\min\{p(x),\mathrm{ReLu}(x)\}$, where p is the polynomial $p(x)=x(\frac{x}{\epsilon}+1)^3$.

In Fig. 1 we have plotted the activation function σ with two different values of ϵ, namely, with $\epsilon=0{,}2$ and $\epsilon=0{,}5$. Similarly in Fig. 2 we have plotted the gradient of σ with the same values of ϵ as in previous figure. From the figures we see that how σ approaches the ReLu function as ϵ goes to zero. On the other hand, the derivatives do not converge as ϵ decreases to zero. What is interesting though is that, the derivative does not explode

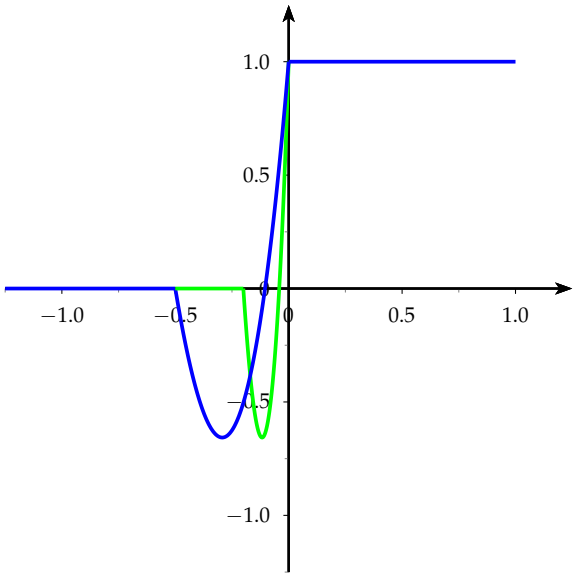

Figure 2: Derivative of the activation function with $\epsilon = 0{,}2$ and $\epsilon = 0{,}5$.

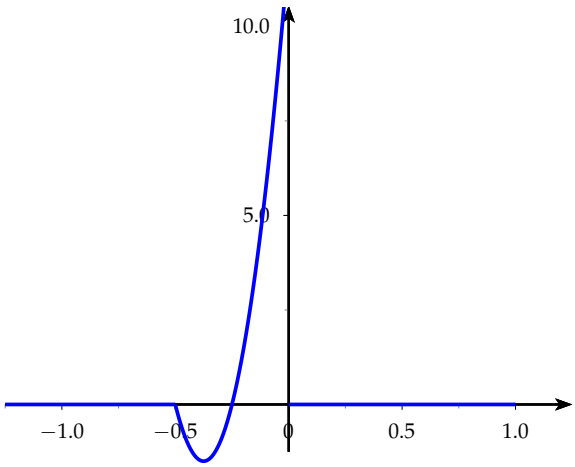

Figure 3: Second order derivative of the activation function σ with $\epsilon = 0{,}5$.

or behave in extreme way with different values of ϵ, but the derivative is rather nicely behaving function. For the sake of clarity, in Fig. 4 we have plotted the Relu function; and in Fig. 5 we have plotted the gradient of the Relu function. What is important of the gradient of Relu is that it is singular at the origin, the derivative is a dirac mass at

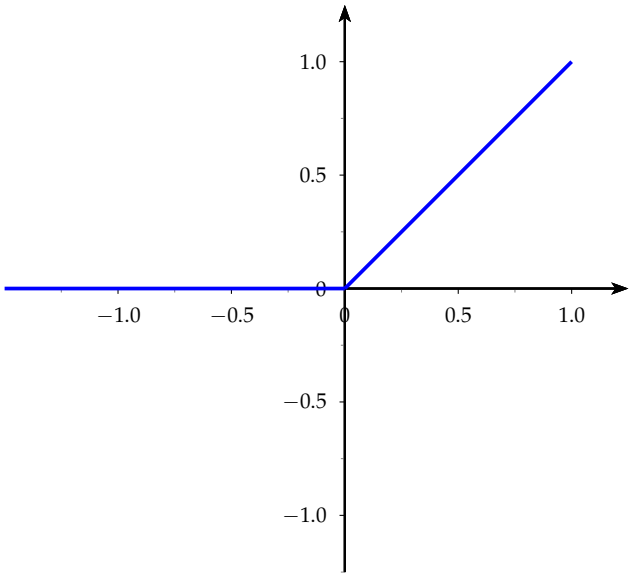

Figure 4: The Relu function.

origin, and we have tried to indicate that in the figure with a vertical line starting from the origin. From the Fig. 3 we see that the second order derivative of σ explodes near origin, and is discontinuous at origin. This is still better than with Relu as the second order derivative of Relu at origin is derivative of dirac mass.

So with the activation function σ we try to bypass the dirac mass on the derivative of relu, but maintaining as much as possible of the good qualities of the Relu function.

Theorem B.1. *The activation function σ satisfies the conditions in Definition* 2.1 *and is thus approachable.*

Proof. For continuity of σ, we see that it is piecewisely defined function. Constant function, identity function, and polynomial $x(x/\epsilon+1)^3$ are continuous, therefore only possible points where σ might be noncontinuous are $x=-\epsilon$ and $x=0$. At $x=-\epsilon$ polynomial $x(x/\epsilon+1)^3$ gets a value 0, so σ is continuous at $x=-\epsilon$. At $x=0$ polynomial $x(x/\epsilon+1)^3$ gets value 0, which is the same for identity function x, and so σ is continuous at the origin as well.

The derivative of σ is continuous and we can formally differentiate the function $\sigma'(x)$. The function we get is a piecewise continuous function defined in (B.3). On intervals $x<-\epsilon$, $-\epsilon<x<0$, and $x>0$ the function $\sigma''(x)$ is well defined. At $x=0$ the second order derivative is not well defined, but this single point has Lebesgue measure zero, and therefore the $\sigma''(x)$ is well defined in the weak sense.

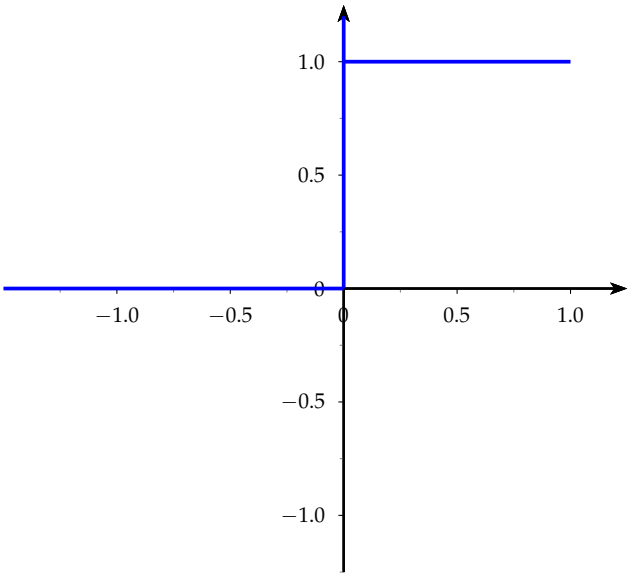

Figure 5: Gradient of the Relu function.

Let's recall that the function σ'' is a following function.

$$\sigma''(x)=\begin{cases}0 & \text{when } x<-\epsilon,\\ \frac{6}{\epsilon}(\frac{x}{\epsilon}+1)^2+\frac{6x}{\epsilon}(\frac{x}{3}+1) & \text{when } -\epsilon\leq x<0,\\ 0 & \text{when } x>0.\end{cases} \tag{B.3}$$

Function is compactly supported and in its support, the function is polynomial, continuous, and bounded. Therefore it is also locally Riemann integrable.

It is not difficult to see that $\gamma(\sigma)$ will be bounded. The minimal value for σ and σ' is 0, therefore $\inf_{x\in\mathbb{R}}g(x)$ for σ is 0, and as $\sigma''(x)$ is compactly supported function, being polynomial in its support, it is not hard to say that $\gamma_0(\sigma)<\infty$. But let's not settle for such an grude estimate, but let's calculate the value $\gamma(\sigma)$

$$\begin{aligned}\gamma_0(\sigma)&=\int_{\mathbb{R}}|\sigma''(x)|(|x|+1)dx=\int_{-\epsilon}^{0}|\sigma''(x)|(|x|+1)dx\\&=\int_{-\epsilon}^{0}\left|\frac{6}{\epsilon}\Big(\frac{x}{\epsilon}+1\Big)^2+\frac{6x}{\epsilon^2}\Big(\frac{x}{\epsilon}+1\Big)\right|\Big(|x|+1\Big)dx\end{aligned}$$

$$
\begin{aligned}
&= \int_{-\epsilon}^{0} \frac{6}{\epsilon} \left| \frac{2}{\epsilon^2} x^2 + \frac{3}{\epsilon} x + 1 \right| \Big(|x| + 1 \Big) dx \\
&= \int_{-\epsilon}^{-\epsilon/2} -\frac{6}{\epsilon} \Big(\frac{2}{\epsilon^2} x^2 + \frac{3}{\epsilon} x + 1 \Big) \Big(-x + 1 \Big) dx \\
&\quad + \int_{-\epsilon/2}^{0} \frac{6}{\epsilon} \Big(\frac{2}{\epsilon^2} x^2 + \frac{3}{\epsilon} x + 1 \Big) \Big(-x + 1 \Big) dx.
\end{aligned}
$$

The first integral is

$$
\begin{aligned}
&\int_{-\epsilon}^{-\epsilon/2} -\frac{6}{\epsilon} \Big(\frac{2}{\epsilon^2} x^2 + \frac{3}{\epsilon} x + 1 \Big) \Big(-x + 1 \Big) dx \\
&= -\frac{6}{\epsilon} \int_{-\epsilon}^{-\epsilon/2} \Big(\frac{2}{\epsilon^2} (-x^3) + \frac{3}{\epsilon} (-x^2) - x + \frac{2}{\epsilon^2} x^2 + \frac{3}{\epsilon} x + 1 \Big) dx \\
&= -\frac{6}{\epsilon} \int_{-\epsilon}^{-\epsilon/2} \Big(-\frac{2}{\epsilon^2} x^3 - \frac{3}{\epsilon} x^2 - x + \frac{2}{\epsilon^2} x^2 + \frac{3}{\epsilon} x + 1 \Big) dx \\
&= -\frac{6}{\epsilon} \Big|_{-\epsilon}^{-\epsilon/2} \Big(-\frac{2}{\epsilon^2} \cdot \frac{1}{4} x^4 - \frac{3}{\epsilon} \cdot \frac{1}{3} x^3 - \frac{1}{2} x^2 + \frac{2}{\epsilon^2} \cdot \frac{1}{3} x^3 + \frac{3}{\epsilon} \cdot \frac{1}{2} x^2 + x \Big) \\
&= -\frac{6}{\epsilon} \Bigg(-\frac{2}{\epsilon^2} \cdot \frac{1}{4} \Big(-\frac{\epsilon}{2} \Big)^4 - \frac{1}{\epsilon} \Big(-\frac{\epsilon}{2} \Big)^3 - \frac{1}{2} \Big(-\frac{\epsilon}{2} \Big)^2 \\
&\qquad + \frac{2}{\epsilon^2} \cdot \frac{1}{3} \cdot \Big(-\frac{\epsilon}{2} \Big)^3 + \frac{3}{\epsilon} \cdot \frac{1}{2} \cdot \Big(-\frac{\epsilon}{2} \Big)^2 + \Big(-\frac{\epsilon}{2} \Big) \\
&\qquad - \Big[-\frac{2}{\epsilon^2} \cdot \frac{1}{4} \cdot (-\epsilon)^4 - \frac{1}{3} \cdot (-\epsilon)^3 - \frac{1}{2} \cdot (-\epsilon)^2 \\
&\qquad + \frac{2}{\epsilon^2} \cdot \frac{1}{3} \cdot (-\epsilon)^3 + \frac{3}{\epsilon} \cdot \frac{1}{2} \cdot (-\epsilon)^2 + (-\epsilon) \Big] \Bigg) \\
&= -\frac{6}{\epsilon} \Bigg(-\frac{1}{32} \epsilon^2 + \frac{1}{8} \epsilon^2 - \frac{1}{8} \epsilon^2 - \frac{1}{12} \epsilon + \frac{3}{8} \epsilon - \frac{\epsilon}{2} \\
&\qquad - \Big[-\frac{1}{2} \epsilon^2 + \epsilon^2 - \frac{1}{2} \epsilon^2 - \frac{2}{3} \epsilon + \frac{3}{2} \epsilon - \epsilon \Big] \Bigg) \\
&= -\frac{6}{\epsilon} \Bigg(-\frac{1}{32} \epsilon^2 + \frac{1}{8} \epsilon^2 - \frac{1}{8} \epsilon^2 - \frac{1}{12} \epsilon + \frac{3}{8} \epsilon - \frac{\epsilon}{2} - \Big[-\frac{4}{6} \epsilon + \frac{9}{6} \epsilon - \frac{6}{9} \epsilon \Big] \Bigg)
\end{aligned}
$$

$$
\begin{aligned}
&= -\frac{6}{\epsilon}\left(-\frac{1}{32}\epsilon^2 - \frac{2}{24}\epsilon + \frac{9}{24}\epsilon - \frac{12}{24}\epsilon + \frac{1}{6}\epsilon \right) \\
&= -\frac{6}{\epsilon}\left(-\frac{1}{32}\epsilon^2 - \frac{1}{24}\epsilon \right) \\
&= \frac{6}{32}\epsilon + \frac{6}{24} \\
&= \frac{3}{16}\epsilon + \frac{3}{12}.
\end{aligned}
$$

The second integral is

$$
\begin{aligned}
&\int_{-\epsilon/2}^{0} \frac{6}{\epsilon}\left(\frac{2}{\epsilon^2}x^2 + \frac{3}{\epsilon}x + 1\right)\left(-x+1\right)dx \\
&= \frac{6}{\epsilon}\int_{-\epsilon/2}^{0} \left(-\frac{2}{\epsilon^2}x^3 + \frac{2}{\epsilon^2}x^2 - \frac{3}{\epsilon}x^2 + \frac{3}{\epsilon}x - x + 1 \right)dx \\
&= \frac{6}{\epsilon}\Bigg|_{-\epsilon/2}^{0} \left(-\frac{2}{\epsilon^2}\cdot\frac{1}{4}x^4 + \frac{2}{\epsilon^2}\cdot\frac{1}{3}x^3 - \frac{3}{\epsilon}\cdot\frac{1}{3}x^3 + \frac{3}{\epsilon}\cdot\frac{1}{2}x^2 - \frac{1}{2}x^2 + x \right) \\
&= \frac{6}{\epsilon}\Bigg(0 - \Big[-\frac{2}{\epsilon^2}\cdot\frac{1}{4}(-\frac{\epsilon}{2})^4 + \frac{2}{\epsilon^2}\cdot\frac{1}{3}(-\frac{\epsilon}{2})^3 \\
&\qquad -\frac{3}{\epsilon}\cdot\frac{1}{3}(-\frac{\epsilon}{2})^3 + \frac{3}{\epsilon}\cdot\frac{1}{2}(-\frac{\epsilon}{2})^2 - \frac{1}{2}(-\frac{\epsilon}{2})^2 + (-\frac{\epsilon}{2}) \Big] \Bigg) \\
&= \frac{6}{\epsilon}\left(-\left[-\frac{1}{32}\epsilon^2 - \frac{1}{12}\epsilon + \frac{1}{8}\epsilon^2 + \frac{3}{8}\epsilon - \frac{1}{8}\epsilon^2 - \frac{\epsilon}{2} \right] \right) \\
&= \frac{6}{\epsilon}\left(-\left[-\frac{1}{32}\epsilon^2 - \frac{2}{24}\epsilon + \frac{9}{24}\epsilon - \frac{12}{24}\epsilon \right] \right) \\
&= 6\left(\frac{1}{32}\epsilon - \frac{5}{24} \right] \Big) \\
&= \frac{3}{16}\epsilon + \frac{15}{12}.
\end{aligned}
$$

Therefore the integral as whole becomes

$$
\begin{aligned}
\gamma_0(\sigma) &= \int_{-\epsilon}^{0} |\sigma''(x)|(|x|+1)dx \\
&= \frac{3}{16}\epsilon + \frac{3}{12} + \frac{3}{16}\epsilon + \frac{15}{12} \\
&= \frac{3}{8}\epsilon + \frac{3}{2}.
\end{aligned}
$$

DOI https://doi.org/10.4208/cicp.OA-2023-0253 | Generated on 2024-12-19 11:09:00

For σ the function g is

$$g(x)=|\sigma(x)|+(|x|+2)|\sigma'(x)|, \tag{B.4}$$

and the infimum of the function g over $\mathbb{R}$ is

$$\inf_{x\in\mathbb{R}} g(x)=0. \tag{B.5}$$

Therefore the value of $\gamma(\sigma)$ is

$$\gamma(\sigma)=\gamma_0(\sigma)+\inf_{x\in\mathbb{R}} g(x)=\frac{3}{8}\epsilon+\frac{3}{2}. \tag{B.6}$$

Thus $\gamma(\sigma)<\infty$. □

Theorem B.2. *The integrand of the Deep Ritz method is Lipschitz, the neural network function is a shallow network (OBS! Give a reference to the shallow network definition!).*

Let u be a shallow network, thus

$$u(x,\theta)=\sum_{i=1}^{m} a_i\sigma(\langle\omega_i,x\rangle).$$

Thus the gradient of u is with respect to spatial variable $x\in\mathbb{R}^2$

$$\nabla u(x,\theta)=\sum_{i=1}^{m} a_i\sigma'(\langle\omega_i,x\rangle)\omega_i.$$

And so it follows that the integrand is

$$\ell(u)=\frac{1}{2}\sum_{j=1}^{2}\Big(\sum_{i=1}^{m} a_i\sigma'(\langle\omega_i,x\rangle)\omega_{ij}\Big)^2-\Big(\sum_{i=1}^{m} a_i\sigma(\langle\omega_i,x\rangle)\Big).$$

To show that ℓ is Lipschitz function, we need to show that for two shallow networks u and v is holds that

$$|\ell(u)-\ell(v)|\leq K|u-v|,$$

for some $K\in\mathbb{R}$.

Let u and v be shallow networks. Let see how $\ell(u)-\ell(v)$ looks like

$$\begin{aligned}
\ell(u)-\ell(v) &= \frac{1}{2}\sum_{j=1}^{2}\Big(\sum_{i=1}^{m}a_i\sigma'(\langle\omega_i,x\rangle)\omega_{ij}\Big)^2-\Big(\sum_{i=1}^{m}a_i\sigma(\langle\omega_i,x\rangle)\Big)\\
&\quad-\left(\frac{1}{2}\sum_{j=1}^{2}\Big(\sum_{i=1}^{m}b_i\sigma'(\langle z_i,x\rangle)z_{ij}\Big)^2-\Big(\sum_{i=1}^{m}b_i\sigma(\langle z_i,x\rangle)\Big)\right)\\
&=\frac{1}{2}\sum_{j=1}^{2}\left(\Big(\sum_{i=1}^{m}a_i\sigma'(\langle\omega_i,x\rangle)\omega_{ij}\Big)^2-\Big(\sum_{i=1}^{m}b_i\sigma'(\langle z_i,x\rangle)z_{ij}\Big)^2\right)\\
&\quad-\sum_{i=1}^{m}\Big(a_i\sigma(\langle\omega_i,x\rangle)-b_i\sigma(\langle z_i,x\rangle)\Big)\\
&=\frac{1}{2}\sum_{j=1}^{2}\left(\sum_{k,i=1}^{m}a_ia_k\sigma'(\langle\omega_i,x\rangle)\sigma'(\langle\omega_k,x\rangle)\omega_{ij}\omega_{kj}-b_ib_k\sigma'(\langle z_i,x\rangle)\sigma'(\langle z_k,x\rangle)z_{ij}z_{kj}\right)\\
&\quad-\sum_{i=1}^{m}\Big(a_i\sigma(\langle\omega_i,x\rangle)-b_i\sigma(\langle z_i,x\rangle)\Big).
\end{aligned}$$

Let's start by focusing on the first term. Let's simplify the notation so that $\sigma'(\langle\omega_i,x\rangle)=\sigma'_{\omega_i}$

$$\begin{aligned}
&\frac{1}{2}\sum_{j=1}^{2}\left(\sum_{k,i=1}^{m}a_ia_k\sigma'_{\omega_i}\sigma'_{\omega_k}\omega_{ij}\omega_{kj}-b_ib_k\sigma'_{z_i}\sigma'_{z_k}z_{ij}z_{kj}\right)\\
&=\frac{1}{2}\sum_{j=1}^{2}\left(\sum_{k,i=1}^{m}a_ia_k\sigma'_{\omega_i}\sigma'_{\omega_k}\omega_{ij}\omega_{kj}-a_ia_k\sigma'_{\omega_i}\sigma'_{\omega_k}z_{ij}z_{kj}+a_ia_k\sigma'_{\omega_i}\sigma'_{\omega_k}z_{ij}z_{kj}-b_ib_k\sigma'_{z_i}\sigma'_{z_k}z_{ij}z_{kj}\right)\\
&=\frac{1}{2}\sum_{j=1}^{2}\left(\sum_{k,i=1}^{m}a_ia_k\sigma'_{\omega_i}\sigma'_{\omega_k}\Big(\omega_{ij}\omega_{kj}-z_{ij}z_{kj}\Big)+\Big(a_ia_k\sigma'_{\omega_i}\sigma'_{\omega_k}-b_ib_k\sigma'_{z_i}\sigma'_{z_k}\Big)z_{ij}z_{kj}\right)\\
&=\frac{1}{2}\sum_{j=1}^{2}\left(\sum_{k,i=1}^{m}a_ia_k\sigma'_{\omega_i}\sigma'_{\omega_k}\Big(\omega_{ij}\big(\omega_{kj}-z_{kj}\big)-\big(\omega_{ij}-z_{ij}\big)z_{kj}\Big)\right.\\
&\qquad\left.+\Big(a_k(a_i-b_i)\sigma'_{\omega_i}\sigma'_{\omega_k}+a_k\big(\sigma'_{\omega_i}\sigma'_{\omega_k}-\sigma'_{z_i}\sigma'_{z_k}\big)+\big(a_k-b_k\big)\sigma'_{z_i}\sigma'_{z_k}\Big)z_{ij}z_{kj}\right)\\
&=\frac{1}{2}\sum_{j=1}^{2}\left(\sum_{k,i=1}^{m}a_ia_k\sigma'_{\omega_i}\sigma'_{\omega_k}\Big(\omega_{ij}\big(\omega_{kj}-z_{kj}\big)-\big(\omega_{ij}-z_{ij}\big)z_{kj}\Big)\right.\\
&\qquad\left.+\Big(a_k(a_i-b_i)\sigma'_{\omega_i}\sigma'_{\omega_k}+a_k\big(\sigma'_{\omega_i}(\sigma'_{\omega_k}-\sigma'_{z_k})+(\sigma'_{\omega_i}-\sigma'_{z_i})\sigma'_{z_k}\big)+\big(a_k-b_k\big)\sigma'_{z_i}\sigma'_{z_k}\Big)z_{ij}z_{kj}\right).
\end{aligned}$$

DOI https://doi.org/10.4208/cicp.OA-2023-0253 | Generated on 2024-12-19 11:09:00

Then we can bound the first summation term

$$\begin{aligned}
&\left|\frac{1}{2}\sum_{j=1}^{2}\left(\sum_{k,i=1}^{m} a_i a_k \sigma'_{\omega_i}\sigma'_{\omega_k}\omega_{ij}\omega_{kj} - b_i b_k \sigma'_{z_i}\sigma'_{z_k} z_{ij} z_{kj}\right)\right| \\
\leq &\frac{1}{2}\sum_{j=1}^{2}\left(\sum_{k,i=1}^{m} |a_i||a_k||\sigma'_{\omega_i}||\sigma'_{\omega_k}|\Big(|\omega_{ij}||\omega_{kj}-z_{kj}| - |\omega_{ij}-z_{ij}||z_{kj}|\Big)\right. \\
&\qquad + \Big(|a_k||a_i-b_i||\sigma'_{\omega_i}||\sigma'_{\omega_k}| + |a_k|\big(|\sigma'_{\omega_i}||\sigma'_{\omega_k}-\sigma'_{z_k}| + |\sigma'_{\omega_i}-\sigma'_{z_i}||\sigma'_{z_k}|\big) \\
&\qquad \left. + |a_k-b_k||\sigma'_{z_i}||\sigma'_{z_k}|\Big)|z_{ij}||z_{kj}|\right).
\end{aligned}$$

As we have chosen functions u and v all the weights must be bounded by some value C_ω. Also by our assumptions on the activation function, we can say that σ' is bounded for all values of weights (ω, z) and spatial variable x, let's denote that by C_σ

$$\begin{aligned}
\leq &\frac{1}{2}\sum_{j=1}^{2}\left(\sum_{k,i=1}^{m} C_\omega C_\omega C_{\sigma'} C_{\sigma'}\Big(C_\omega|\omega_{kj}-z_{kj}| - |\omega_{ij}-z_{ij}|C_\omega\Big)\right. \\
&\left. + \Big(C_\omega|a_i-b_i|C_{\sigma'}C_{\sigma'} + C_\omega\big(C_{\sigma'}|\sigma'_{\omega_k}-\sigma'_{z_k}| + |\sigma'_{\omega_i}-\sigma'_{z_i}|C_{\sigma'}\big) + |a_k-b_k|C_{\sigma'}C_{\sigma'}\Big)C_\omega C_\omega\right).
\end{aligned}$$

The second summation can be handled as follows:

$$\begin{aligned}
&\sum_{i=1}^{m}\Big(a_i\sigma(\langle\omega_i,x\rangle) - b_i\sigma(\langle z_i,x\rangle)\Big) \\
=&\sum_{i=1}^{m}\Big(a_i\sigma(\langle\omega_i,x\rangle) - b_i\sigma(\langle\omega_i,x\rangle) + b_i\sigma(\langle\omega_i,x\rangle) - b_i\sigma(\langle z_i,x\rangle)\Big) \\
=&\sum_{i=1}^{m}\Big((a_i-b_i)\sigma(\langle\omega_i,x\rangle) + b_i\big(\sigma(\langle\omega_i,x\rangle) - \sigma(\langle z_i,x\rangle)\big)\Big).
\end{aligned}$$

Due to Lipschitz property of σ we know

$$\begin{aligned}
&\big|\sigma(\langle\omega_i,x\rangle) + \sigma(\langle z_i,x\rangle)\big| \\
\leq &K|\langle\omega_i,x\rangle - \langle z_i,x\rangle| \\
\leq &K\|\omega_i-z_i\|_1\|x\|_\infty \\
\leq &K\|\omega_i-z_i\|_1.
\end{aligned}$$

DOI https://doi.org/10.4208/cicp.OA-2023-0253 | Generated on 2024-12-19 11:09:00

So we have

$$\begin{aligned}
&\Big|\sum_{i=1}^{m}\Big(a_i\sigma(\langle\omega_i,x\rangle)-b_i\sigma(\langle z_i,x\rangle)\Big)\Big| \\
\leq&\sum_{i=1}^{m}\Big(\big|a_i-b_i\big||\sigma(\langle\omega_i,x\rangle)|+|b_i|\big|\sigma(\langle\omega_i,x\rangle)-\sigma(\langle z_i,x\rangle)\big|\Big) \\
\leq&\sum_{i=1}^{m}\Big(\big|a_i-b_i\big||\sigma(\langle\omega_i,x\rangle)|+|b_i|\big|\sigma(\langle\omega_i,x\rangle)-\sigma(\langle z_i,x\rangle)\big|\Big) \\
\leq&\sum_{i=1}^{m}\Big(\big|a_i-b_i\big||\langle\omega_i,x\rangle|+|b_i|K\big|\langle\omega_i,x\rangle-\langle z_i,x\rangle\big|\Big) \\
\leq&\sum_{i=1}^{m}\Big(\big|a_i-b_i\big||\langle\omega_i,x\rangle|+|b_i|K\big|\langle\omega_i-z_i,x\rangle\big|\Big) \\
\leq&\sum_{i=1}^{m}\Big(\big|a_i-b_i\big|\|\omega_i\|_1\|x\|_\infty+|b_i|K\|\omega_i-z_i\|_1\|x\|_\infty\Big) \\
\leq&\sum_{i=1}^{m}\Big(\big|a_i-b_i\big|\|\omega_i\|_1+|b_i|K\|\omega_i-z_i\|_1\Big).
\end{aligned}$$

So both upperbounds together we have,

$$\begin{aligned}
&|\ell(u)-\ell(u)| \\
\leq&\frac{1}{2}\sum_{j=1}^{2}\Bigg(\sum_{k,i=1}^{m}|a_i||a_k||\sigma'_{\omega_i}||\sigma'_{\omega_k}|\Big(|\omega_{ij}|\big|\omega_{kj}-z_{kj}\big|-\big|\omega_{ij}-z_{ij}\big||z_{kj}|\Big) \\
&+\Big(|a_k|\big|a_i-b_i\big||\sigma'_{\omega_i}||\sigma'_{\omega_k}|+|a_k|\big(|\sigma'_{\omega_i}|\big|\sigma'_{\omega_k}-\sigma'_{z_k}\big|+\big|\sigma'_{\omega_i}-\sigma'_{z_i}\big||\sigma'_{z_k}|\big) \\
&+\big|a_k-b_k\big||\sigma'_{z_i}||\sigma'_{z_k}|\Big)|z_{ij}||z_{kj}|\Bigg)+\sum_{i=1}^{m}\Big(\big|a_i-b_i\big|\|\omega_i\|_1+|b_i|K\|\omega_i-z_i\|_1\Big).
\end{aligned}$$

Then as we apply the bounds on the weights and the on the derivatives of the activation function we get

$$\begin{aligned}
&|\ell(u)-\ell(u)| \\
\leq&\frac{1}{2}\sum_{j=1}^{2}\Bigg(\sum_{k,i=1}^{m}C_\omega C_\omega C_{\sigma'}C_{\sigma'}\Big(C_\omega\big|\omega_{kj}-z_{kj}\big|+\big|\omega_{ij}-z_{ij}\big|C_\omega\Big) \\
&+\Big(C_\omega\big|a_i-b_i\big|C_{\sigma'}C_{\sigma'}+C_\omega\big(C_{\sigma'}\big|\sigma'_{\omega_k}-\sigma'_{z_k}\big|+\big|\sigma'_{\omega_i}-\sigma'_{z_i}\big|C_{\sigma'}\big) \\
&+\big|a_k-b_k\big|C_{\sigma'}C_{\sigma'}\Big)C_\omega C_\omega\Bigg)+\sum_{j=1}^{2}\sum_{i=1}^{m}\Big(\big|a_i-b_i\big||\omega_{ij}|+|b_i|K|\omega_{ij}-z_{ij}|\Big)
\end{aligned}$$

$$
\begin{aligned}
&\leq \frac{1}{2}\sum_{j=1}^{2}\Bigg(\sum_{k,i=1}^{m} C_\omega C_\omega C_{\sigma'} C_{\sigma'}\Big(C_\omega|\omega_{kj}-z_{kj}|+|\omega_{ij}-z_{ij}|C_\omega\Big)\\
&\qquad+\Big(C_\omega|a_i-b_i|C_{\sigma'}C_{\sigma'}+C_\omega\big(C_{\sigma'}|\sigma'_{\omega_k}-\sigma'_{z_k}|+|\sigma'_{\omega_i}-\sigma'_{z_i}|C_{\sigma'}\big)\\
&\qquad+|a_k-b_k|C_{\sigma'}C_{\sigma'}\Big)C_\omega C_\omega\Bigg)+\sum_{j=1}^{2}\sum_{i=1}^{m}\Big(|a_i-b_i|C_\omega+C_\omega K|\omega_{ij}-z_{ij}|\Big)\\
&\leq \frac{1}{2}\sum_{j=1}^{2}\Bigg[\sum_{k,i}^{m} C_\omega^3 C_{\sigma'}^2\Big(|\omega_{kj}-z_{kj}|+|\omega_{ij}-z_{ij}|+|a_k-b_k|\Big)\\
&\qquad+C_\omega^3 C_{\sigma'}K_2\Big(|\omega_{kj}-z_{kj}|+|\omega_{ij}-z_{ij}|\Big)+C_\omega^2 C_{\sigma'}^2|a_k-b_k|\Bigg]\\
&\qquad+\sum_{j=1}^{2}\sum_{i=1}^{m}\Big(C_\omega|a_i-b_i|+C_\omega K|\omega_{ij}-z_{ij}|\Big).
\end{aligned}
$$

C Lipschitz continuity of the loss function

In this section our main goal is to prove that the loss function is Lipshitz continuous. We will not show what the Lipschitz constant is exactly, but we prove that such a constant exists.

Theorem C.1. *Continuously differentiable functions $\phi: B(0,1)\to\mathbb{R}^d$, with bounded gradient, are Lipschitz.*

Proof. Let $B(0,1)\subset\mathbb{R}^d$ be a unit sphere in $\mathbb{R}^d$. Let $\phi: B(0,1)\to\mathbb{R}$ be a continuously differentiable mapping with bounded $\|\nabla\phi\|<M$, where $M<\infty$. Let $g(t)=\phi(x_1+t(x_2-x_1))$

$$
\begin{aligned}
|\phi(x_1)-\phi(x_2)| = |g(0)-g(1)| &= \Big|\int_0^1 g'(t)dt\Big|\\
&= \Big|\int_0^1 \nabla\phi(x_1+t(x_2-x_1))(x_2-x_1)dt\Big|\\
&= \int_0^1 |\nabla\phi(x_1+t(x_2-x_1))||x_2-x_1|dt\\
&\leq M|x_2-x_1|.
\end{aligned}
$$

Therefore ϕ is Lipschitz. □

Theorem C.2. *Let u be a shallow neural network with bounded path norm and bounded weights. Then the function $\ell(u)=\ell(x,\theta)=\frac{|\nabla u|}{2}-u$ is a Lipschitz mapping.*

Proof. See the end of the section. □

Lemma C.1. *The gradient of the loss function ℓ with respect to the spatial variables ($x \in B(0,1)$) is*

$$\nabla \ell(u) = \nabla u H(u) - \nabla u, \tag{C.1}$$

where $H(u)$ is the Hessian matrix of the function u.

Proof. Let's differentiate the loss function $\ell(u) = \frac{\nabla u}{2} - u$. In the following all gradients are taken with respect to the spatial variable x.

$$\begin{aligned}
\nabla\Big(\frac{|\nabla u|^2}{2}\Big) &= \Big(\partial_1\big(\frac{(\partial_1 u)+(\partial_2 u)}{2}\big), \partial_2\big(\frac{(\partial_1 u)+(\partial_2 u)}{2}\big)\Big) \\
&= \Big(\partial_1 u(\partial_1^2 u) + \partial_2 u(\partial_2\partial_1 u), \partial_1 u(\partial_2\partial_1 u) + \partial_2 u(\partial_2^2 u)\Big) \\
&= \Big(\big(\partial_1 u, \partial_2 u\big)\begin{bmatrix}\partial_1^2 u \\ \partial_2\partial_1 u\end{bmatrix}, \big(\partial_1 u, \partial_2 u\big)\begin{bmatrix}\partial_2\partial_1 u \\ \partial_2^2 u\end{bmatrix}\Big) \\
&= \big(\partial_1 u, \partial_2 u\big)\begin{bmatrix}\partial_1^2 u & \partial_2\partial_1 u \\ \partial_2\partial_1 u & \partial_2^2 u\end{bmatrix} \\
&= \nabla u H(u).
\end{aligned}$$

And so the statement follows from this. □

Lemma C.2. *Derivative of shallow network u is*

$$\nabla u = \sum_i a_i \sigma'(\langle \omega_i, x\rangle)\omega_{ij}.$$

Proof. Simple application of basic rules of differential calculus. □

Lemma C.3. *If $u \in \mathcal{F}_Q$ is a shallow network with bounded path norm, then its gradient is bounded.*

Proof.

$$\begin{aligned}
|\nabla u|_1 &= \sum_{j=1}^{2}\Big|\sum_{i=1}^{m} a_i \sigma'(\langle \omega_i, x\rangle)\omega_{ij}\Big| \\
&\leq \sum_{j=1}^{2}\sum_{i=1}^{m} |a_i|\big|\sigma'(\langle \omega_i, x\rangle)\big||\omega_{ij}| \\
&\leq C_{\sigma'}\sum_{j=1}^{2}\sum_{i=1}^{m} |a_i||\omega_{ij}| \\
&\leq C_{\sigma'} Q.
\end{aligned}$$

This completes the proof. □

DOI https://doi.org/10.4208/cicp.OA-2023-0253 | Generated on 2024-12-19 11:09:00

Lemma C.4. *Let $u \in \mathcal{F}_Q$ be a shallow network with a bounded path norm. Then its second order derivative is bounded, i.e. $|\partial_j \partial_k u| < \infty$, where $j,k = 1,2$.*

Proof. Second order derivative of u is

$$\begin{aligned}\partial_k \partial_j u &= \partial_k \sum_{i=1}^{m} a_i \sigma'(\langle \omega_i, x \rangle) \omega_{ij} \\ &= \sum_{i=1}^{m} a_i \sigma''(\langle \omega_i, x \rangle) \omega_{ij} \omega_{ik}.\end{aligned}$$

Then its bound is

$$|\partial_k \partial_j u| = C_{\sigma''} \sum_{i=1}^{m} |a_i||\omega_{ij}||\omega_{ik}|.$$

If $j = k$ then the bound becomes

$$|\partial_j^2 u| = C_{\sigma''} \sum_{i=1}^{m} |a_i||\omega_{ij}|^2 \leq C_{\sigma''} \sum_{i=1}^{m} |a_i| \|\omega_i\|_2^2 \leq C_{\sigma''} \sum_{i=1}^{m} |a_i| \|\omega_i\|_1^2 < \infty.$$

As we have assumed that $\|\theta\|_P < Q$. Therefore we have

$$\sum |a_i||\omega_{i1}| + \sum |a_i||\omega_{i2}| < Q.$$

Therefore if $a_i > 0$ then we may deduce that $|\omega_{i1}|$, $|\omega_{i2}| < \infty$. And then

$$\sum_{i=1}^{m} |a_i||\omega_{i1}||\omega_{i2}| < \infty.$$

This completes the proof. □

Corollary C.1. *Let u be a shallow network with bounded path norm. The norm of Hessian $H(u)$ is bounded.*

Corollary C.2. *Let u be a shallow network with bounded path norm, then $\nabla \ell(u)$ is bounded.*

Proof. By Theorem C.1, Corollary C.1, Lemma C.3. □

Now we can prove that ℓ is Lipschitz in the case of shallow networks with bounded path norm.

Proof. (for Theorem C.2) Proof follows by Theorem C.1 and Corollary C.2. □

We do not specify the Lipschitz constant. Next we prove the same thing for residual neural networks.

DOI https://doi.org/10.4208/cicp.OA-2023-0253 | Generated on 2024-12-19 11:09:00

Lemma C.5. *Derivative of residual neural network u is*

$$\nabla u = a^T V + a^T U_1 \big(\operatorname{diag}\sigma'\big)(W_1 V x) W_1 V + a^T \sum_{k=2}^{L} U_k \sum_{i=k}^{2} \operatorname{diag}\sigma'(W_k h_{k-1}) W_i U_{i-1} \nabla g_{i-1},$$

where

$$\nabla g_k = \sum_{i=k}^{2} \operatorname{diag}\sigma'(W_k h_{k-1}) W_i U_{i-1} \nabla g_{i-1},$$

and

$$\nabla g_1 = \operatorname{diag}\sigma'(W_1 V x) W_1 V.$$

Proof.

$$\begin{aligned} u(x,\theta) &= a^T h_L \\ &= a^T h_{L-1} + a^T U_L g_L \\ &= a^T h_0 + a^T U_1 g_1 + \cdots + a^T U_L g_L. \end{aligned}$$

Derivative of the first term is

$$\nabla a^T h_0 = a^T V.$$

Derivative of the second term is

$$\nabla a^T U_1 g_1 = a^T U_1 \big(\operatorname{diag}\sigma'\big)(W_1 V x) W_1 V.$$

Derivative of the rest of the layers g are

$$\nabla g_k = \sum_{i=k}^{2} \operatorname{diag}\sigma'(W_k h_{k-1}) W_i U_{i-1} \nabla g_{i-1},$$

where $k = 2, \cdots, L$. □

The principal term of ∇u is

$$\begin{aligned} \nabla u \Big| P = a^T U_L \big(\operatorname{diag}\sigma'\big)(W_L h_{L-1}) W_L U_{L-1} \big(\operatorname{diag}\sigma'\big)(W_{L-1} h_{L-2}) \\ \times \cdots \times W_2 U_1 \big(\operatorname{diag}\sigma'\big)(W_1 h_0) W_1 V. \end{aligned} \tag{C.2}$$

One gets the rest of the (summation) terms in ∇u by dropping out some of the multiplication factors in the principal term.

Lemma C.6. *Let u be a residual neural network with bounded path norm. The principal term of the gradient ∇u is bounded.*

Proof.

$$\begin{aligned}|\nabla u\Big|P| &\leq |a^T||U_L|\big|\mathrm{diag}\sigma'(W_L h_{L-1})\big||W_L||U_{L-1}|\big|\mathrm{diag}\sigma'(W_{L-1}h_{L-2})\big| \\ &\quad \times\cdots\times|W_2||U_1|\big|\mathrm{diag}\sigma'(W_1h_0)\big||W_1||V|. \\ &\leq C_{\sigma'}^L a^T||U_L||W_L||U_{L-1}|\times\cdots\times|W_2||U_1||W_1||V| \\ &\leq C_{\sigma'}^L\|\theta\|_P.\end{aligned}$$

This completes the proof. □

Lemma C.7. *Let u be a residual neural network with bounded path norm. The gradient ∇u is bounded.*

Proof. The principal term of the gradient is given in (C.2). The gradient is a sum of terms which are alike the principal term, except that other terms lack some of the factorial tensors. There is a finite number of these terms, let's call this number $\#(P)$, the number of paths within the gradient network. Then we have

$$|\nabla u|\leq\#(P)C_{\sigma'}^L\|\theta\|_P.$$

This completes the proof. □

Second order gradient of the residual neural network is

$$\nabla^2 u = a^T U_1\big(\mathrm{diag}\sigma''\big)(W_1Vx)W_1VW_1V + a^T\sum_{k=2}^{L}U_k\nabla^2 g_k,$$

$$\begin{aligned}\nabla^2 g_k = \sum_{i=k}^{2}\Big[&\big(\mathrm{diag}\sigma''\big)(W_kh_{k-1})\times W_iU_{i-1}\nabla g_{i-1}W_iU_{i-1}\nabla g_{i-1} \\ &+\big(\mathrm{diag}\sigma'\big)(W_kh_{k-1})W_iU_{i-1}\nabla^2 g_{i-1}\Big].\end{aligned}$$

The main thing is that second order derivative of a residual neural network consists of second order, and first order derivatives of the activation function and multiplications of the weight tensors. As we deriviate with respect to the spatial variable x, all the weights of the neural network stay in the multiplication, i.e. all the paths stay included; there will be just some not but two times in the path.

The principal term of the second order gradient is

$$\begin{aligned}\nabla^2 u\Big|_P = &a^T U_L\big(\mathrm{diag}\sigma''\big)(W_Lh_{L-1})\Big[W_LU_{L-1}\big(\mathrm{diag}\sigma'\big)(W_{L-1}h_{L-2})\cdots \\ &W_2U_1\big(\mathrm{diag}\sigma'\big)(W_1h_0)W_1V\Big]\Big[W_LU_{L-1}\big(\mathrm{diag}\sigma'\big)(W_{L-1}h_{L-2})\cdots\end{aligned}$$

$$W_2U_1\big(\mathrm{diag}\sigma'\big)(W_1h_0)W_1V\Big].$$

The bound for the principal term is

$$\begin{aligned}|\nabla^2u\Big|_P| \leq & C_{\sigma''}C_{\sigma'}^{2L-2}|a^T||U_L|\Big[|W_L||U_{L-1}|\cdots|W_2||U_1||W_1||V|\Big]\\ &\Big[|W_L||U_{L-1}|\cdots|W_2||U_1||W_1||V|\Big]\\ &\leq C_{\sigma''}C_{\sigma'}^{2L-2}\|\theta\|_P^2.\end{aligned}$$

The bound on the principal term bounds also the other terms in the full second order derivative. Therefore we have a bound for it.

Lemma C.8. *Let u be a residual neural network with bounded path norm. Then its second order derivative is bounded.*

Proof. As all the bound for the principal term $\nabla^2u\Big|_P$ is a bound also for the other terms in the full gradient, and there is a finite number of terms making the full second order derivative, then we can say that $\nabla^2u\Big|_P$ is bounded. □

Theorem C.3. *Let u be a residual neural network with bounded path norm and bounded weights. Then the function $\ell(u)=\ell(x,\theta)=\frac{|\nabla u|}{2}-u$ is a Lipschitz mapping.*

Proof. As ∇u and $H(u)$ are bounded, this implies that $\nabla\ell$ is bounded. Then by Theorem C.1 we know that ℓ is a Lipschitz mapping. □

D Barron space

In this section we prove that the solution u can be extended to $\mathbb{R}^2$ as a Schwartz function. Fourier transform of a Schwartz function is a Schwartz function, i.e. fourier transform is a map from $\mathscr{S}(\mathbb{R}^d)\to\mathscr{S}(\mathbb{R}^d)$ (Theorem 1.3 in [50]). This will ensure that the solution is in the space of Barron functions, as Schwartz functions are rapidly decreasing functions.

The solution to the Poisson equation is

$$u(x)=\frac{1}{4}(1-|x|^2),\quad x\in B(0,1). \tag{D.1}$$

We can extend the function u to the whole plane $\mathbb{R}^2$ as it is, for clarity let's denote this extension with another symbol

$$\tilde{u}(x)=\frac{1}{4}(1-|x|^2),\quad x\in\mathbb{R}^2. \tag{D.2}$$

Let's define

$$f(x)=\min\{(4-|x|^2),1\}. \tag{D.3}$$

DOI https://doi.org/10.4208/cicp.OA-2023-0253 | Generated on 2024-12-19 11:09:00

Idea of the function f is that, when we consider $x\in B(0,1)$ and we multiply the functions $\tilde{u}$ and f pointwise, the result is $u(x)$; this is simply because f is constant 1 inside the unit ball $B(0,1)$. Another nice property is that the multiplication decays rapidly to zero when $|x|\to\infty$. This way we get an extension of u to the whole plane such that it decays rapidly to zero.

We need still one more thing, and that is that we need to smooth the extension in order to make it a Schwartz function. Let η_ϵ be a standard mollifier [51]. Let's define our extension to be

$$F(x)=\big(f*\eta_\epsilon\big)(x)=\int_{B(0,\epsilon)}\eta_\epsilon(y)f(x-y)dy. \tag{D.4}$$

Properties of the mollifier ensure that the function F is C^∞. Now the product $\tilde{u}(x)F(x)$ is the extension we are looking for. Properties of the function f ensure that the function F decays to zero faster than any polynomial, and that it is constant 1 inside $B(0,1)$. Therefore $\tilde{u}(x)F(x)$ is an extension of the solution u to the whole plane, and $\tilde{u}(x)F(x)$ is a Schwartz function.

Theorem D.1. *Let $u,\tilde{u},f,F$ be defined as in $(D.1)$, $(D.2)$, $(D.3)$, and $(D.4)$. Then $\tilde{u}(x)F(x)$ is an extension of the solution u to the whole plane $\mathbb{R}^2$. In addition, $\tilde{u}(x)F(x)$ is C^∞ and decays to zero faster than any polynomial, i.e. is a Schwartz function.*

In article [52] the authors proved the following theorem (p.29, theorem 6); see also [53] where the theorem is presented in proposition 2.

Theorem D.2. *Let $f\in C(X)$, the space of continuous functions on X, and assume that f satisfies:*

$$\gamma(f):=\inf_{\widehat{f}}\int_{\mathbb{R}^d}\|\omega\|_1^2|\widehat{f}(\omega)|d\omega<\infty,$$

where $\widehat{f}$ is the Fourier transform of an extension of f to $\mathbb{R}^d$. Then f belongs to the Barron space, and its Barron norm is bounded by

$$\|f\|_{\mathcal{B}}\leq 2\gamma(f)+2\|\nabla f(0)\|_1+2|f(0)|.$$

Theorem D.3. *Solution u belongs to the Barron space.*

Proof. By Theorems D.1 and D.2 we can say that the solution u belongs to the Barron space. The extension $\tilde{u}(x)F(x)$ is probably not giving the spectral norm $\gamma(u)$ for the solution, but if we have one extension with bounded integral of the same form as in spectral norm, then we know that the minimal extension also gives a bounded spectral norm. □

In article [12] authors mention (see also [54]) a result for Barron functions and shallow neural networks: for any $f\in\mathcal{B}_2$ there exists a shallow neural network $f(\cdot,\theta)$ of width m

such that

$$\mathbb{E}_x[(f(x)-f(x,\theta))^2] \leq \frac{3\gamma_2^2(f)}{m},$$
$$\|\theta\|_{\mathcal{P}} \leq 2\gamma_2(f).$$

And in article [29] (theorem 2.5) the authors showed that residual networks can approximate the Barron functions in a following way: Let f^* be the Barron function. There exists a residual network $f(\cdot,\theta)$ with depth L and width m such that

$$\|f(x,\theta)-f^*\| \leq \frac{3\|f^*\|_{\mathcal{B}}^2}{Lm}$$

and

$$\|\theta\|_P \leq 4\|f^*\|_{\mathcal{B}}.$$

With this information and with the fact that the solution to the Poisson's equation belongs to the Barron space, we know something about the approximation error in the situation. At least we can say that by increasing the width and depth of the residual neural network, we can push the approximation error to zero without blowing the pathnorm of the network. And in this sense residual networks and shallow networks could be a natural choice for the network architecture.

References

[1] W. E, B. Yu, The deep Ritz method: A deep learning-based numerical algorithm for solving variational problems, Communications in Mathematics and Statistics 6 (Mar. 2018). doi:10.1007/s40304-018-0127-z.

[2] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016, http://www.deeplearningbook.org.

[3] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (2015) 436–444. doi:10.1038/nature14539.

[4] B. Maury, Numerical analysis of a finite element/volume penalty method, SIAM Journal on Numerical Analysis 47 (2) (2009) 1126–1148. doi:10.1137/080712799.
URL https://doi.org/10.1137/080712799

[5] J. Müller, M. Zeinhofer, Error estimates for the deep Ritz method with boundary penalty, in: B. Dong, Q. Li, L. Wang, Z.-Q. J. Xu (Eds.), Proceedings of Mathematical and Scientific Machine Learning, Vol. 190 of Proceedings of Machine Learning Research, PMLR, 2022, pp. 215–230.
URL https://proceedings.mlr.press/v190/muller22a.html

[6] Y. Lu, J. Lu, M. Wang, A priori generalization analysis of the deep Ritz method for solving high dimensional elliptic partial differential equations, in: M. Belkin, S. Kpotufe (Eds.), Proceedings of Thirty Fourth Conference on Learning Theory, Vol. 134 of Proceedings of Machine Learning Research, PMLR, 2021, pp. 3196–3241.
URL https://proceedings.mlr.press/v134/lu21a.html

[7] F. He, T. Liu, D. Tao, Why resnet works? residuals generalize, IEEE Transactions on Neural Networks and Learning Systems 31 (12) (2020) 5349–5362. doi:10.1109/TNNLS.2020.2966319.
[8] M. Costabel, M. Dauge, A singularly perturbed mixed boundary value problem, Communications in Partial Differential Equations 21 (11-12) (12 1996).
URL https://www.osti.gov/biblio/530804
[9] T. Hu, B. Jin, Z. Zhou, Solving elliptic problems with singular sources using singularity splitting deep Ritz method, SIAM Journal on Scientific Computing 45 (4) (2023) A2043–A2074. arXiv:https://doi.org/10.1137/22M1520840, doi:10.1137/22M1520840.
URL https://doi.org/10.1137/22M1520840
[10] C. Duan, Y. Jiao, Y. Lai, X. Lu, Q. Quan, J. Z. Yang, Analysis of deep Ritz methods for Laplace equations with Dirichlet boundary conditions (2021). arXiv:2111.02009.
[11] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Networks 2 (5) (1989) 359–366. doi:https://doi.org/10.1016/0893-6080(89)90020-8.
URL https://www.sciencedirect.com/science/article/pii/0893608089900208
[12] W. E, C. Ma, L. Wu, A priori estimates of the population risk for two-layer neural networks, Communications in Mathematical Sciences 17 (2019) 1407–1425. doi:10.4310/CMS.2019.v17.n5.a11.
[13] Y. Jiao, Y. Lai, Y. Lo, Y. Wang, Y. Yang, Error analysis of deep Ritz methods for elliptic equations, Analysis and Applications 22 (01) (2024) 57–87. doi:10.1142/S021953052350015X.
URL https://doi.org/10.1142/S021953052350015X
[14] C. Duan, Y. Jiao, Y. Lai, D. Li, X. Lu, J. Z. Yang, Convergence rate analysis for deep Ritz method, Communications in Computational Physics 31 (4) (2022) 1020–1048. doi:10.4208/cicp.oa-2021-0195.
URL https://doi.org/10.4208%2Fcicp.oa-2021-0195
[15] F. Fu, X. Wang, Convergence analysis of a quasi-Monte Carlo-based deep learning algorithm for solving partial differential equations, Numerical Mathematics: Theory, Methods and Applications 16 (3) (2023) 668–700. doi:https://doi.org/10.4208/nmtma.OA-2022-0166.
URL http://global-sci.org/intro/article_detail/nmtma/21962.html
[16] T. Luo, H. Yang, Two-layer neural networks for partial differential equations: Optimization and generalization theory (2020). arXiv:2006.15733.
[17] M. Raissi, P. Perdikaris, G. 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 (2019) 686–707. doi:https://doi.org/10.1016/j.jcp.2018.10.045.
URL https://www.sciencedirect.com/science/article/pii/S0021999118307125
[18] P. Minakowski, T. Richter, A priori and a posteriori error estimates for the deep Ritz method applied to the Laplace and Stokes problem, Journal of Computational and Applied Mathematics 421 (2023) 114845. doi:10.1016/j.cam.2022.114845.
URL https://doi.org/10.1016%2Fj.cam.2022.114845
[19] Y. Shin, J. Darbon, G. E. Karniadakis, On the convergence of physics informed neural networks for linear second-order elliptic and parabolic type PDEs, Communications in Computational Physics 28 (5) (2020) 2042–2074. doi:10.4208/cicp.oa-2020-0193.
URL https://doi.org/10.4208%2Fcicp.oa-2020-0193
[20] Y. Shin, Z. Zhang, G. E. Karniadakis, Error estimates of residual minimization using neural networks for linear PDEs, Journal of Machine Learning for Modeling and Computing 4 (4)

(2023) 73–101.
[21] S. Cuomo, V. S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi, F. Piccialli, Scientific machine learning through physics–informed neural networks: Where we are and what's next, Journal of Scientific Computing 92 (3) (Sep 2022). doi:10.1007/s10915-022-01939-z.
URL https://doi.org/10.1007/s10915-022-01939-z
[22] K. O. Lye, S. Mishra, R. Molinaro, A multi-level procedure for enhancing accuracy of machine learning algorithms, European Journal of Applied Mathematics 32 (3) (2021) 436–469.
[23] M. Wang, C. Ma, Generalization error bounds for deep neural networks trained by SGD (2022). doi:10.48550/ARXIV.2206.03299.
URL https://arxiv.org/abs/2206.03299
[24] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.
[25] K. He, X. Zhang, S. Ren, J. Sun, Identity mappings in deep residual networks, in: B. Leibe, J. Matas, N. Sebe, M. Welling (Eds.), Computer Vision – ECCV 2016, Springer International Publishing, Cham, 2016, pp. 630–645.
[26] V. N. Vapnik, Wiley, New York (NY), 1998.
[27] M. Anthony, P. L. Bartlett, Neural Network Learning: Theoretical Foundations, Cambridge University Press, Cambridge, 1999.
[28] Z. Li, C. Ma, L. Wu, Complexity measures for neural networks with general activation functions using path-based norms (2020). doi:10.48550/ARXIV.2009.06132.
URL https://arxiv.org/abs/2009.06132
[29] W. E, C. Ma, Q. Wang, Rademacher complexity and the generalization error of residual networks, Communications in Mathematical Sciences 18 (2020) 1755–1774. doi:10.4310/CMS.2020.v18.n6.a10.
[30] S. Shalev-Shwartz, S. Ben-David, Understanding Machine Learning: From Theory to Algorithms, Cambridge University Press, Cambridge, 2014.
[31] B. Neyshabur, R. Tomioka, N. Srebro, Norm-based capacity control in neural networks, in: P. Grünwald, E. Hazan, S. Kale (Eds.), Proceedings of The 28th Conference on Learning Theory, Vol. 40 of Proceedings of Machine Learning Research, Paris, France, 2015.
[32] P. L. Bartlett, D. J. Foster, M. Telgarsky, Spectrally-normalized margin bounds for neural networks, NIPS'17, 2017.
[33] T. Liang, T. Poggio, A. Rakhlin, J. Stokes, Fisher-Rao metric, geometry, and complexity of neural networks, in: K. Chaudhuri, M. Sugiyama (Eds.), Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics, Vol. 89 of Proceedings of Machine Learning Research, PMLR, 2019, pp. 888–896.
URL https://proceedings.mlr.press/v89/liang19a.html
[34] A. Krogh, J. A. Hertz, A simple weight decay can improve generalization, in: Proceedings of the 4th International Conference on Neural Information Processing Systems, NIPS'91, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1991, p. 950–957.
[35] A. Kaltenbach, M. Zeinhofer, The deep Ritz method for parametric p-Dirichlet problems (2022). doi:10.48550/ARXIV.2207.01894.
URL https://arxiv.org/abs/2207.01894
[36] P. Dondl, J. Müller, M. Zeinhofer, Uniform convergence guarantees for the deep Ritz method for nonlinear problems, Advances in Continuous and Discrete Models (07 2022).
[37] L. Lu, X. Meng, Z. Mao, G. E. Karniadakis, DeepXDE: A deep learning library for solving differential equations, SIAM Review 63 (1) (2021) 208–228. doi:10.1137/19M1274067.

URL https://epubs.siam.org/doi/10.1137/19M1274067

[38] W. E, J. Han, 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) (2017) 349–380. doi:10.1007/s40304-017-0117-6.
URL http://dx.doi.org/10.1007/s40304-017-0117-6

[39] J. Sirignano, K. Spiliopoulos, DGM: A deep learning algorithm for solving partial differential equations, Journal of Computational Physics 375 (2018) 1339–1364. doi:10.1016/j.jcp.2018.08.029.
URL http://dx.doi.org/10.1016/j.jcp.2018.08.029

[40] J. Yang, Q. Zhu, A local deep learning method for solving high order partial differential equations, Numerical Mathematics: Theory, Methods and Applications 15 (1) (2022) 42–67. doi:https://doi.org/10.4208/nmtma.OA-2021-0035.
URL http://global-sci.org/intro/article_detail/nmtma/20220.html

[41] Y. Lu, H. Chen, J. Lu, L. Ying, J. H. Blanchet, Machine learning for elliptic PDEs: Fast rate generalization bound, neural scaling law and minimax optimality, in: The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022, OpenReview.net, 2022.
URL https://openreview.net/forum?id=mhYUBYNoGz

[42] N. S. Bakhvalov, On the approximate calculation of multiple integrals, Journal of Complexity 31 (4) (2015) 502–516. doi:https://doi.org/10.1016/j.jco.2014.12.003.
URL https://www.sciencedirect.com/science/article/pii/S0885064X14001204

[43] C. Zhang, S. Bengio, M. Hardt, B. Recht, O. Vinyals, Understanding deep learning (still) requires rethinking generalization, Communications of the ACM 64 (3) (2021) 107–115. doi:10.1145/3446776.
URL https://doi.org/10.1145/3446776

[44] B. Neyshabur, R. Tomioka, N. Srebro, In search of the real inductive bias: On the role of implicit regularization in deep learning, in: Y. Bengio, Y. LeCun (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings, 2015.
URL http://arxiv.org/abs/1412.6614

[45] Q. Li, C. Tai, W. E, Stochastic modified equations and adaptive stochastic gradient algorithms, in: D. Precup, Y. W. Teh (Eds.), Proceedings of the 34th International Conference on Machine Learning, Vol. 70 of Proceedings of Machine Learning Research, PMLR, 2017, pp. 2101–2110.
URL https://proceedings.mlr.press/v70/li17f.html

[46] D. Hendrycks, K. Gimpel, Gaussian error linear units (gelus), arXiv (2016).
URL https://arxiv.org/abs/1606.08415

[47] P. Ramachandran, B. Zoph, Q. V. Le, Searching for activation functions (2017). doi:10.48550/ARXIV.1710.05941.
URL https://arxiv.org/abs/1710.05941

[48] K. Fukushima, Cognitron: A self-organizing multilayered neural network, Biological Cybernetics 20 (3) (1975) 121–136. doi:10.1007/BF00342633.
URL https://doi.org/10.1007/BF00342633

[49] V. Nair, G. E. Hinton, Rectified linear units improve restricted boltzmann machines, in: Proceedings of the 27th International Conference on International Conference on Machine Learning, ICML'10, Omnipress, Madison, WI, USA, 2010, p. 807–814.

[50] E. M. Stein, R. Shakarchi, Fourier Analysis: An Introduction, Princeton University Press, 2003.
[51] L. C. Evans, Partial Differential Equations, 2nd Edition, American Mathematical Society, 2010.
[52] J. M. Klusowski, A. R. Barron, Risk bounds for high-dimensional ridge function combinations including neural networks (2018). arXiv:1607.01434.
[53] W. E, C. Ma, L. Wu, The barron space and the flow-induced function spaces for neural network models, Constructive Approximation 55 (Feb 2022). doi:10.1007/s00365-021-09549-y. URL https://doi.org/10.1007/s00365-021-09549-y
[54] A. Barron, Universal approximation bounds for superpositions of a sigmoidal function, IEEE Transactions on Information Theory 39 (3) (1993) 930–945. doi:10.1109/18.256500.