Two-Layer Neural Networks for Partial Differential Equations: Optimization and Generalization Theory

Tao Luo[1] and Haizhao Yang[2]

[1]School of Mathematical Sciences, Institute of Natural Sciences, MOE-LSC,
and Qing Yuan Research Institute,
Shanghai Jiao Tong University, Shanghai, 200240, P.R. China
[2]Department of Mathematics, Purdue University, West Lafayette, IN 47907, USA

August 18, 2021

Abstract

The problem of solving partial differential equations (PDEs) can be formulated into a least-squares minimization problem, where neural networks are used to parametrize PDE solutions. A global minimizer corresponds to a neural network that solves the given PDE. In this paper, we show that the gradient descent method can identify a global minimizer of the least-squares optimization for solving second-order linear PDEs with two-layer neural networks under the assumption of over-parametrization. We also analyze the generalization error of the least-squares optimization for second-order linear PDEs and two-layer neural networks, when the right-hand-side function of the PDE is in a Barron-type space and the least-squares optimization is regularized with a Barron-type norm, without the over-parametrization assumption.

Keywords. Deep learning, over-parametrization, partial differential equations, optimization convergence, generalization error.

AMS subject classifications: 68U99, 65N30 and 65N25.

1 Introduction

Deep learning, originated in computer science, has revolutionized many fields of science and engineering recently. This revolution also includes broad applications of deep learning in computational and applied mathematics, e.g., many breakthroughs in solving partial differential equations (PDEs) [8, 28, 40, 5, 20, 12, 2, 27, 39, 47, 24, 19]. The key idea of these approaches is to reformulate the PDE solution into a global minimizer of an expectation minimization problem, where deep neural networks (DNNs) are applied for discretization and the stochastic gradient descent (SGD) is adopted to solve the minimization problem. These methods probably date back to the 1990s (e.g., see [8, 28]) and were revisited recently [40, 20, 12, 2, 27, 47, 39] due to the significant development of GPU computing that accelerates DNN computation. Though these approaches have remarkable empirical successes, their theoretical justification remains vastly open.

For simplicity, let us use a PDE defined on a domain Ω in a compact form with equality constrains to illustrate the main idea, e.g.,

$$\begin{cases} \mathcal{L}u = f & \text{in } \Omega, \\ \mathcal{B}u = g & \text{on } \partial\Omega, \end{cases} \tag{1.1}$$

where $\mathcal{L}$ is a differential operator and $\mathcal{B}$ is the operator for specifying an appropriate boundary condition. In the least squares-type methods, DNNs, denoted as $\phi(\boldsymbol{x};\boldsymbol{\theta})$ with a parameter set $\boldsymbol{\theta}$, are applied to parametrize the solution space of the PDE and a best parameter set $\boldsymbol{\theta}_{\mathcal{D}}$ is identified via minimizing an expectation called the population risk (also known as the population loss):

$$\boldsymbol{\theta}_{\mathcal{D}} = \arg\min_{\boldsymbol{\theta}} R_{\mathcal{D}}(\boldsymbol{\theta}) := \mathbb{E}_{\boldsymbol{x}\sim U(\Omega)}\left[\ell(\mathcal{L}\phi(\boldsymbol{x};\boldsymbol{\theta}), f(\boldsymbol{x}))\right] + \gamma\mathbb{E}_{\boldsymbol{x}\sim U(\partial\Omega)}\left[\ell(\mathcal{B}\phi(\boldsymbol{x};\boldsymbol{\theta}), g(\boldsymbol{x}))\right], \tag{1.2}$$

with a positive parameter γ and a loss function typically taken as $\ell(y,y') = \frac{1}{2}|y-y'|^2$, where the expectation are taken with uniform distributions $U(\Omega)$ and $U(\partial\Omega)$ over Ω and $\partial\Omega$, respectively. To implement the expectation minimization above using the gradient descent method (GD), a discrete set of samples are randomly drawn to obtain an empirical risk (or empirical loss) function

$$R_S(\boldsymbol{\theta}) := \frac{1}{n}\sum_{\{\boldsymbol{x}_i\}_{i=1}^n\subset\Omega} \ell(\mathcal{L}\phi(\boldsymbol{x}_i;\boldsymbol{\theta}), f(\boldsymbol{x}_i)) + \gamma\frac{1}{n}\sum_{\{\boldsymbol{x}_i\}_{i=1}^n\subset\partial\Omega} \ell(\mathcal{B}\phi(\boldsymbol{x}_i;\boldsymbol{\theta}), g(\boldsymbol{x}_i)) \tag{1.3}$$

used in each GD iteration to update $\boldsymbol{\theta}$. The set of random samples is usually renewed per iteration resulting in the SGD algorithm for minimizing (1.2). In this paper, we will focus on the case when these samples are fixed in all iterations. There are mainly three theoretical point of view to study the above deep learning-based PDE solver:

1. **Approximation theory:** given a budget of the size of DNNs, e.g. width m and depth L, or a budget of the total number of parameters N_{para}, what is the accuracy of $\phi(\boldsymbol{x};\boldsymbol{\theta}_{\mathcal{D}})$ approximating the solution of the PDE?

2. **Optimization convergence:** under what condition can gradient descent converges to a global minimizer of (1.2) and (1.3)?

3. **Generalization analysis:** if only finitely many samples are available, how good is the global minimizer of (1.3) compared to the global minimizer of (1.2)?

Deep network approximation theory has shown that DNNs admit powerful approximation capacity. First, DNNs can approximate high-dimensional functions with an appealing approximation rate, e.g., Barron spaces [1, 14, 13], Korobov spaces [34], band-limited functions [6, 36], compositional functions [38, 48], smooth functions [51, 31, 35], solution spaces of certain PDEs [25], and even general continuous functions [45, 44]. Second, DNNs can achieve exponential approximation rates, i.e., the approximation error exponentially decays when the number of parameters increases, for target functions in the polynomial spaces [50, 36, 31], the smooth function spaces [36, 29], the analytic function space [16], the function space admitting a holomorphic extension to a Bernstein polyellipse [37], and even general continuous functions [45]. Theories in deep network approximation have provided attractive upper bounds of the accuracy of $\phi(\boldsymbol{x};\boldsymbol{\theta}_{\mathcal{D}})$ approximating the solution of the PDE in various function spaces. In realistic applications, it might be more interesting to characterize deep network approximation in terms of m and L simultaneously than the characterization in terms of N_{para}. We refer reader to [42, 43, 31, 45, 49] for examples in terms of m and L.

Though DNNs are powerful in terms of approximation theory, obtaining the best DNN $\phi(\boldsymbol{x};\boldsymbol{\theta}_{\mathcal{D}})$ in (1.2) to approximate the PDE solution is still challenging. It is conjectured that, under certain conditions, SGD is able to identify an approximate global minimizer of (1.2) with accuracy depending on N_{para} and the sample size n. Though deep learning-based PDE solvers have been proposed since the 1990s, there might be no existing literature to investigate this conjecture, to the best of our knowledge. In this paper, assuming that the same set of random samples are used in minimizing (1.3), it is shown that GD can converge to a global minimizer of (1.3), denoted as $\boldsymbol{\theta}_S$, for second-order linear PDEs and two-layer neural networks, as long as N_{para} is sufficiently large depending on n, i.e., in the over-parametrization regime. Furthermore, we will quantify how good the global minimizer $\boldsymbol{\theta}_S$ of the empirical loss in (1.3) is compared to the global minimizer $\boldsymbol{\theta}_{\mathcal{D}}$ of the population loss in (1.2), when the empirical loss is regularized with a penalty term using the path norm of $\boldsymbol{\theta}$ and the PDE solution is in a Barron- type space, a variant of the Barron-type space in [1, 14]. Our analysis is an extension of the seminal work of neural tangent kernels [26, 9, 10] and the generalization analysis in [1, 14] for function regression problems to the case of PDE solvers.

Though the convergence of deep learning-based regression under the over-parametrization assumption has been proposed recently [26, 9, 33, 10, 32], we would like to emphasize that the minimization of solving a PDE via (1.2) is more difficult and techinical. In the case of solving PDEs, differential operators have changed the optimization objective function considered in the literature. Balancing between the differential operator and the boundary operator makes it more challenging to solve the optimization problem. For example, we consider a second order elliptic equation with variable coefficients, i.e., $\mathcal{L}u = f$ where $\mathcal{L}u = \sum_{\alpha,\beta=1}^{d} A_{\alpha\beta}(\boldsymbol{x})u_{x_\alpha x_\beta}$. Given a two-layer neural network $\phi(\boldsymbol{x};\boldsymbol{\theta}) = \sum_{k=1}^{m} a_k\sigma(\boldsymbol{w}_k^\intercal \boldsymbol{x})$ with an activation function $\sigma(z) = \max\{0, \frac{1}{6}z^3\}$ to parametrize the PDE solution, solving the original PDE via deep learning is equivalent to solving a regression problem with another type of neural network $f(\boldsymbol{x};\boldsymbol{\theta}) := \mathcal{L}\phi(\boldsymbol{x};\boldsymbol{\theta}) = \sum_{k=1}^{m} a_k\boldsymbol{w}_k^\intercal \boldsymbol{A}(\boldsymbol{x})\boldsymbol{w}_k\sigma''(\boldsymbol{w}_k^\intercal \boldsymbol{x})$ to fit $f(\boldsymbol{x})$. Note that $\sigma''(z) = \text{ReLU}(z) = \max\{0, z\}$. Thus, the dependence of $f(\boldsymbol{x};\boldsymbol{\theta})$ on $\boldsymbol{w}_k$ is essentially cubic rather than linear (more precisely, positive homogeneous).

The generalization analysis of deep learning-based regression under the over-parametrization assumption was studied recently in [26, 4, 7]. The generalization analysis with a regularization term based on the path norm without the over-parametrization assumption was proposed in [14, 13, 15]. In the case of PDE solvers, differential operators have enhanced the nonlinearity of the generalization analysis and hence make it more difficult to analyze. In the case of Linear Kolmogorov Equations and parabolic PDEs, examples of generalization analysis of PDE solvers were presented in [3, 21]. In the case of linear second-order elliptic and parabolic type PDEs, the generalization error of the physics-informed neural network was analyzed in [46]. However, the generalization analysis for generic PDEs is vastly open. Our attempt is for second-order linear PDEs with variable coefficients. Let us consider the second order elliptic equation with variable coefficients in the above paragraph again. The variable coefficients $A_{\alpha\beta}(\boldsymbol{x})$ lead to highly nonlinearity in the network $f(\boldsymbol{x};\boldsymbol{\theta})$ depending on $\boldsymbol{x}$, since we do not make any assumption on the smoothness of $\boldsymbol{A}(\boldsymbol{x})$. We develop new analysis of the Rademacher complexity to overcome these difficulties. Unlike existing work, our a priori estimates do not require any truncation on $f(\boldsymbol{x};\boldsymbol{\theta})$ (or $\phi(\boldsymbol{x};\boldsymbol{\theta})$). This is important because a common truncation trick does not lead to the boundedness of $f(\boldsymbol{x};\boldsymbol{\theta})$ in our PDE solver. In fact, if one considers the standard truncation on $\phi(\boldsymbol{x};\boldsymbol{\theta})$, e.g., $\mathcal{T}_{[0,1]}\phi(\boldsymbol{x};\boldsymbol{\theta}) := \min\{\max\{\phi(\boldsymbol{x};\boldsymbol{\theta}), 0\}, 1\}$, then $\mathcal{L}[\mathcal{T}_{[0,1]}\phi(\boldsymbol{x};\boldsymbol{\theta})]$ might still be unbounded because $\mathcal{L}$ is a second order differential operator. Another naive trick is to truncate $f(\boldsymbol{x};\boldsymbol{\theta})$, i.e., $\mathcal{T}_{[0,1]}f(\boldsymbol{x};\boldsymbol{\theta}) := \min\{\max\{f(\boldsymbol{x};\boldsymbol{\theta}), 0\}, 1\}$. But this does not make sense since we want to find a solution satisfying $\mathcal{L}\phi(\boldsymbol{x};\boldsymbol{\theta}) \approx f(\boldsymbol{x})$ instead of $\mathcal{T}_{[0,1]}\mathcal{L}\phi(\boldsymbol{x};\boldsymbol{\theta}) \approx f(\boldsymbol{x})$.

This paper will be organized as follows. In Section 2, deep learning-based PDE solvers will be introduced in detail. In Section 3, our main theorems for the convergence and generalization analysis of GD for minimizing (1.3) will be presented. In Section 4, the proof of the GD convergence theorems will be shown. In Section 5, the proof of the generalization bound will be given. Finally, we conclude our paper in Section 6.

2 Deep Learning-based PDE Solvers

We will introduce deep learning-based PDE solvers with necessary notations in this paper in preparation for our main theorems in Section 3.

2.1 Notations, Definitions, and Basic Lemmas

The main notations of this paper are listed as follows.

- Vectors and matrices are denoted in bold font. All vectors are column vectors.
- For a parameter set Θ, $\text{vec}\{\Theta\}$ denotes the vector consists of all the elements of Θ.
- $[n]$ denotes $\{1, 2, \ldots, n\}$.
- $\|\cdot\|_1$ and $\|\cdot\|_\infty$ represent the ℓ_1 and ℓ_∞ norms of a vector, respectively.
- Big "O" notation: for any functions $g_1, g_2 : \mathbb{R} \to \mathbb{R}^+$, $g_1(z) = O(g_2(z))$ as $z \to +\infty$ means that $g_1(z) \leq C g_2(z)$ for some constants C, z_0 and any $z \geq z_0$.
- Small "o" notation: for any functions $g_1, g_2 : \mathbb{R} \to \mathbb{R}^+$, $g_1(z) = o(g_2(z))$ as $z \to +\infty$ means that $\lim_{z\to\infty} \frac{f(z)}{g(z)} = 0$.
- Let $\sigma : \mathbb{R} \to \mathbb{R}$ denote the activation function, e.g., $\sigma(x) = \max\{0, \frac{1}{6}x^3\}$ is the activation function used in this paper. With the abuse of notations, we define $\sigma : \mathbb{R}^d \to \mathbb{R}^d$ as $\sigma(\boldsymbol{x}) = (\max\{0, x_1\}, \ldots, \max\{0, x_d\})^\intercal$ for any $\boldsymbol{x} = (x_1, \ldots, x_d)^\intercal \in \mathbb{R}^d$, where $\intercal$ denotes the transpose of a matrix. Similarly, for any function f defined on $\mathbb{R}$ and vector $\boldsymbol{x} \in \mathbb{R}^d$, $f(\boldsymbol{x}) = [f(x_1), \ldots, f(x_d)]^\intercal$.

Mathematically, DNNs are a form of function parametrization via the compositions of simple non-linear functions [17]. Let us focus on the so-called fully connected feed-forward neural network (FNN) defined below. The FNN is a general DNN structure that includes other advanced structures as its special cases, e.g., convolutional neural network [17], ResNet [22], and DenseNet [23].

Definition 2.1 (Fully connected feed-forward neural network (FNN))**.** *An FNN of depth L defined on $\mathbb{R}^d$ is the composition of L simple nonlinear functions as follows:*

$$\phi(\boldsymbol{x}; \boldsymbol{\theta}) := \boldsymbol{a}^\intercal \boldsymbol{h}^{[L]} \circ \boldsymbol{h}^{[L-1]} \circ \cdots \circ \boldsymbol{h}^{[1]}(\boldsymbol{x}),$$

where $\boldsymbol{h}^{[l]}(\boldsymbol{x}) = \sigma\left(\boldsymbol{W}^{[l]}\boldsymbol{x} + \boldsymbol{b}^{[l]}\right)$ with $\boldsymbol{W}^{[l]} \in \mathbb{R}^{m_l \times m_{l-1}}$, $\boldsymbol{b}_l \in \mathbb{R}^{m_l}$ for $l = 1, \ldots, L$, $\boldsymbol{a} \in \mathbb{R}^{m_L}$, $m_0 = d$, and σ is a non-linear activation function. Each $\boldsymbol{h}^{[l]}$ is referred as a hidden layer, m_l is the width of the l-th layer, and L is called the depth of the FNN. $\boldsymbol{\theta} := \text{vec}\{\boldsymbol{a}, \{\boldsymbol{W}^{[l]}, \boldsymbol{b}^{[l]}\}_{l=1}^L\}$ denotes the set of all parameters in ϕ.

Without loss of generality, we consider FNNs omitting $\boldsymbol{b}^{[l]}$'s. In fact, for a network with $\boldsymbol{b}^{[l]}$'s, one can simply set $\tilde{\boldsymbol{x}} = (\boldsymbol{x}^\intercal, 1)^\intercal$ and $\tilde{\boldsymbol{W}}^{[l]} = (\boldsymbol{W}^{[l]}, \boldsymbol{b}^{[l]})$ for each $l \in [L]$, and work on $\boldsymbol{\theta} = \text{vec}\{\boldsymbol{a}, \{\tilde{\boldsymbol{W}}^{[l]}\}_{l=1}^L\}$ by noting that $\tilde{\boldsymbol{W}}^{[l]}\tilde{\boldsymbol{x}} = \boldsymbol{W}^{[l]}\boldsymbol{x} + \boldsymbol{b}^{[l]}$. In this paper, we will focus on networks with $L = 1$.

To analyze PDE solvers, we introduce a new kind of Barron functions with their associated Barron norm, and a path norm defined below.

Definition 2.2 (Path norm)**.** *The path norm of a two-layer neural network*

$$\phi(\boldsymbol{x};\boldsymbol{\theta})=\sum_{k=1}^{m}a_k\sigma(\boldsymbol{w}_k^\intercal\boldsymbol{x}),$$

with an activation function σ and a parameter set $\boldsymbol{\theta}$ is defined as

$$\|\boldsymbol{\theta}\|_{\mathcal{P}}:=\sum_{j=1}^{m}|a_j|\|\boldsymbol{w}_j\|_1^3.$$

Definition 2.3. *A function $f:\Omega\to\mathbb{R}$ is called a Barron-type function if f has an integral representation*

$$f(\boldsymbol{x})=\mathbb{E}_{(a,\boldsymbol{w})\sim\rho}a[\boldsymbol{w}^\intercal\boldsymbol{A}(\boldsymbol{x})\boldsymbol{w}\sigma''(\boldsymbol{w}^\intercal\boldsymbol{x})+\boldsymbol{b}^\intercal(\boldsymbol{x})\boldsymbol{w}\sigma'(\boldsymbol{w}^\intercal\boldsymbol{x})+c(\boldsymbol{x})\sigma(\boldsymbol{w}^\intercal\boldsymbol{x})]\quad\text{for all}\quad\boldsymbol{x}\in\Omega,$$

where ρ is a probability distribution over $\mathbb{R}^{d+1}$. The associated Barron norm of a Barron-type function is defined as

$$\|f\|_{\mathcal{B}}:=\inf_{\rho\in\mathcal{P}_f}\left(\mathbb{E}_{(a,\boldsymbol{w})\sim\rho}|a|^2\|\boldsymbol{w}\|_1^6\right)^{1/2},$$

where $\mathcal{P}_f=\{\rho\mid f(\boldsymbol{x})=\mathbb{E}_{(a,\boldsymbol{w})\sim\rho}a[\boldsymbol{w}^\intercal\boldsymbol{A}(\boldsymbol{x})\boldsymbol{w}\sigma''(\boldsymbol{w}^\intercal\boldsymbol{x})+\boldsymbol{b}^\intercal(\boldsymbol{x})\boldsymbol{w}\sigma'(\boldsymbol{w}^\intercal\boldsymbol{x})+c(\boldsymbol{x})\sigma(\boldsymbol{w}^\intercal\boldsymbol{x})],\boldsymbol{x}\in\Omega\}$. The Barron-type space is defined as $\mathcal{B}(\Omega)=\{f:\Omega\to\mathbb{R}\mid\|f\|_{\mathcal{B}}<\infty\}$.

Since $R_{\mathcal{D}}(\boldsymbol{\theta})$ cannot be realized in realistic applications due to the fact that the empirical loss $R_S(\boldsymbol{\theta})$ of finitely many samples is actually used in the computation, an immediate question is: how well $\phi(\boldsymbol{x};\boldsymbol{\theta}_S)\approx\phi(\boldsymbol{x};\boldsymbol{\theta}_{\mathcal{D}})$? Here $\boldsymbol{\theta}_S$ is a global minimizer when we minimize the empirical loss of $R_S(\boldsymbol{\theta})$. This is the generalization error analysis of deep learning-based PDE solvers and we will use the Rademacher complexity below to estimate the generalization error in terms of $|R_{\mathcal{D}}(\boldsymbol{\theta}_S)-R_S(\boldsymbol{\theta}_S)|$.

Definition 2.4 (The Rademacher complexity of a function class $\mathcal{F}$)**.** *Given a sample set $S=\{z_1,\ldots,z_n\}$ on a domain $\mathcal{Z}$, and a class $\mathcal{F}$ of real-valued functions defined on $\mathcal{Z}$, the empirical Rademacher complexity of $\mathcal{F}$ on S is defined as*

$$\mathrm{Rad}_S(\mathcal{F})=\frac{1}{n}\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{f\in\mathcal{F}}\sum_{i=1}^{n}\tau_i f(z_i)\right],$$

where τ_1, …, τ_n are independent random variables drawn from the Rademacher distribution, i.e., $\mathbb{P}(\tau_i=+1)=\mathbb{P}(\tau_i=-1)=\frac{1}{2}$ for $i=1,\ldots,n$.

The Rademacher complexity is a basic tool for generalization analysis. In our analysis, we will use several important lemmas and theorems related to it. For the purpose of being self-contained, they are listed as follows.

First, we recall a well-known contraction lemma for the Rademacher complexity.

Lemma 2.1 (Contraction lemma [41])**.** *Suppose that $\psi_i:\mathbb{R}\to\mathbb{R}$ is a C_{L}-Lipschitz function for each $i\in[n]$. For any $\boldsymbol{y}\in\mathbb{R}^n$, let $\boldsymbol{\psi}(\boldsymbol{y})=(\psi_1(y_1),\cdots,\psi_n(y_n))^\intercal$. For an arbitrary set of functions $\mathcal{F}$ on an arbitrary domain $\mathcal{Z}$ and an arbitrary choice of samples $S=\{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n\}\subset\mathcal{Z}$, we have*

$$\mathrm{Rad}_S(\boldsymbol{\psi}\circ\mathcal{F})\le C_{\mathrm{L}}\mathrm{Rad}_S(\mathcal{F}).$$

Second, the Rademacher complexity of linear predictors can be characterized by the lemma below.

Lemma 2.2 (Rademacher complexity for linear predictors [41])**.** *Let $\Theta = \{\boldsymbol{w}_1, \cdots, \boldsymbol{w}_m\} \in \mathbb{R}^d$. Let $\mathcal{G} = \{g(\boldsymbol{w}) = \boldsymbol{w}^\intercal \boldsymbol{x} : \|\boldsymbol{x}\|_1 \leq 1\}$ be the linear function class with parameter $\boldsymbol{x}$ whose ℓ^1 norm is bounded by* 1*. Then*

$$\mathrm{Rad}_\Theta(\mathcal{G}) \leq \max_{1\leq k\leq m} \|\boldsymbol{w}_k\|_\infty \sqrt{\frac{2\log(2d)}{m}}.$$

Finally, let us state a general theorem concerning the Rademacher complexity and generalization gap of an arbitrary set of functions $\mathcal{F}$ on an arbitrary domain $\mathcal{Z}$, which is essentially given in [41].

Theorem 2.1 (Rademacher complexity and generalization gap [41])**.** *Suppose that f's in $\mathcal{F}$ are non-negative and uniformly bounded, i.e., for any $f \in \mathcal{F}$ and any $\boldsymbol{z} \in \mathcal{Z}$, $0 \leq f(\boldsymbol{z}) \leq B$. Then for any $\delta \in (0,1)$, with probability at least $1-\delta$ over the choice of n i.i.d. random samples $S = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n\} \subset \mathcal{Z}$, we have*

$$\sup_{f\in\mathcal{F}} \left| \frac{1}{n}\sum_{i=1}^n f(\boldsymbol{z}_i) - \mathbb{E}_{\boldsymbol{z}} f(\boldsymbol{z}) \right| \leq 2\mathbb{E}_S \mathrm{Rad}_S(\mathcal{F}) + B\sqrt{\frac{\log(2/\delta)}{2n}},$$

$$\sup_{f\in\mathcal{F}} \left| \frac{1}{n}\sum_{i=1}^n f(\boldsymbol{z}_i) - \mathbb{E}_{\boldsymbol{z}} f(\boldsymbol{z}) \right| \leq 2\mathrm{Rad}_S(\mathcal{F}) + 3B\sqrt{\frac{\log(4/\delta)}{2n}}.$$

2.2 Expectation Minimization

We will focus on the least-squares method in (1.2) for the boundary value problem (BVP) in (1.1) to discuss the expectation minimization, though the expectation minimization can either be formulated from the least-squares method [2, 47, 39] or the variational formulation [11, 30]. As we shall see in the next subsection, an initial value problem (IVP) can also be formulated into a BVP and solved by the expectation minimization in this subsection.

The objective function in (1.2) consists of two parts: one part for the PDE operator in the domain interior and another part for the boundary condition at the boundary. Therefore, GD has to balance between these two parts and its performance heavily relies on the choice of the parameter γ in (1.2). To remove the hyper-parameter γ and solve the balancing issue, we will introduce special DNNs in [19, 18] satisfying various boundary conditions by design, i.e., $\mathcal{B}\phi(\boldsymbol{x};\boldsymbol{\theta}) = g(\boldsymbol{x})$ is always fulfilled on $\partial\Omega$. Then the expectation minimization in (1.2) is reduced to

$$\boldsymbol{\theta}_\mathcal{D} = \arg\min_{\boldsymbol{\theta}} R_\mathcal{D}(\boldsymbol{\theta}) := \mathbb{E}_{\boldsymbol{x}\in\Omega}\left[\ell(\mathcal{L}\phi(\boldsymbol{x};\boldsymbol{\theta}), f(\boldsymbol{x}))\right]. \tag{2.1}$$

Special neural networks for three types of boundary conditions will be introduced. Without loss of generality, we will take the example of one-dimensional problems on the domain $\Omega = [a,b]$. Networks for more complicated boundary conditions in high-dimensional domains can be constructed similarly.

Case 1. Dirichlet Boundary Conditions: $u(a) = a_0,\ u(b) = b_0$.

In this case, two special functions $h_1(x)$ and $h_2(x)$ are used to augment a neural network $\tilde{\phi}(x;\boldsymbol{\theta})$ to construct the final neural network $\phi(x;\boldsymbol{\theta})$ as the solution network:

$$\phi(x;\boldsymbol{\theta}) = h_1(x)\tilde{\phi}(x;\boldsymbol{\theta}) + h_2(x).$$

$h_1(x)$ and $h_2(x)$ are chosen such that $\phi(x;\boldsymbol{\theta})$ automatically satisfies the Dirichlet boundary conditions no matter what $\boldsymbol{\theta}$ is. Then $\phi(x;\boldsymbol{\theta})$ is trained to satisfy the differential operator in the interior of the domain Ω by solving (2.1).

To achieve this goal, $h_1(x)$ and $h_2(x)$ are constructed for two purposes: 1) construct $h_1(x)$ such that $h_1(x)\tilde{\phi}(x;\boldsymbol{\theta})$ satisfies the homogeneous Dirichlet boundary condition; 2) construct $h_2(x)$ such that $h_2(x)$ satisfies the given inhomogeneous Dirichlet boundary conditions. Therefore, $h_1(x)$ can be set as

$$h_1(x) = (x-a)^{p_a}(x-b)^{p_b},$$

where $0 < p_a,\ p_b \leq 1$, and $h_2(x)$ can be chosen as

$$h_2(x) = (b_0 - a_0)(x-a)/(b-a) + a_0.$$

Note that p_a and p_b should be chosen appropriately to avoid introducing a singular function that $\tilde{\phi}(x;\boldsymbol{\theta})$ needs to approximate. For instance, if the exact PDE solution is $u(x) = (x-a)^s(x-b)^s v(x) + h_1(x)$ with $v(x)$ as a smooth function and $s > 0$, $p_a = p_b > s$ results in $\tilde{\phi}(x;\boldsymbol{\theta}) \approx (x-a)^{s-p_a}(x-b)^{s-p_b}v(x)$, which makes the approximation very challenging.

Case 2. Mixed Boundary Conditions: $u'(a) = a_0$, $u(b) = b_0$.

Similar to Case 1, two special functions $h_1(x)$ and $h_2(x)$ are used to augment a neural network $\tilde{\phi}(x;\boldsymbol{\theta})$ to construct the final neural network $\phi(x;\boldsymbol{\theta})$ as the solution network:

$$\phi(x;\boldsymbol{\theta}) = h_1(x)\tilde{\phi}(x;\boldsymbol{\theta}) + h_2(x).$$

$h_1(x)$ and $h_2(x)$ are chosen such that $\phi(x;\boldsymbol{\theta})$ automatically satisfies the mixed boundary conditions no matter what $\boldsymbol{\theta}$ is. Then $\phi(x;\boldsymbol{\theta})$ is trained to satisfy the differential operator in the interior of the domain Ω by solving (2.1).

To achieve this goal, $h_1(x)$ and $h_2(x)$ are constructed as

$$h_1(x) = (x-a)^{p_a}$$

with $1 < p_a \leq 2$ and $h_2(x)$ can be chosen as

$$h_2(x) = -(b-a)^{p_a}\tilde{\phi}(b;\boldsymbol{\theta}) + a_0 x + b_0 - a_0 b.$$

Case 3. Neumann Boundary Conditions: $u'(a) = a_0$, $u'(b) = b_0$.

Similar to Case 1 and 2, we augment a neural network $\tilde{\phi}(x;\boldsymbol{\theta})$ to construct the final neural network $\phi(x;\boldsymbol{\theta},c_1,c_2)$ as the solution network:

$$\phi(x;\boldsymbol{\theta},c_1,c_2) = \exp(\frac{p_a x}{a-b})(x-a)^{p_a}\big((x-b)^{p_b}\tilde{\phi}(x;\boldsymbol{\theta}) + c_2\big) + c_1 + \frac{(b_0-a_0)}{2(b-a)}(x-a)^2 + a_0 x.$$

where $1 < p_a, p_b \leq 2$, c_1 and c_2 are two parameters to be trained together with $\boldsymbol{\theta}$. Then $\phi(x;\boldsymbol{\theta},c_1,c_2)$ automatically satisfies the Neumann boundary conditions no matter what parameters are and $\phi(x;\boldsymbol{\theta},c_1,c_2)$ is trained to satisfy the differential operator in the interior of the domain Ω by solving (2.1).

2.3 Scope of Analysis and Applications

In Section 2.2, we have simplified the optimization problem from (1.2) to (2.1) for BVP in (1.1). Now we will show that various initial/boundary value problems can be formulated as a BVP in the form of (1.1). This helps us to simplify the optimization convergence and generalization analysis of deep learning-based PDE solvers to the case of BVP in (1.1) solved by (2.1). The analysis of a larger scope of applications has been naturally included in the analysis of BVPs.

Let us assume that the domain $\Omega \subset \mathbb{R}^d$ is bounded. Typical PDE problems of interest can be summerized as:

- Elliptic equation:

$$\begin{aligned} &\mathcal{L}u(\boldsymbol{x}) = f(\boldsymbol{x}) \text{ in } \Omega, \\ &\mathcal{B}u(\boldsymbol{x}) = g_0(\boldsymbol{x}) \text{ on } \partial\Omega. \end{aligned} \tag{2.2}$$

- Parabolic equation:

$$\begin{aligned} &\frac{\partial u(\boldsymbol{x},t)}{\partial t} - \mathcal{L}u(\boldsymbol{x},t) = f(\boldsymbol{x},t) \text{ in } \Omega \times (0,T), \\ &\mathcal{B}u(\boldsymbol{x},t) = g_0(\boldsymbol{x},t) \text{ on } \partial\Omega \times (0,T), \\ &u(\boldsymbol{x},0) = h_0(\boldsymbol{x}) \text{ in } \Omega. \end{aligned} \tag{2.3}$$

- Hyperbolic equation:

$$\begin{aligned} &\frac{\partial^2 u(\boldsymbol{x},t)}{\partial t^2} - \mathcal{L}u(\boldsymbol{x},t) = f(\boldsymbol{x},t) \text{ in } \Omega \times (0,T), \\ &\mathcal{B}u(\boldsymbol{x},t) = g_0(\boldsymbol{x},t) \text{ on } \partial\Omega \times (0,T), \\ &u(\boldsymbol{x},0) = h_0(\boldsymbol{x}), \quad \frac{\partial u(\boldsymbol{x},0)}{\partial t} = h_1(\boldsymbol{x}) \text{ in } \Omega. \end{aligned} \tag{2.4}$$

In the above equations, u is the unknown solution function; f, g_0, h_0, h_1 are given data functions; $\mathcal{L}$ is a spatial differential operator with respect to x; $\mathcal{B}$ is a boundary operator specifying a certain type of boundary conditions.

As discussed in [19], when the temporal variable t is treated as an extra spatial coordinate, we can unify the above initial/boundary value problems in (2.2)-(2.4) in the following form

$$\begin{aligned} &\mathcal{L}u(\boldsymbol{y}) = f(\boldsymbol{y}) \text{ in } Q, \\ &\mathcal{B}u(\boldsymbol{y}) = g(\boldsymbol{y}) \text{ in } \Gamma, \end{aligned} \tag{2.5}$$

where $\boldsymbol{y}$ includes the spatial variable $\boldsymbol{x}$ and possibly the temporal variable t; $\mathcal{L}u = f$ represents a generic time-independent PDE; $\mathcal{B}u = g$ specifies the original boundary condition on $\boldsymbol{x}$ and possibly the initial condition of t; Q and Γ are the corresponding new domains of the equations. For the purpose of convenience, we will still use the BVP in (1.1) instead of (2.5) afterwards.

Though deep learning-based PDE solvers work for high-order differential equations in general domains, we consider second order differential equations with variable coefficients in $\Omega = [0,1]^d$ in our analysis. The generalization to high-order differential equations and other domains follows straightforwardly and we leave it as future work. We will use the second order differential operator $\mathcal{L}$ in a non-divergence form

$$\mathcal{L}u = \sum_{\alpha,\beta=1}^{d} A_{\alpha\beta}(\boldsymbol{x}) u_{x_\alpha x_\beta} + \sum_{\alpha=1}^{d} b_\alpha(\boldsymbol{x}) u_{x_\alpha} + c(\boldsymbol{x})u. \tag{2.6}$$

If $\mathcal{L}$ is in a divergence form, e.g.,

$$\mathcal{L}u = \sum_{\alpha,\beta=1}^{d} \left(A_{\alpha\beta}(\boldsymbol{x}) u_{x_\alpha}\right)_{x_\beta} + \sum_{\alpha=1}^{d} b_\alpha(\boldsymbol{x}) u_{x_\alpha} + c(\boldsymbol{x})u,$$

then we can represent it in a non-divergence form as

$$\mathcal{L}u = \sum_{\alpha,\beta=1}^{d} A_{\alpha\beta}(\boldsymbol{x}) u_{x_\alpha x_\beta} + \sum_{\alpha=1}^{d} \hat{b}_\alpha(\boldsymbol{x}) u_{x_\alpha} + c(\boldsymbol{x}) u$$

with

$$\hat{b}_\alpha = b_\alpha + \sum_{\beta=1}^{d} \frac{\partial A_{\alpha\beta}}{\partial x_\beta}.$$

Recall that we introduce two functions $h_1(\boldsymbol{x})$ and $h_2(\boldsymbol{x})$ to augment a neural network $\tilde{\phi}(x; \boldsymbol{\theta})$ to construct the final neural network

$$\phi(\boldsymbol{x}; \boldsymbol{\theta}) = h_1(\boldsymbol{x}) \tilde{\phi}(\boldsymbol{x}; \boldsymbol{\theta}) + h_2(\boldsymbol{x})$$

as the solution network that automatically satisfies given Dirichlet boundary conditions, which makes it sufficient to solve the optimization problem in (2.1) to get the desired neural network. In this case, $\mathcal{L}\phi(\boldsymbol{x}; \boldsymbol{\theta}) = f(\boldsymbol{x})$ is equivalent to $\tilde{\mathcal{L}}\tilde{\phi}(\boldsymbol{x}; \boldsymbol{\theta}) = \tilde{f}(\boldsymbol{x})$, where

$$\tilde{\mathcal{L}} = \sum_{\alpha,\beta=1}^{d} \tilde{A}_{\alpha\beta}(\boldsymbol{x}) u_{x_\alpha x_\beta} + \sum_{\alpha=1}^{d} \tilde{b}_\alpha(\boldsymbol{x}) u_{x_\alpha} + \tilde{c}(\boldsymbol{x}),$$

$$\tilde{A}_{\alpha\beta}(\boldsymbol{x}) = A_{\alpha\beta}(\boldsymbol{x}) h_1(\boldsymbol{x}),$$

$$\tilde{b}_\alpha(\boldsymbol{x}) = b_\alpha(\boldsymbol{x}) h_1(\boldsymbol{x}) + \sum_{\beta=1}^{d} \left(A_{\alpha\beta}(\boldsymbol{x}) + A_{\beta\alpha}(\boldsymbol{x})\right) \partial_{x_\beta} h_1(\boldsymbol{x}),$$

$$\tilde{c}(\boldsymbol{x}) = \sum_{\alpha,\beta=1}^{d} A_{\alpha\beta}(\boldsymbol{x}) \partial_{x_\alpha} \partial_{x_\beta} h_1(\boldsymbol{x}) + \sum_{\alpha=1}^{d} b_\alpha(\boldsymbol{x}) \partial_{x_\alpha} h_1(\boldsymbol{x}) + c(\boldsymbol{x}) h_1(\boldsymbol{x}),$$

and

$$\tilde{f}(\boldsymbol{x}) = f(\boldsymbol{x}) - \mathcal{L}(h_2(\boldsymbol{x})).$$

Therefore, the optimization convergence and generalization analysis of (2.1) is equivalent to

$$\boldsymbol{\theta}_{\mathcal{D}} = \arg\min_{\boldsymbol{\theta}} R_{\mathcal{D}}(\boldsymbol{\theta}) := \mathbb{E}_{\boldsymbol{x} \in \Omega} \left[\ell(\tilde{\mathcal{L}}\tilde{\phi}(\boldsymbol{x}; \boldsymbol{\theta}), \tilde{f}(\boldsymbol{x})) \right], \tag{2.7}$$

which gives

$$\phi(\boldsymbol{x}; \boldsymbol{\theta}_{\mathcal{D}}) = h_1(\boldsymbol{x}) \tilde{\phi}(\boldsymbol{x}; \boldsymbol{\theta}_{\mathcal{D}}) + h_2(\boldsymbol{x})$$

as a best solution to the PDE in (1.1) parametrized by DNNs. The corresponding empirical risk is

$$R_S(\boldsymbol{\theta}) := \frac{1}{n} \sum_{\{\boldsymbol{x}_i\}_{i=1}^{n} \subset \Omega} \ell(\tilde{\mathcal{L}}\tilde{\phi}(\boldsymbol{x}_i; \boldsymbol{\theta}), \tilde{f}(\boldsymbol{x}_i)), \tag{2.8}$$

which gives $\boldsymbol{\theta}_S = \arg\min_{\boldsymbol{\theta}} R_S(\boldsymbol{\theta})$ and

$$\phi(\boldsymbol{x}; \boldsymbol{\theta}_S) = h_1(\boldsymbol{x}) \tilde{\phi}(\boldsymbol{x}; \boldsymbol{\theta}_S) + h_2(\boldsymbol{x}).$$

Similarly, in the case of other two types of boundary conditions, the corresponding optimization problem in (1.2) can also be transformed to (2.7) and its discretization in (2.8) with an appropriate differential operator $\tilde{\mathcal{L}}$ and a right-hand-side function $\tilde{f}$.

In sum, the discussion in Section 2.2 and here indicates that the optimization and generalization analysis of deep learning-based PDE solvers for various IVPs and BVPs with different boundary conditions can be reduced to the analysis of (2.7) and (2.8) with $\tilde{\mathcal{L}}$ in a non-divergence form. In the next section, we will present our main theorems for this analysis. For simplicity, we will still use the notation of $\mathcal{L}$ and f instead of $\tilde{\mathcal{L}}$ and $\tilde{f}$ in our analysis afterwards.

3 Main Results

In this section, we introduce our main results on the convergence of GD and the generalization error of neural network-based least-squares solvers for PDEs using two-layer neural networks on $\Omega = [0,1]^d$. Throughout our analysis, we we assume $|f| \leq 1$ and focus on second-order differential operators $\mathcal{L}$ given in (2.6) satisfying the assumption below.

Assumption 3.1 (Symmetry and boundedness of $\mathcal{L}$)**.** *Throughout the analysis of this paper, we assume $\mathcal{L}$ in* (2.6) *satisfies the condition: there exists $M \geq 1$*[1] *such that for all $\boldsymbol{x} \in \Omega = [0,1]^d$, $\alpha, \beta \in [d]$, we have $A_{\alpha\beta} = A_{\beta\alpha}$*

$$|A_{\alpha\beta}(\boldsymbol{x})| \leq M, \quad |b_\alpha(\boldsymbol{x})| \leq M, \quad \textit{and} \quad |c(\boldsymbol{x})| \leq M. \tag{3.1}$$

First, we show that, under suitable assumptions, the emprical risk $R_S(\boldsymbol{\theta})$ of the PDE solution represented by an over-parametrized two-layer neural network converges to zero, i.e., achieving a global minimizer, with a linear convergence rate by GD. In particular, as discussed in Section 2, it is sufficient to prove the convergence for minimizing the empirical loss

$$\boldsymbol{\theta}_S = \arg\min_{\boldsymbol{\theta}} R_S(\boldsymbol{\theta}) := \frac{1}{n} \sum_{S=\{\boldsymbol{x}_i\}_{i=1}^n \subset \Omega} \ell(\mathcal{L}\phi(\boldsymbol{x}_i;\boldsymbol{\theta}), f(\boldsymbol{x}_i)), \tag{3.2}$$

where $S := \{\boldsymbol{x}_i\}_{i=1}^n$ is a given set of i.i.d. samples with the uniform distribution $\mathcal{D}$ over $\Omega = [0,1]^d$, and the two-layer neural network used here is constructed as

$$\phi(\boldsymbol{x};\boldsymbol{\theta}) = \sum_{k=1}^m a_k \sigma(\boldsymbol{w}_k^\intercal \boldsymbol{x}), \tag{3.3}$$

where for $k \in [m]$, $a_k \in \mathbb{R}$, $\boldsymbol{w}_k \in \mathbb{R}^d$, $\boldsymbol{\theta} = \mathrm{vec}\{a_k, \boldsymbol{w}_k\}_{k=1}^m$, and $\sigma(x) = \max\{\frac{1}{6}x^3, 0\}$. Our main result of the linear convergence rate is summarized in Theorem 3.1 below.

Theorem 3.1 (Linear convergence rate)**.** *Let $\boldsymbol{\theta}^0 := \mathrm{vec}\{a_k^0, \boldsymbol{w}_k^0\}_{k=1}^m$ at the GD initialization for solving* (3.2)*, where $a_k^0 \sim \mathcal{N}(0,\gamma^2)$ and $\boldsymbol{w}_k^0 \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$ with any $\gamma \in (0,1)$. Let $C_d := \mathbb{E}\|\boldsymbol{w}\|_1^{12} < +\infty$ with $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$ and λ_S be a positive constant in Assumption 4.1. For any $\delta \in (0,1)$, if*

$$\begin{aligned} m \geq \max\Bigg\{ & \frac{512 n^4 M^4 C_d}{\lambda_S^2 \delta}, \frac{200\sqrt{2} M d^3 n \log(4m(d+1)/\delta) \sqrt{R_S(\boldsymbol{\theta}^0)}}{\lambda_S}, & (3.4) \\ & \frac{2^{23} M^3 d^9 n^2 (\log(4m(d+1)/\delta))^4 \sqrt{R_S(\boldsymbol{\theta}^0)}}{\lambda_S^2} \Bigg\}, & (3.5) \end{aligned}$$

then with probability at least $1-\delta$ over the random initialization $\boldsymbol{\theta}^0$, we have, for all $t \geq 0$,

$$R_S(\boldsymbol{\theta}(t)) \leq \exp\left(-\frac{m\lambda_S t}{n}\right) R_S(\boldsymbol{\theta}^0).$$

[1] The upper bound M is not necessarily greater than 1. We set this for simplicity.

Remark 3.1. *For the estimate of $R_S(\boldsymbol{\theta}^0)$, see Lemma 4.2. In particular, if $\gamma = O(\frac{1}{\sqrt{m}(\log m)^2})$, then $R_S(\boldsymbol{\theta}^0) = O(1)$. One may also use the Anti-Symmetrical Initialization (ASI) [52], a general but simple trick that ensures $R_S(\boldsymbol{\theta}^0) \leq \frac{1}{2}$.*

Second, we prove that the a posteriori generalization error $|R_{\mathcal{D}}(\boldsymbol{\theta}) - R_S(\boldsymbol{\theta})|$ is bounded by $O\left(\frac{\|\boldsymbol{\theta}\|_{\mathcal{P}}^2 \log\|\boldsymbol{\theta}\|_{\mathcal{P}}}{\sqrt{n}}\right)$, where $\|\boldsymbol{\theta}\|_{\mathcal{P}}$ is the path norm introduced in Definition 2.2, and the a priori generalization error $R_{\mathcal{D}}(\boldsymbol{\theta}_{S,\lambda})$ is bounded by $O\left(\frac{\|f\|_{\mathcal{B}}^2}{m}\right) + O\left(\frac{\|f\|_{\mathcal{B}}^2 \log\|f\|_{\mathcal{B}}}{\sqrt{n}}\right)$, where $\|f\|_{\mathcal{B}}$ is the Barron norm for Barron-type functions $f(\boldsymbol{x})$ introduced in Definition 2.3, and $\boldsymbol{\theta}_{S,\lambda}$ is a global minimizer of a regularized empirical loss using the path norm. Our results of the generalization errors can be summarized in Theorems 3.2 and 3.3 below.

Theorem 3.2 (A posteriori generalization bound)**.** *For any $\delta \in (0,1)$, with probability at least $1-\delta$ over the choice of random samples $S := \{\boldsymbol{x}_i\}_{i=1}^n$ in* (3.2)*, for any two-layer neural network $\phi(\boldsymbol{x};\boldsymbol{\theta})$ in* (3.3)*, we have*

$$|R_{\mathcal{D}}(\boldsymbol{\theta}) - R_S(\boldsymbol{\theta})| \leq \frac{(\|\boldsymbol{\theta}\|_{\mathcal{P}}+1)^2}{\sqrt{n}} 2M^2(14d^2\sqrt{2\log(2d)} + \log[\pi(\|\boldsymbol{\theta}\|_{\mathcal{P}}+1)] + \sqrt{2\log(1/3\delta)}).$$

Theorem 3.3 (A priori generalization bound)**.** *Suppose that $f(\boldsymbol{x})$ is in the Barron-type space $\mathcal{B}([0,1]^d)$ and $\lambda \geq 4M^2[2 + 14d^2\sqrt{2\log(2d)} + \sqrt{2\log(2/3\delta)}]$. Let*

$$\boldsymbol{\theta}_{S,\lambda} = \arg\min_{\boldsymbol{\theta}} J_{S,\lambda}(\boldsymbol{\theta}) := R_S(\boldsymbol{\theta}) + \frac{\lambda}{\sqrt{n}}\|\boldsymbol{\theta}\|_{\mathcal{P}}^2 \log[\pi(\|\boldsymbol{\theta}\|_{\mathcal{P}}+1)].$$

Then for any $\delta \in (0,1)$, with probability at least $1-\delta$ over the choice of random samples $S := \{\boldsymbol{x}_i\}_{i=1}^n$ in (3.2)*, we have*

$$\begin{aligned} R_{\mathcal{D}}(\boldsymbol{\theta}_{S,\lambda}) &:= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\tfrac{1}{2}(\mathcal{L}\phi(\boldsymbol{x};\boldsymbol{\theta}_{S,\lambda}) - f(\boldsymbol{x}))^2 \\ &\leq \frac{6M^2\|f\|_{\mathcal{B}}^2}{m} + \frac{\|f\|_{\mathcal{B}}^2+1}{\sqrt{n}}(4\lambda + 16M^2)\left\{\log[\pi(2\|f\|_{\mathcal{B}}+1)] + 14d^2\sqrt{\log(2d)} + \sqrt{\log(2/3\delta)}\right\}. \end{aligned} \tag{3.6}$$

The proof of Theorem 3.1 will be given in Section 4 and the proofs of Theorems 3.2 and 3.3 will be presented in Section 5.

4 Global Convergence of Gradient Descent

In this section, we will prove the global convergence of GD with a linear convergence rate for deep learning-based PDE solvers as stated in Theorem 3.1. We will first summarize the notations and assumptions for the proof of Theorem 3.1 in Section 4.1. Several important lemmas will be proved in Section 4.2. Finally, Theorem 3.1 is proved in Section 4.3.

4.1 Notations and Main Ideas

Let us first summarize the notations and assumptions used in the proof of Theorem 3.1.

Recall that we use the two-layer neural network $\phi(\boldsymbol{x};\boldsymbol{\theta})$ in (3.3) with $\boldsymbol{\theta} = \mathrm{vec}\{a_k, \boldsymbol{w}_k\}_{k=1}^m$. In the GD iteration, we use t to denote the iteration or the artificial time variable in the gradient flow. Hence, we define the following notations for the evolution of parameters at time t:

$$a_k^t := a_k(t), \quad \boldsymbol{w}_k^t := \boldsymbol{w}_k(t), \quad \boldsymbol{\theta}^t := \boldsymbol{\theta}(t) := \mathrm{vec}\{a_k^t, \boldsymbol{w}_k^t\}_{k=1}^m.$$

In the analysis, we also use $\bar{a}_k^t := \bar{a}_k(t) := \gamma^{-1} a_k(t)$ with $0 < \gamma < 1$, e.g., $\gamma = \frac{1}{\sqrt{m}}$ or $\gamma = \frac{1}{m}$. $\bar{\boldsymbol{\theta}}(t)$ means $\mathrm{vec}\{\bar{a}_k^t, \boldsymbol{w}_k^t\}_{k=1}^m$. Similarly, we can introduce t to other functions or variables depending on $\boldsymbol{\theta}(t)$. When the dependency of t is clear, we will drop the index t. In the initialization of GD, we set

$$a_k^0 := a_k(0) \sim \mathcal{N}(0, \gamma^2), \quad \boldsymbol{w}_k^0 := \boldsymbol{w}_k(0) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d), \quad \boldsymbol{\theta}^0 := \boldsymbol{\theta}(0) := \mathrm{vec}\{a_k^0, \boldsymbol{w}_k^0\}_{k=1}^m. \tag{4.1}$$

Note that we use $\sigma(x) = \max\{\frac{1}{6}x^3, 0\}$ as the activation of our two-layer neural network. Therefore, $\sigma'(x) = \max\{\frac{1}{2}x^2, 0\}$, and $\sigma''(x) = \mathrm{ReLU}(x) = \max\{x, 0\}$. For simplicity, we define

$$\begin{aligned} f_{\boldsymbol{\theta}}(\boldsymbol{x}) &:= f(\boldsymbol{x}; \boldsymbol{\theta}) := \mathcal{L}\phi(\boldsymbol{x}; \boldsymbol{\theta}) \\ &= \sum_{k=1}^m a_k [\boldsymbol{w}_k^\intercal \boldsymbol{A}(\boldsymbol{x}) \boldsymbol{w}_k \sigma''(\boldsymbol{w}_k^\intercal \boldsymbol{x}) + \boldsymbol{b}^\intercal(\boldsymbol{x}) \boldsymbol{w}_k \sigma'(\boldsymbol{w}_k^\intercal \boldsymbol{x}) + c(\boldsymbol{x}) \sigma(\boldsymbol{w}_k^\intercal \boldsymbol{x})], \end{aligned} \tag{4.2}$$

which can be treated as a special two-layer neural network for a regression problem $f_{\boldsymbol{\theta}}(\boldsymbol{x}) \approx f(\boldsymbol{x})$.

For simplicity, we denote $e_i = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - f(\boldsymbol{x}_i)$ for $i \in [n]$ and $\boldsymbol{e} = (e_1, e_2, \ldots, e_n)^\intercal$. Then the empirical risk can be written as

$$R_S(\boldsymbol{\theta}) = \frac{1}{2n} \sum_{i=1}^n (f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - f(\boldsymbol{x}_i))^2 = \frac{1}{2n} \boldsymbol{e}^\intercal \boldsymbol{e}.$$

Hence, the GD dynamics is

$$\dot{\boldsymbol{\theta}} = -\nabla_{\boldsymbol{\theta}} R_S(\boldsymbol{\theta}), \tag{4.3}$$

or equivalently in terms of a_k and $\boldsymbol{w}_k$ as follows:

$$\begin{aligned} \dot{a}_k &= -\nabla_{a_k} R_S(\boldsymbol{\theta}) = -\frac{1}{n} \sum_{i=1}^n e_i \left[\boldsymbol{w}_k^\intercal \boldsymbol{A}(\boldsymbol{x}_i) \boldsymbol{w}_k \sigma''(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i) + \boldsymbol{b}^\intercal(\boldsymbol{x}_i) \boldsymbol{w}_k \sigma'(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i) + c(\boldsymbol{x}_i) \sigma(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i)\right], \\ \dot{\boldsymbol{w}}_k &= -\nabla_{\boldsymbol{w}_k} R_S(\boldsymbol{\theta}) = -\frac{1}{n} \sum_{i=1}^n e_i a_k \Big[2\boldsymbol{A}(\boldsymbol{x}_i) \boldsymbol{w}_k \sigma''(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i) + \boldsymbol{w}_k^\intercal \boldsymbol{A}(\boldsymbol{x}_i) \boldsymbol{w}_k \sigma^{(3)}(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i) \boldsymbol{x}_i \\ &\qquad + \sigma'(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i) \boldsymbol{b}(\boldsymbol{x}_i) + \boldsymbol{b}^\intercal(\boldsymbol{x}_i) \boldsymbol{w}_k \sigma''(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i) \boldsymbol{x}_i + c(\boldsymbol{x}_i) \sigma'(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i) \boldsymbol{x}_i\Big]. \end{aligned}$$

Adopting the neuron tangent kernel point of view [26], in the case of a two-layer neural network with an infinite width, the corresponding kernels $k^{(a)}$ for parameters in the last linear transform and $k^{(w)}$ for parameters in the first layer are functions from $\Omega \times \Omega$ to $\mathbb{R}$ defined by

$$\begin{aligned} k^{(a)}(\boldsymbol{x}, \boldsymbol{x}') &:= \mathbb{E}_{\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)} g^{(a)}(\boldsymbol{w}; \boldsymbol{x}, \boldsymbol{x}'), \\ k^{(w)}(\boldsymbol{x}, \boldsymbol{x}') &:= \mathbb{E}_{(a, \boldsymbol{w}) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{d+1})} g^{(w)}(a, \boldsymbol{w}; \boldsymbol{x}, \boldsymbol{x}'), \end{aligned}$$

where

$$\begin{aligned} g^{(a)}(\boldsymbol{w}; \boldsymbol{x}, \boldsymbol{x}') &:= \left[\boldsymbol{w}^\intercal \boldsymbol{A}(\boldsymbol{x}) \boldsymbol{w} \sigma''(\boldsymbol{w}^\intercal \boldsymbol{x}) + \boldsymbol{b}^\intercal(\boldsymbol{x}) \boldsymbol{w} \sigma'(\boldsymbol{w}^\intercal \boldsymbol{x}) + c(\boldsymbol{x}) \sigma(\boldsymbol{w}^\intercal \boldsymbol{x})\right] \\ &\quad \cdot \left[\boldsymbol{w}^\intercal \boldsymbol{A}(\boldsymbol{x}') \boldsymbol{w} \sigma''(\boldsymbol{w}^\intercal \boldsymbol{x}') + \boldsymbol{b}^\intercal(\boldsymbol{x}') \boldsymbol{w} \sigma'(\boldsymbol{w}^\intercal \boldsymbol{x}') + c(\boldsymbol{x}') \sigma(\boldsymbol{w}^\intercal \boldsymbol{x}')\right], \\ g^{(w)}(a, \boldsymbol{w}; \boldsymbol{x}, \boldsymbol{x}') &:= a^2 \big[2\boldsymbol{A}(\boldsymbol{x}) \boldsymbol{w} \sigma''(\boldsymbol{w}^\intercal \boldsymbol{x}) + \boldsymbol{w}^\intercal \boldsymbol{A}(\boldsymbol{x}) \boldsymbol{w} \sigma^{(3)}(\boldsymbol{w}^\intercal \boldsymbol{x}) \boldsymbol{x} + \sigma'(\boldsymbol{w}^\intercal \boldsymbol{x}) \boldsymbol{b}(\boldsymbol{x}) \\ &\quad + \boldsymbol{b}^\intercal(\boldsymbol{x}) \boldsymbol{w} \sigma''(\boldsymbol{w}^\intercal \boldsymbol{x}) \boldsymbol{x} + c(\boldsymbol{x}) \sigma'(\boldsymbol{w}^\intercal \boldsymbol{x}) \boldsymbol{x}\big] \cdot \big[2\boldsymbol{A}(\boldsymbol{x}') \boldsymbol{w} \sigma''(\boldsymbol{w}^\intercal \boldsymbol{x}') \\ &\quad + \boldsymbol{w}^\intercal \boldsymbol{A}(\boldsymbol{x}') \boldsymbol{w} \sigma^{(3)}(\boldsymbol{w}^\intercal \boldsymbol{x}') \boldsymbol{x}' + \sigma'(\boldsymbol{w}^\intercal \boldsymbol{x}') \boldsymbol{b}(\boldsymbol{x}') \\ &\quad + \boldsymbol{b}^\intercal(\boldsymbol{x}') \boldsymbol{w} \sigma''(\boldsymbol{w}^\intercal \boldsymbol{x}') \boldsymbol{x}' + c(\boldsymbol{x}) \sigma'(\boldsymbol{w}^\intercal \boldsymbol{x}') \boldsymbol{x}'\big]. \end{aligned}$$

These kernels evaluated at $n \times n$ pairs of samples lead to $n \times n$ Gram matrices $\boldsymbol{K}^{(a)}$ and $\boldsymbol{K}^{(\boldsymbol{w})}$ with $K_{ij}^{(a)} = k^{(a)}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $K_{ij}^{(w)} = k^{(w)}(\boldsymbol{x}_i, \boldsymbol{x}_j)$, respectively. Our analysis requires the matrix $\boldsymbol{K}^{(a)}$ to be positive definite, which has been verified for regression problems under mild conditions on random training data $S = \{\boldsymbol{x}_i\}_{i=1}^n$ and can be generalized to our case. Hence, we assume this as follows for simplicity.

Assumption 4.1. *We assume that*

$$\lambda_S := \lambda_{\min}\left(\boldsymbol{K}^{(a)}\right) > 0.$$

For a two-layer neural network with m neurons, the $n \times n$ Gram matrix $\boldsymbol{G}(\boldsymbol{\theta}) = \boldsymbol{G}^{(a)}(\boldsymbol{\theta}) + \boldsymbol{G}^{(w)}(\boldsymbol{\theta})$ is given by the following expressions for the (i, j)-th entry

$$\boldsymbol{G}_{ij}^{(a)}(\boldsymbol{\theta}) := \frac{1}{m}\sum_{k=1}^{m} g^{(a)}(\boldsymbol{w}_k; \boldsymbol{x}_i, \boldsymbol{x}_j),$$
$$\boldsymbol{G}_{ij}^{(w)}(\boldsymbol{\theta}) := \frac{1}{m}\sum_{k=1}^{m} g^{(w)}(a_k, \boldsymbol{w}_k; \boldsymbol{x}_i, \boldsymbol{x}_j).$$

Clearly, $\boldsymbol{G}^{(a)}(\boldsymbol{\theta})$ and $\boldsymbol{G}^{(w)}(\boldsymbol{\theta})$ are both positive semi-definite for any $\boldsymbol{\theta}$. By using the Gram matrix $\boldsymbol{G}(\boldsymbol{\theta})$, we have the following evolution equations to understand the dynamics of GD:

$$\frac{\mathrm{d}}{\mathrm{d}t} f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) = -\frac{1}{n}\sum_{j=1}^{n} \boldsymbol{G}_{ij}(\boldsymbol{\theta})(f_{\boldsymbol{\theta}}(\boldsymbol{x}_j) - f(\boldsymbol{x}_j))$$

and

$$\frac{\mathrm{d}}{\mathrm{d}t} R_S(\boldsymbol{\theta}) = -\|\nabla_{\boldsymbol{\theta}} R_S(\boldsymbol{\theta})\|_2^2 = -\frac{m}{n^2}\boldsymbol{e}^\intercal \boldsymbol{G}(\boldsymbol{\theta})\boldsymbol{e} \leq -\frac{m}{n^2}\boldsymbol{e}^\intercal \boldsymbol{G}^{(a)}(\boldsymbol{\theta})\boldsymbol{e}. \tag{4.4}$$

Our goal is to show that the above evolution equation has a solution $f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$ converging to $f(\boldsymbol{x}_i)$ for all training samples $\boldsymbol{x}_i$, or equivalently, to show that $R_S(\boldsymbol{\theta})$ converges to zero. These goals are true if the smallest eigenvalue $\lambda_{\min}\left(\boldsymbol{G}^{(a)}(\boldsymbol{\theta})\right)$ of $\boldsymbol{G}^{(a)}(\boldsymbol{\theta})$ has a positive lower bound uniformly in t, since in this case we can solve (4.4) and bound $R_S(\boldsymbol{\theta})$ with a function in t converging to zero when $t \to \infty$ as shown in Lemma 4.4. In fact, a uniform lower bound of $\lambda_{\min}\left(\boldsymbol{G}^{(a)}(\boldsymbol{\theta})\right)$ can be $\frac{1}{2}\lambda_S$, which can be proved in the following three steps:

- (**Initial phase**) By Assumption 4.1 of $\boldsymbol{K}^{(a)}$, we can show $\lambda_{\min}\left(\boldsymbol{G}^{(a)}(\boldsymbol{\theta}(0))\right) \approx \lambda_S$ in Lemma 4.3 using the observation that $\boldsymbol{K}_{ij}^{(a)}$ is the mean of $g(\boldsymbol{w}; \boldsymbol{x}_i, \boldsymbol{x}_j)$ over the normal random variable $\boldsymbol{w}$, while $\boldsymbol{G}_{ij}^{(a)}(\boldsymbol{\theta}(0))$ is the mean of $g(\boldsymbol{w}; \boldsymbol{x}_i, \boldsymbol{x}_j)$ with m independent realizations.
- (**Evolution phase**) The GD dynamics results in $\boldsymbol{\theta}(t) \approx \boldsymbol{\theta}(0)$ under the assumption of over-parametrization as shown in Lemma 4.5, which indicates that
$$\lambda_{\min}\left(\boldsymbol{G}^{(a)}(\boldsymbol{\theta}(0))\right) \approx \lambda_{\min}\left(\boldsymbol{G}^{(a)}(\boldsymbol{\theta}(t))\right).$$
- (**Final phase**) To show the uniform bound $\lambda_{\min}\left(\boldsymbol{G}^{(a)}(\boldsymbol{\theta}(t))\right) \geq \frac{1}{2}\lambda_S$ for all $t \geq 0$, we introduce a stopping time t^* via
$$t^* = \inf\{t \mid \boldsymbol{\theta}(t) \notin \mathcal{M}(\boldsymbol{\theta}^0)\}, \tag{4.5}$$
where
$$\mathcal{M}(\boldsymbol{\theta}^0) := \left\{\boldsymbol{\theta} \mid \|\boldsymbol{G}^{(a)}(\boldsymbol{\theta}) - \boldsymbol{G}^{(a)}(\boldsymbol{\theta}^0)\|_{\mathrm{F}} \leq \frac{1}{4}\lambda_S\right\}, \tag{4.6}$$
and show that t^* is in fact equal to infinity in the final proof of Theorem 3.1 in Section 4.3.

4.2 Proofs of Lemmas for Theorem 3.1

In this subsection, we will prove several lemmas in preparation for the proof of Theorem 3.1.

Lemma 4.1. *For any $\delta \in (0,1)$ with probability at least $1-\delta$ over the random initialization in* (4.1)*, we have*

$$\begin{aligned}\max_{k\in[m]}\left\{|\bar{a}_k^0|,\ \|\boldsymbol{w}_k^0\|_\infty\right\} &\leq \sqrt{2\log\frac{2m(d+1)}{\delta}},\\ \max_{k\in[m]}\left\{|a_k^0|\right\} &\leq \gamma\sqrt{2\log\frac{2m(d+1)}{\delta}}.\end{aligned} \tag{4.7}$$

Proof. If $\mathrm{X}\sim\mathcal{N}(0,1)$, then $\mathbb{P}(|\mathrm{X}|>\varepsilon)\leq 2\mathrm{e}^{-\frac{1}{2}\varepsilon^2}$ for all $\varepsilon>0$. Since $\bar{a}_k^0\sim\mathcal{N}(0,1)$, $(\boldsymbol{w}_k^0)_\alpha\sim\mathcal{N}(0,1)$ for $k\in[m],\alpha\in[d]$, and they are all independent, by setting

$$\varepsilon = \sqrt{2\log\frac{2m(d+1)}{\delta}},$$

one can obtain

$$\begin{aligned}\mathbb{P}\left(\max_{k\in[m]}\left\{|\bar{a}_k^0|,\|\boldsymbol{w}_k^0\|_\infty\right\}>\varepsilon\right) &= \mathbb{P}\left(\left(\bigcup_{k\in[m]}\left\{|\bar{a}_k^0|>\varepsilon\right\}\right)\bigcup\left(\bigcup_{k\in[m],\alpha\in[d]}\left\{|(\boldsymbol{w}_k^0)_\alpha|>\varepsilon\right\}\right)\right)\\ &\leq \sum_{k=1}^m\mathbb{P}\left(|\bar{a}_k^0|>\varepsilon\right)+\sum_{k=1}^m\sum_{\alpha=1}^d\mathbb{P}\left(|(\boldsymbol{w}_k^0)_\alpha|>\varepsilon\right)\\ &\leq 2m\mathrm{e}^{-\frac{1}{2}\varepsilon^2}+2md\mathrm{e}^{-\frac{1}{2}\varepsilon^2}\\ &= 2m(d+1)\mathrm{e}^{-\frac{1}{2}\varepsilon^2}\\ &= \delta,\end{aligned}$$

which implies the conclusions of this lemma. □

Lemma 4.2. *For any $\delta\in(0,1)$ with probability at least $1-\delta$ over the random initialization in* (4.1)*, we have*

$$R_S(\boldsymbol{\theta}^0)\leq\frac{1}{2}\left(1+32\gamma\sqrt{m}Md^3\left(\log\frac{4m(d+1)}{\delta}\right)^2\left(\sqrt{2\log(2d)}+\sqrt{2\log(8/\delta)}\right)\right)^2,$$

Proof. From Lemma 4.1 we know that with probability at least $1-\delta/2$,

$$|\bar{a}_k^0|\leq\sqrt{2\log\frac{4m(d+1)}{\delta}}\quad\text{and}\quad\|\boldsymbol{w}_k^0\|_1\leq d\sqrt{2\log\frac{4m(d+1)}{\delta}}.$$

Let

$$\mathcal{H}=\{h(\bar{a},\boldsymbol{w};\boldsymbol{x})\mid h(\bar{a},\boldsymbol{w};\boldsymbol{x})=\bar{a}\left[\boldsymbol{w}^\intercal\boldsymbol{A}(\boldsymbol{x})\boldsymbol{w}\sigma''(\boldsymbol{w}^\intercal\boldsymbol{x})+\boldsymbol{b}^\intercal(\boldsymbol{x})\boldsymbol{w}\sigma'(\boldsymbol{w}^\intercal\boldsymbol{x})+c(\boldsymbol{x})\sigma(\boldsymbol{w}^\intercal\boldsymbol{x})\right],\boldsymbol{x}\in\Omega\}.$$

Note that $\boldsymbol{A}$, $\boldsymbol{b}$, and c are known functions of $\boldsymbol{x}$. Each element in the above set is a function of $\bar{a}$ and $\boldsymbol{w}$ while $\boldsymbol{x}\in\Omega=[0,1]^d$ is a parameter. Since $\|\boldsymbol{x}\|_\infty\leq 1$, we have

$$\begin{aligned}|h(\bar{a}_k^0,\boldsymbol{w}_k^0;\boldsymbol{x})| &\leq |\bar{a}_k^0|\left[M\|\boldsymbol{w}_k^0\|_1^3+\frac{1}{2}M\|\boldsymbol{w}_k^0\|_1^3+\frac{1}{6}M\|\boldsymbol{w}_k^0\|_1^3\right]\\ &\leq 2M|\bar{a}_k^0|\|\boldsymbol{w}_k^0\|_1^3\\ &\leq 8Md^3\left(\log\frac{4m(d+1)}{\delta}\right)^2.\end{aligned}$$

Then with probability at least $1-\delta/2$, by the Rademacher-based uniform convergence theorem, we have

$$\begin{aligned}\frac{1}{\gamma m}\sup_{\boldsymbol{x}\in\Omega}|f_{\boldsymbol{\theta}^0}(\boldsymbol{x})| &= \sup_{\boldsymbol{x}\in\Omega}\left|\frac{1}{m}\sum_{k=1}^{m}h(\bar{a}_k^0,\boldsymbol{w}_k^0;\boldsymbol{x})-\mathbb{E}_{(\bar{a},\boldsymbol{w})\sim\mathcal{N}(0,\boldsymbol{I}_{d+1})}h(\bar{a},\boldsymbol{w};\boldsymbol{x})\right|\\ &\leq 2\mathrm{Rad}_{\bar{\boldsymbol{\theta}}^0}(\mathcal{H})+24Md^3\left(\log\frac{4m(d+1)}{\delta}\right)^2\sqrt{\frac{2\log(8/\delta)}{m}},\end{aligned}$$

where

$$\begin{aligned}\mathrm{Rad}_{\bar{\boldsymbol{\theta}}^0}(\mathcal{H}) := \frac{1}{m}\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\boldsymbol{x}\in\Omega}\sum_{k=1}^{m}\tau_k h(\bar{a}_k^0,\boldsymbol{w}_k^0;\boldsymbol{x})\right] &\leq I_1+I_2+I_3,\\ I_1 &= \frac{1}{m}\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\boldsymbol{x}\in\Omega}\sum_{k=1}^{m}\tau_k\bar{a}_k^0\boldsymbol{w}_k^{0\intercal}\boldsymbol{A}(\boldsymbol{x})\boldsymbol{w}_k^0\sigma''(\boldsymbol{w}_k^{0\intercal}\boldsymbol{x})\right],\\ I_2 &= \frac{1}{m}\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\boldsymbol{x}\in\Omega}\sum_{k=1}^{m}\tau_k\bar{a}_k^0\boldsymbol{b}^\intercal(\boldsymbol{x})\boldsymbol{w}_k^0\sigma'(\boldsymbol{w}_k^{0\intercal}\boldsymbol{x})\right],\\ I_3 &= \frac{1}{m}\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\boldsymbol{x}\in\Omega}\sum_{k=1}^{m}\tau_k\bar{a}_k^0 c(\boldsymbol{x})\sigma(\boldsymbol{w}_k^{0\intercal}\boldsymbol{x})\right],\end{aligned}$$

where $\boldsymbol{\tau}$ is a random vector in $\mathbb{N}^m$ with i.i.d. entries $\{\tau_k\}_{k=1}^m$ following the Rademacher distribution.

We only prove for I_1. It can be straightforwardly extended to I_2 and I_3.

$$\begin{aligned}I_1 &= \frac{1}{m}\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\boldsymbol{x}\in\Omega}\sum_{k=1}^{m}\tau_k\bar{a}_k^0\boldsymbol{w}_k^{0\intercal}\boldsymbol{A}(\boldsymbol{x})\boldsymbol{w}_k^0\sigma''(\boldsymbol{w}_k^{0\intercal}\boldsymbol{x})\right]\\ &\leq \frac{1}{m}\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\boldsymbol{x},\boldsymbol{y}\in\Omega}\sum_{k=1}^{m}\tau_k\bar{a}_k^0\boldsymbol{w}_k^{0\intercal}\boldsymbol{A}(\boldsymbol{y})\boldsymbol{w}_k^0\sigma''(\boldsymbol{w}_k^{0\intercal}\boldsymbol{x})\right]\\ &= \frac{1}{m}\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\boldsymbol{x},\boldsymbol{y}\in\Omega}\sum_{k=1}^{m}\sum_{\alpha,\beta=1}^{d}\tau_k\bar{a}_k^0(\boldsymbol{w}_k^{0\intercal})_\alpha A_{\alpha\beta}(\boldsymbol{y})(\boldsymbol{w}_k^0)_\beta\sigma''(\boldsymbol{w}_k^{0\intercal}\boldsymbol{x})\right]\\ &\leq \sum_{\alpha,\beta=1}^{d}\frac{1}{m}\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\boldsymbol{x},\boldsymbol{y}\in\Omega}\sum_{k=1}^{m}\tau_k\bar{a}_k^0(\boldsymbol{w}_k^{0\intercal})_\alpha A_{\alpha\beta}(\boldsymbol{y})(\boldsymbol{w}_k^0)_\beta\sigma''(\boldsymbol{w}_k^{0\intercal}\boldsymbol{x})\right].\end{aligned} \tag{4.8}$$

For any $\alpha, \beta \in [d]$, we have

$$
\begin{aligned}
&\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\boldsymbol{x},\boldsymbol{y}\in\Omega}\sum_{k=1}^{m}\tau_k\bar{a}_k^0(\boldsymbol{w}_k^{0\intercal})_\alpha A_{\alpha\beta}(\boldsymbol{y})(\boldsymbol{w}_k^0)_\beta\sigma''(\boldsymbol{w}_k^{0\intercal}\boldsymbol{x})\right] \\
&\leq \mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\boldsymbol{x},\boldsymbol{y}\in\Omega}|A_{\alpha\beta}(\boldsymbol{y})|\left|\sum_{k=1}^{m}\tau_k\bar{a}_k^0(\boldsymbol{w}_k^{0\intercal})_\alpha(\boldsymbol{w}_k^0)_\beta\sigma''(\boldsymbol{w}_k^{0\intercal}\boldsymbol{x})\right|\right] \\
&\leq M\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\boldsymbol{x}\in\Omega}\left|\sum_{k=1}^{m}\tau_k\bar{a}_k^0(\boldsymbol{w}_k^{0\intercal})_\alpha(\boldsymbol{w}_k^0)_\beta\sigma''(\boldsymbol{w}_k^{0\intercal}\boldsymbol{x})\right|\right] \\
&\leq M\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\boldsymbol{x}\in\Omega}\sum_{k=1}^{m}\tau_k\bar{a}_k^0(\boldsymbol{w}_k^{0\intercal})_\alpha(\boldsymbol{w}_k^0)_\beta\sigma''(\boldsymbol{w}_k^{0\intercal}\boldsymbol{x})\right]+M\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\boldsymbol{x}\in\Omega}\sum_{k=1}^{m}-\tau_k\bar{a}_k^0(\boldsymbol{w}_k^{0\intercal})_\alpha(\boldsymbol{w}_k^0)_\beta\sigma''(\boldsymbol{w}_k^{0\intercal}\boldsymbol{x})\right] \\
&= 2M\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\boldsymbol{x}\in\Omega}\sum_{k=1}^{m}\tau_k\bar{a}_k^0(\boldsymbol{w}_k^{0\intercal})_\alpha(\boldsymbol{w}_k^0)_\beta\sigma''(\boldsymbol{w}_k^{0\intercal}\boldsymbol{x})\right], \qquad (4.9)
\end{aligned}
$$

where in the third inequality, we have used the fact that $\sigma''(\boldsymbol{w}_k^{0\intercal}\boldsymbol{x}) = 0$ for $\boldsymbol{x} = 0$ and for any $\boldsymbol{w}_k^0$. Applying Lemma 2.1 with $\psi_k(y_k) = \bar{a}_k(\boldsymbol{w}_k^{0\intercal})_\alpha(\boldsymbol{w}_k^0)_\beta\sigma''(y_k)$ for $k \in [m]$, whose Lipschitz constant is $\left(\sqrt{2\log\frac{4m(d+1)}{\delta}}\right)^3$, we have for all $\alpha, \beta \in [d]$

$$
\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\boldsymbol{x}\in\Omega}\sum_{k=1}^{m}\tau_k\bar{a}_k^0(\boldsymbol{w}_k^{0\intercal})_\alpha(\boldsymbol{w}_k^0)_\beta\sigma''(\boldsymbol{w}_k^{0\intercal}\boldsymbol{x})\right] \leq \left(\sqrt{2\log\frac{4m(d+1)}{\delta}}\right)^3\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\boldsymbol{x}\in\Omega}\sum_{k=1}^{m}\tau_k\boldsymbol{w}_k^{0\intercal}\boldsymbol{x}\right]. \qquad (4.10)
$$

Therefore, combining (4.8), (4.9), and (4.10), we obtain

$$
\begin{aligned}
I_1 &\leq \frac{2Md^2}{m}\left(\sqrt{2\log\frac{4m(d+1)}{\delta}}\right)^3\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\boldsymbol{x}\in\Omega}\sum_{k=1}^{m}\tau_k\boldsymbol{w}_k^{0\intercal}\boldsymbol{x}\right] \\
&\leq \frac{2Md^3}{\sqrt{m}}\left(\sqrt{2\log\frac{4m(d+1)}{\delta}}\right)^4\sqrt{2\log(2d)} \\
&\leq \frac{8Md^3\sqrt{2\log(2d)}}{\sqrt{m}}\left(\log\frac{4m(d+1)}{\delta}\right)^2,
\end{aligned}
$$

where the second inequality is by the Rademacher bound for linear predictors in Lemma 2.2. For I_2 and I_3, we note that $\sigma(z) = \frac{1}{6}z^2\sigma''(z)$ and $\sigma'(z) = \frac{1}{2}z\sigma''(z)$. Then by a similar argument, we have

$$
\begin{aligned}
I_2 &\leq \frac{4Md^2\sqrt{2\log(2d)}}{\sqrt{m}}\left(\log\frac{4m(d+1)}{\delta}\right)^2, \\
I_3 &\leq \frac{4Md\sqrt{2\log(2d)}}{3\sqrt{m}}\left(\log\frac{4m(d+1)}{\delta}\right)^2, \\
\mathrm{Rad}_{\bar{\boldsymbol{\theta}}^0}(\mathcal{H}) &\leq \frac{16Md^3\sqrt{2\log(2d)}}{\sqrt{m}}\left(\log\frac{4m(d+1)}{\delta}\right)^2.
\end{aligned}
$$

So one can get

$$
\begin{aligned}
\sup_{\boldsymbol{x}\in\Omega}|f_{\boldsymbol{\theta}^0}(\boldsymbol{x})| &\leq 32\gamma M d^3\sqrt{m}\sqrt{2\log(2d)}\left(\log\frac{4m(d+1)}{\delta}\right)^2 + 24\gamma\sqrt{m}Md^3\left(\log\frac{4m(d+1)}{\delta}\right)^2\sqrt{2\log(8/\delta)} \\
&\leq 32\gamma\sqrt{m}Md^3\left(\log\frac{4m(d+1)}{\delta}\right)^2\left(\sqrt{2\log(2d)}+\sqrt{2\log(8/\delta)}\right).
\end{aligned}
$$

Then

$$
\begin{aligned}
R_S(\boldsymbol{\theta}^0) &\leq \frac{1}{2n}\sum_{i=1}^{n}(1+|f_{\boldsymbol{\theta}^0}(\boldsymbol{x}_i)|)^2 \\
&\leq \frac{1}{2}\left(1+32\gamma\sqrt{m}Md^3\left(\log\frac{4m(d+1)}{\delta}\right)^2\left(\sqrt{2\log(2d)}+\sqrt{2\log(8/\delta)}\right)\right)^2,
\end{aligned}
$$

where the first inequality comes from the fact that $|f|\leq 1$ by our assumption of the PDE. □

The following lemma shows the positive definiteness of $\boldsymbol{G}^{(a)}$ at initialization.

Lemma 4.3. *For any $\delta\in(0,1)$, if $m\geq\frac{256n^4M^4C_d}{\lambda_S^2\delta}$, then with probability at least $1-\delta$ over the random initialization in* (4.1)*, we have*

$$
\lambda_{\min}\left(\boldsymbol{G}^{(a)}(\boldsymbol{\theta}^0)\right)\geq\frac{3}{4}\lambda_S,
$$

where $C_d:=\mathbb{E}\|\boldsymbol{w}\|_1^{12}<+\infty$ with $\boldsymbol{w}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I}_d)$.

Proof. We define $\Omega_{ij}:=\{\boldsymbol{\theta}^0\mid|\boldsymbol{G}_{ij}^{(a)}(\boldsymbol{\theta}^0)-\boldsymbol{K}_{ij}^{(a)}|\leq\frac{\lambda_S}{4n}\}$. Note that

$$
|g^{(a)}(\boldsymbol{w}_k^0;\boldsymbol{x}_i,\boldsymbol{x}_j)|\leq\left(M\|\boldsymbol{w}_k^0\|_1^3+\frac{1}{2}M\|\boldsymbol{w}_k^0\|_1^3+\frac{1}{6}M\|\boldsymbol{w}_k^0\|_1^3\right)^2\leq 4M^2\|\boldsymbol{w}_k^0\|_1^6.
$$

So

$$
\mathrm{Var}\left(g^{(a)}(\boldsymbol{w}_k^0;\boldsymbol{x}_i,\boldsymbol{x}_j)\right)\leq\mathbb{E}\left(g^{(a)}(\boldsymbol{w}_k^0;\boldsymbol{x}_i,\boldsymbol{x}_j)\right)^2\leq 16M^4\mathbb{E}\|\boldsymbol{w}_k^0\|_1^{12}=16M^4C_d,
$$

and

$$
\mathrm{Var}\left(\boldsymbol{G}_{ij}^{(a)}(\boldsymbol{\theta}^0)\right)=\frac{1}{m^2}\sum_{k=1}^{m}\mathrm{Var}\left(g^{(a)}(\boldsymbol{w}_k^0;\boldsymbol{x}_i,\boldsymbol{x}_j)\right)\leq\frac{16M^4C_d}{m}.
$$

Then the probability of the event Ω_{ij} has the lower bound:

$$
\mathbb{P}(\Omega_{ij})\geq 1-\frac{\mathrm{Var}\left(\boldsymbol{G}_{ij}^{(a)}(\boldsymbol{\theta}^0)\right)}{[\lambda_S/(4n)]^2}\geq 1-\frac{256M^4n^2C_d}{\lambda_S^2 m}.
$$

Thus, with probability at least $\left(1-\frac{256M^4n^2C_d}{\lambda_S^2 m}\right)^{n^2}\geq 1-\frac{256M^4n^4C_d}{\lambda_S^2 m}$, we have all events Ω_{ij} for $i,j\in[n]$ happen. This implies that with probability at least $1-\frac{256M^4n^4C_d}{\lambda_S^2 m}$, we have

$$
\|\boldsymbol{G}^{(a)}(\boldsymbol{\theta}^0)-\boldsymbol{K}^{(a)}\|_{\mathrm{F}}\leq\frac{\lambda_S}{4}
$$

and

$$\lambda_{\min}\left(\boldsymbol{G}^{(a)}(\boldsymbol{\theta}^0)\right) \geq \lambda_S - \|\boldsymbol{G}^{(a)}(\theta^0) - \boldsymbol{K}^{(a)}\|_{\mathrm{F}} \geq \frac{3}{4}\lambda_S.$$

For any $\delta \in (0,1)$, if $m \geq \frac{256n^4M^4C_d}{\lambda_S^2\delta}$, then with probability at least $1 - \frac{256M^4n^4C_d}{\lambda_S^2 m} \geq 1-\delta$ over the initialization $\boldsymbol{\theta}^0$, we have $\lambda_{\min}\left(\boldsymbol{G}^{(a)}(\boldsymbol{\theta}^0)\right) \geq \frac{3}{4}\lambda_S$. □

The following lemma estimates the empirical loss dynamics before the stopping time t^* in (4.5).

Lemma 4.4. *For any $\delta \in (0,1)$, if $m \geq \frac{256n^4M^4C_d}{\lambda_S^2\delta}$, then with probability at least $1-\delta$ over the random initialization in* (4.1)*, we have for any $t \in [0,t^*)$*

$$R_S(\boldsymbol{\theta}(t)) \leq \exp\left(-\frac{m\lambda_S t}{n}\right) R_S(\boldsymbol{\theta}^0).$$

Proof. From Lemma 4.3, for any $\delta \in (0,1)$ with probability at least $1-\delta$ over initialization $\boldsymbol{\theta}^0$ and for any $t \in [0,t^*)$ with t^* defined in (4.5), we have $\boldsymbol{\theta}(t) \in \mathcal{M}(\boldsymbol{\theta}^0)$ defined in (4.6) and

$$\begin{aligned}
\lambda_{\min}\left(\boldsymbol{G}^{(a)}(\boldsymbol{\theta})\right) &\geq \lambda_{\min}\left(\boldsymbol{G}^{(a)}(\boldsymbol{\theta}^0)\right) - \|\boldsymbol{G}^{(a)}(\boldsymbol{\theta}) - \boldsymbol{G}^{(a)}(\boldsymbol{\theta}^0)\|_{\mathrm{F}} \\
&\geq \frac{3}{4}\lambda_S - \frac{1}{4}\lambda_S \\
&= \frac{1}{2}\lambda_S.
\end{aligned}$$

Note that $\boldsymbol{G}_{ij} = \frac{1}{m}\nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) \cdot \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}}(\boldsymbol{x}_j)$ and $\nabla_{\boldsymbol{\theta}} R_S = \frac{1}{n}\sum_{i=1}^n e_i \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$, so

$$\|\nabla_{\boldsymbol{\theta}} R_S(\boldsymbol{\theta}(t))\|_2^2 = \frac{m}{n^2}\boldsymbol{e}^\intercal \boldsymbol{G}(\boldsymbol{\theta}(t))\boldsymbol{e} \geq \frac{m}{n^2}\boldsymbol{e}^\intercal \boldsymbol{G}^{(a)}(\boldsymbol{\theta}(t))\boldsymbol{e},$$

where the last equation is true by the fact that $G^{(w)}(\boldsymbol{\theta}(t))$ is a Gram matrix and hence positive semi-definite. Together with

$$\frac{m}{n^2}\boldsymbol{e}^\intercal \boldsymbol{G}^{(a)}(\boldsymbol{\theta}(t))\boldsymbol{e} \geq \frac{2m}{n}\lambda_{\min}\left(\boldsymbol{G}^{(a)}(\boldsymbol{\theta}(t))\right) R_S(\boldsymbol{\theta}(t)) \geq \frac{m}{n}\lambda_S R_S(\boldsymbol{\theta}(t)),$$

then finally we get

$$\frac{\mathrm{d}}{\mathrm{d}t} R_S(\boldsymbol{\theta}(t)) = -\|\nabla_{\boldsymbol{\theta}} R_S(\boldsymbol{\theta}(t))\|_2^2 \leq -\frac{m}{n}\lambda_S R_S(\boldsymbol{\theta}(t)).$$

Integrating the above equation yields the conclusion in this lemma. □

The following lemma shows that the parameters in the two-layer neural network is uniformly bounded in time during the training before time t^*.

Lemma 4.5. *For any $\delta \in (0,1)$, if $m \geq \max\left\{\frac{512n^4M^4C_d}{\lambda_S^2\delta}, \frac{200\sqrt{2}Md^3n\log(4m(d+1)/\delta)\sqrt{R_S(\boldsymbol{\theta}^0)}}{\lambda_S}\right\}$, then with probability at least $1-\delta$ over the random initialization in* (4.1)*, for any $t \in [0,t^*)$ and any $k \in [m]$,*

$$\begin{aligned}
|a_k(t) - a_k(0)| &\leq q, \quad &\|\boldsymbol{w}_k(t) - \boldsymbol{w}_k(0)\|_\infty &\leq q, \\
|a_k(0)| &\leq \gamma\eta, &\|\boldsymbol{w}_k(0)\|_\infty &\leq \eta,
\end{aligned}$$

where

$$q := \frac{320Md^3(\log\frac{4m(d+1)}{\delta})^{3/2}n\sqrt{R_S(\boldsymbol{\theta}^0)}}{m\lambda_S}$$

and

$$\eta := \sqrt{2\log\frac{4m(d+1)}{\delta}}.$$

Proof. Let $\xi(t) = \max_{k\in[m], s\in[0,t]} \{|a_k(s)|, \|\boldsymbol{w}_k(s)\|_\infty\}$. Note that

$$\begin{aligned}
|\nabla_{a_k} R_S(\boldsymbol{\theta})|^2 &= \left\{ \frac{1}{n}\sum_{i=1}^n e_i \left[\boldsymbol{w}_k^\intercal \boldsymbol{A}(\boldsymbol{x}_i)\boldsymbol{w}_k \sigma''(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i) + \boldsymbol{b}^\intercal(\boldsymbol{x}_i)\boldsymbol{w}_k \sigma'(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i) + c(\boldsymbol{x}_i)\sigma(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i) \right] \right\}^2 \\
&\le 8M^2 \|\boldsymbol{w}_k\|_1^6 R_S(\boldsymbol{\theta}) \\
&\le 8M^2 d^6 (\xi(t))^6 R_S(\boldsymbol{\theta}),
\end{aligned}$$

and

$$\begin{aligned}
\|\nabla_{\boldsymbol{w}_k} R_S(\boldsymbol{\theta})\|_\infty^2 &= \Big\| \frac{1}{n}\sum_{i=1}^n e_i a_k \Big[2\boldsymbol{A}(\boldsymbol{x}_i)\boldsymbol{w}_k \sigma''(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i) + \boldsymbol{w}_k^\intercal \boldsymbol{A}(\boldsymbol{x}_i)\boldsymbol{w}_k \sigma^{(3)}(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i)\boldsymbol{x}_i \\
&\qquad + \sigma'(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i)\boldsymbol{b}(\boldsymbol{x}_i) + \boldsymbol{b}^\intercal(\boldsymbol{x}_i)\boldsymbol{w}_k \sigma''(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i)\boldsymbol{x}_i + c(\boldsymbol{x}_i)\sigma'(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i)\boldsymbol{x}_i \Big] \Big\|_\infty^2 \\
&\le |a_k|^2 2R_S(\boldsymbol{\theta}) \Big(2M\|\boldsymbol{w}_k\|_1^2 + M\|\boldsymbol{w}_k\|_1^2 + \frac{1}{2}M\|\boldsymbol{w}_k\|_1^2 + M\|\boldsymbol{w}_k\|_1^2 + M\frac{1}{2}\|\boldsymbol{w}_k\|_1^2 \Big)^2 \\
&\le 50M^2 \|\boldsymbol{w}_k\|_1^4 |a_k|^2 R_S(\boldsymbol{\theta}) \\
&\le 50M^2 d^4 (\xi(t))^6 R_S(\boldsymbol{\theta}).
\end{aligned}$$

From Lemma 4.4, if $m \ge \frac{512M^4n^4C_d}{\lambda_s^2\delta}$, then with probability at least $1-\delta/2$ over initialization

$$\begin{aligned}
|a_k(t) - a_k(0)| &\le \int_0^t |\nabla_{a_k} R_S(\boldsymbol{\theta}(s))| \, \mathrm{d}s \\
&\le 2\sqrt{2}Md^3 \int_0^t \xi^3(t)\sqrt{R_S(\boldsymbol{\theta}(s))} \, \mathrm{d}s \\
&\le 2\sqrt{2}Md^3 \xi^3(t) \int_0^t \sqrt{R_S(\boldsymbol{\theta}^0)} \exp\left(-\frac{m\lambda_S s}{2n}\right) \mathrm{d}s \\
&\le \frac{4\sqrt{2}Md^3 n\sqrt{R_S(\boldsymbol{\theta}^0)}}{m\lambda_S} \xi^3(t) \\
&\le p\xi^3(t),
\end{aligned}$$

where $p := \frac{10\sqrt{2}d^3Mn\sqrt{R_S(\boldsymbol{\theta}^0)}}{m\lambda_S}$. Similarly,

$$\begin{aligned}
\|\boldsymbol{w}_k(t) - \boldsymbol{w}_k(0)\|_\infty &\le \int_0^t \|\nabla_{\boldsymbol{w}_k} R_S(\boldsymbol{\theta}(s))\|_\infty \, \mathrm{d}s \\
&\le 5\sqrt{2}Md^2 \int_0^t \xi^3(t)\sqrt{R_S(\boldsymbol{\theta}(s))} \, \mathrm{d}s \\
&\le 5\sqrt{2}Md^2 \xi^3(t) \int_0^t \sqrt{R_S(\boldsymbol{\theta}^0)} \exp\left(-\frac{m\lambda_S s}{2n}\right) \mathrm{d}s \\
&\le \frac{10\sqrt{2}Md^2 n\sqrt{R_S(\boldsymbol{\theta}^0)}}{m\lambda_S} \xi^3(t) \\
&\le p\xi^3(t).
\end{aligned}$$

So

$$\xi(t) \le \xi(0) + p\xi^3(t). \tag{4.11}$$

From Lemma 4.1 with probability at least $1-\delta/2$,

$$\begin{aligned}\xi(0) = \max_{k\in[m]}\{|a_k(0)|, \|\boldsymbol{w}_k(0)\|_\infty\} &\leq \max\left\{\gamma\sqrt{2\log\frac{4m(d+1)}{\delta}}, \sqrt{2\log\frac{4m(d+1)}{\delta}}\right\}\\ &\leq \sqrt{2\log\frac{4m(d+1)}{\delta}} = \eta. \end{aligned}\tag{4.12}$$

Since

$$m \geq \frac{200\sqrt{2}Md^3 n\log(4m(d+1)/\delta)\sqrt{R_S(\boldsymbol{\theta}^0)}}{\lambda_S} = 10mp\eta^2,$$

then $p \leq \frac{1}{10}\left(2\log\frac{4m(d+1)}{\delta}\right)^{-1} = \frac{1}{10}\eta^{-2}$ and $p(2\eta)^2 \leq \frac{2}{5}$. Let

$$t_0 := \inf\{t \mid \xi(t) > 2\eta\}.$$

We will prove $t_0 \geq t^*$ by contradiction. Suppose that $t_0 < t^*$. For $t \in [0, t_0)$, by (4.11), (4.12), and $\xi(t) \leq 2\eta$, we have

$$\xi(t) \leq \eta + p(2\eta)^2\xi(t) \leq \eta + \frac{2}{5}\xi(t),$$

then

$$\xi(t) \leq \frac{5}{3}\eta.$$

After letting $t \to t_0$, the inequality just above contradicts with the definition of t_0. So $t_0 \geq t^*$ and then $\xi(t) \leq 2\eta$ for all $t \in [0, t^*)$. Thus

$$\begin{aligned}|a_k(t) - a_k(0)| &\leq 8\eta^3 p\\ \|\boldsymbol{w}_k(t) - \boldsymbol{w}_k(0)\|_\infty &\leq 8\eta^3 p.\end{aligned}$$

Finally, notice that

$$\begin{aligned}8\eta^3 p &= 8\sqrt{8}\left(\log\frac{4m(d+1)}{\delta}\right)^{3/2}\frac{10\sqrt{2}Md^3 n\sqrt{R_S(\boldsymbol{\theta}^0)}}{m\lambda_S}\\ &= \frac{320Md^3\left(\log\frac{4m(d+1)}{\delta}\right)^{3/2} n\sqrt{R_S(\boldsymbol{\theta}^0)}}{m\lambda_S}\\ &= q,\end{aligned}\tag{4.13}$$

which ends the proof. □

4.3 Proof of Theorem 3.1

Proof of Theorem 3.1. From Lemma 4.4, it is sufficient to prove that the stopping time t^* in Lemma 4.4 is equal to $+\infty$. We will prove this by contradiction.

Suppose $t^* < +\infty$. Note that

$$|\boldsymbol{G}_{ij}^{(a)}(\boldsymbol{\theta}(t^*)) - \boldsymbol{G}_{ij}^{(a)}(\boldsymbol{\theta}(0))| \leq \frac{1}{m}\sum_{k=1}^{m}|g^{(a)}(\boldsymbol{w}_k(t^*); \boldsymbol{x}_i, \boldsymbol{x}_j) - g^{(a)}(\boldsymbol{w}_k(0); \boldsymbol{x}_i, \boldsymbol{x}_j)|. \tag{4.14}$$

By the mean value theorem,

$$\begin{aligned} &|g^{(a)}(\boldsymbol{w}_k(t^*);\boldsymbol{x}_i,\boldsymbol{x}_j) - g^{(a)}(\boldsymbol{w}_k(0);\boldsymbol{x}_i,\boldsymbol{x}_j)| \\ &\leq \|\nabla g^{(a)}\left(c\boldsymbol{w}_k(t^*) + (1-c)\boldsymbol{w}_k(0);\boldsymbol{x}_i,\boldsymbol{x}_j\right)\|_\infty \|\boldsymbol{w}_k(t^*) - \boldsymbol{w}_k(0)\|_1 \end{aligned}$$

for some $c \in (0,1)$. Further computation yields

$$\begin{aligned} \nabla g^{(a)}(\boldsymbol{w};\boldsymbol{x}_i,\boldsymbol{x}_j) =& \Big[2\boldsymbol{A}(\boldsymbol{x}_i)\boldsymbol{w}\sigma''(\boldsymbol{w}^\intercal\boldsymbol{x}_i) + \boldsymbol{w}^\intercal\boldsymbol{A}(\boldsymbol{x}_i)\boldsymbol{w}\sigma^{(3)}(\boldsymbol{w}^\intercal\boldsymbol{x}_i)\boldsymbol{x}_i + \sigma'(\boldsymbol{w}^\intercal\boldsymbol{x}_i)\boldsymbol{b}(\boldsymbol{x}_i) \\ &\quad + \boldsymbol{b}^\intercal(\boldsymbol{x}_i)\boldsymbol{w}\sigma''(\boldsymbol{w}^\intercal\boldsymbol{x}_i)\boldsymbol{x}_i + c(\boldsymbol{x}_i)\sigma'(\boldsymbol{w}^\intercal\boldsymbol{x}_i)\boldsymbol{x}_i\Big] \\ &\quad \times \Big[\boldsymbol{w}^\intercal\boldsymbol{A}(\boldsymbol{x}_j)\boldsymbol{w}\sigma''(\boldsymbol{w}^\intercal\boldsymbol{x}_j) + \boldsymbol{b}^\intercal(\boldsymbol{x}_j)\boldsymbol{w}\sigma'(\boldsymbol{w}^\intercal\boldsymbol{x}_j) + c(\boldsymbol{x}_j)\sigma(\boldsymbol{w}^\intercal\boldsymbol{x}_j)\Big] \\ &+ \Big[2\boldsymbol{A}(\boldsymbol{x}_j)\boldsymbol{w}\sigma''(\boldsymbol{w}^\intercal\boldsymbol{x}_j) + \boldsymbol{w}^\intercal\boldsymbol{A}(\boldsymbol{x}_j)\boldsymbol{w}\sigma^{(3)}(\boldsymbol{w}^\intercal\boldsymbol{x}_j)\boldsymbol{x}_i + \sigma'(\boldsymbol{w}^\intercal\boldsymbol{x}_i)\boldsymbol{b}(\boldsymbol{x}_i) \\ &\quad + \boldsymbol{b}^\intercal(\boldsymbol{x}_j)\boldsymbol{w}\sigma''(\boldsymbol{w}^\intercal\boldsymbol{x}_j)\boldsymbol{x}_j + c(\boldsymbol{x}_j)\sigma'(\boldsymbol{w}^\intercal\boldsymbol{x}_j)\boldsymbol{x}_j\Big] \\ &\quad \times \Big[\boldsymbol{w}^\intercal\boldsymbol{A}(\boldsymbol{x}_i)\boldsymbol{w}\sigma''(\boldsymbol{w}^\intercal\boldsymbol{x}_i) + \boldsymbol{b}^\intercal(\boldsymbol{x}_i)\boldsymbol{w}\sigma'(\boldsymbol{w}^\intercal\boldsymbol{x}_i) + c(\boldsymbol{x}_i)\sigma(\boldsymbol{w}^\intercal\boldsymbol{x}_i)\Big] \end{aligned}$$

for all $\boldsymbol{w}$. Hence, it holds for all $\boldsymbol{w}$ that

$$\begin{aligned} \|\nabla g^{(a)}(\boldsymbol{w};\boldsymbol{x}_i,\boldsymbol{x}_j)\|_\infty &\leq 2\Big[2M\|\boldsymbol{w}\|_1^2 + M\|\boldsymbol{w}\|_1^2 + \frac{1}{2}M\|\boldsymbol{w}\|_1^2 + M\|\boldsymbol{w}\|_1^2 + \frac{1}{2}M\|\boldsymbol{w}\|_1^2\Big] \\ &\quad \times \Big[M\|\boldsymbol{w}\|_1^3 + \frac{1}{2}M\|\boldsymbol{w}\|_1^3 + \frac{1}{6}M\|\boldsymbol{w}\|_1^3\Big] \\ &\leq 2(5M\|\boldsymbol{w}\|_1^2)(2M\|\boldsymbol{w}\|_1^3) \\ &= 20M^2\|\boldsymbol{w}\|_1^5. \end{aligned}$$

Therefore, the bound in (4.14) becomes

$$|\boldsymbol{G}_{ij}^{(a)}(\boldsymbol{\theta}(t^*)) - \boldsymbol{G}_{ij}^{(a)}(\boldsymbol{\theta}(0))| \leq \frac{20M^2}{m}\sum_{k=1}^m \|c\boldsymbol{w}_k(t^*) + (1-c)\boldsymbol{w}_k(0)\|_1^5 \|\boldsymbol{w}_k(t^*) - \boldsymbol{w}_k(0)\|_1. \tag{4.15}$$

By Lemma 4.5,

$$\|c\boldsymbol{w}_k(t^*) + (1-c)\boldsymbol{w}_k(0)\|_1 \leq \|\boldsymbol{w}_k(0)\|_1 + \|\boldsymbol{w}_k(t^*) - \boldsymbol{w}_k(0)\|_1 \leq d(\eta + q) \leq 2d\eta,$$

where η and q are defined in Lemma 4.5. So, (4.15) and the above inequalities indicate

$$|\boldsymbol{G}_{ij}^{(a)}(\boldsymbol{\theta}(t^*)) - \boldsymbol{G}_{ij}^{(a)}(\boldsymbol{\theta}(0))| \leq 20M^2(2d\eta)^5 dq = 640M^2d^6\eta^5 q,$$

and

$$\begin{aligned} \|\boldsymbol{G}^{(a)}(\boldsymbol{\theta}(t^*)) - \boldsymbol{G}^{(a)}(\boldsymbol{\theta}(0))\|_{\mathrm{F}} &\leq 640M^2d^6n\eta^5 q \\ &< \frac{2^{21}M^3d^9n^2(\log\frac{4m(d+1)}{\delta})^4\sqrt{R_S(\boldsymbol{\theta}^0)}}{m\lambda_S} \\ &\leq \frac{1}{4}\lambda_S, \end{aligned}$$

if we choose

$$m \geq \frac{2^{23}M^3d^9n^2(\log(4m(d+1)/\delta))^4\sqrt{R_S(\boldsymbol{\theta}^0)}}{\lambda_S^2}.$$

The fact that $\|\boldsymbol{G}^{(a)}(\boldsymbol{\theta}(t^*)) - \boldsymbol{G}^{(a)}(\boldsymbol{\theta}(0))\|_{\mathrm{F}} \leq \frac{1}{4}\lambda_S$ above contradicts with the definition of t^* in (4.5). Hence, we have completed the proof. □

5 A priori Estimates of Generalization Error for Two-layer Neural Networks

To obtain good generalization, instead of minimizing R_S, we minimize the regularized risk of $R_S(\boldsymbol{\theta})$:

$$J_{S,\lambda}(\boldsymbol{\theta}) := R_S(\boldsymbol{\theta}) + \frac{\lambda}{\sqrt{n}}\|\boldsymbol{\theta}\|_{\mathcal{P}}^3 \tag{5.1}$$

to obtain

$$\boldsymbol{\theta}_{S,\lambda} = \arg\min_{\boldsymbol{\theta}} J_{S,\lambda}(\boldsymbol{\theta}). \tag{5.2}$$

Our work is inspired by the seminal work in [14, 13] and the proof is a variant of the proof therein. But as we shall see, the differential operator increases the technical difficulty in the analysis: extra non-linearity in the parameters, which makes existing mean field analysis [33] not applicable. We will use the path norm defined in Definition 2.2 adaptive to the PDE problem, instead of using the path norm in [14, 13] for regression problems. We will show that the PDE solution network $\phi(\boldsymbol{x}; \boldsymbol{\theta}_{S,\lambda})$ generalize well if the true solution is in the Barron-type space defined in Definition 2.3, which is also a variance of the Barron-type space in [14, 13]. The generalization error is measured in terms of how well $f(\boldsymbol{x}; \boldsymbol{\theta}_{S,\lambda}) := \mathcal{L}\phi(\boldsymbol{x}; \boldsymbol{\theta}_{S,\lambda}) \approx f(\boldsymbol{x})$ generalizes from the random training samples $S = \{\boldsymbol{x}_i\}_{i=1}^n \subset \Omega$ to arbitrary samples in Ω.

Recall that $f(\boldsymbol{x}; \boldsymbol{\theta})$, also denoted as $f_{\boldsymbol{\theta}}(\boldsymbol{x})$, is the result of the differential operator $\mathcal{L}$ acting on a two-layer neural network $\phi(\boldsymbol{x}; \boldsymbol{\theta})$ in the domain Ω. In fact, $f(\boldsymbol{x}; \boldsymbol{\theta})$ is also a two-layer neural network as explained in (4.2). Hence, the generalization error analysis of deep learning-based PDE solvers is reduced to the generalization analysis of the special two-layer neural network $f(\boldsymbol{x}; \boldsymbol{\theta})$ fitting $f(\boldsymbol{x})$. The special structure of $f(\boldsymbol{x}; \boldsymbol{\theta})$ leads to significant difficulty in analyzing the generalization error compared to traditional two-layer neural networks in the literature.

We will first summarize and prove several lemmas related to Rademacher complexity in Section 5.1. The proofs of our main theorems for the generalization bound in Theorems 3.2 and 3.3 are presented in Section 5.2.

5.1 Preliminary Lemmas of Rademacher Complexity

First, we define the set of functions

$$\mathcal{F}_Q = \{f(\boldsymbol{x}; \boldsymbol{\theta}) = \sum_{k=1}^m a_k[\boldsymbol{w}_k^\intercal \boldsymbol{A}(\boldsymbol{x})\boldsymbol{w}_k \sigma''(\boldsymbol{w}_k^\intercal \boldsymbol{x}) + \boldsymbol{b}^\intercal(\boldsymbol{x})\boldsymbol{w}_k \sigma'(\boldsymbol{w}_k^\intercal \boldsymbol{x}) + c(\boldsymbol{x})\sigma(\boldsymbol{w}_k^\intercal \boldsymbol{x})] \mid \|\boldsymbol{\theta}\|_{\mathcal{P}} \leq Q\}.$$

Second, we estimate the Rademacher complexity of the class of special two-layer neural networks $\mathcal{F}_Q$.

Lemma 5.1 (Rademacher complexity of two-layer neural networks)**.** *The Rademacher complexity of $\mathcal{F}_Q$ over a set of n uniform distributed random samples of Ω, denoted as $S = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, has an upper bound*

$$\mathrm{Rad}_S(\mathcal{F}_Q) \leq \frac{4MQd^2\sqrt{2\log(2d)}}{\sqrt{n}},$$

where M is the upper bound of the differential operator $\mathcal{L}$ introduced in (3.1)*.*

Proof. Let $\hat{\boldsymbol{w}}_k = \boldsymbol{w}_k/\|\boldsymbol{w}_k\|_1$ for $k = 1, \cdots, m$ and $\boldsymbol{\tau}$ be a random vector in $\mathbb{N}^d$ with i.i.d. entries following the Rademacher distribution. Then

$$\begin{aligned}
&n\mathrm{Rad}_S(\mathcal{F}_Q) \\
&= \mathbb{E}_{\boldsymbol{\tau}}\left\{\sup_{\|\boldsymbol{\theta}\|_{\mathcal{P}}\leq Q}\sum_{i=1}^n \tau_i \sum_{k=1}^m a_k[\boldsymbol{w}_k^\intercal \boldsymbol{A}(\boldsymbol{x}_i)\boldsymbol{w}_k\sigma''(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i) + \boldsymbol{b}^\intercal(\boldsymbol{x}_i)\boldsymbol{w}_k\sigma'(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i) + c(\boldsymbol{x}_i)\sigma(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i)]\right\} \\
&\leq \mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{\theta}\|_{\mathcal{P}}\leq Q}\sum_{i=1}^n \tau_i \sum_{k=1}^m a_k\boldsymbol{w}_k^\intercal \boldsymbol{A}(\boldsymbol{x}_i)\boldsymbol{w}_k\sigma''(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i)\right] + \mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{\theta}\|_{\mathcal{P}}\leq Q}\sum_{i=1}^n \tau_i \sum_{k=1}^m a_k\boldsymbol{b}^\intercal(\boldsymbol{x}_i)\boldsymbol{w}_k\sigma'(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i)\right] \\
&\quad + \mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{\theta}\|_{\mathcal{P}}\leq Q}\sum_{i=1}^n \tau_i \sum_{k=1}^m a_k c(\boldsymbol{x}_i)\sigma(\boldsymbol{w}_k^\intercal \boldsymbol{x}_i)\right] \\
&=: I_1 + I_2 + I_3.
\end{aligned} \tag{5.3}$$

We first estimate I_1 as follows

$$\begin{aligned}
I_1 &= \mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{\theta}\|_{\mathcal{P}}\leq Q}\sum_{i=1}^n \tau_i \sum_{k=1}^m a_k\|\boldsymbol{w}_k\|_1^3\hat{\boldsymbol{w}}_k^\intercal \boldsymbol{A}(\boldsymbol{x}_i)\hat{\boldsymbol{w}}_k\sigma''(\hat{\boldsymbol{w}}_k^\intercal \boldsymbol{x}_i)\right] \\
&\leq \mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{\theta}\|_{\mathcal{P}}\leq Q, \|\boldsymbol{u}_k\|_1=1, \forall k}\sum_{i=1}^n \tau_i \sum_{k=1}^m a_k\|\boldsymbol{w}_k\|_1^3\boldsymbol{u}_k^\intercal \boldsymbol{A}(\boldsymbol{x}_i)\boldsymbol{u}_k\sigma''(\boldsymbol{u}_k^\intercal \boldsymbol{x}_i)\right] \\
&\leq \mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{\theta}\|_{\mathcal{P}}\leq Q, \|\boldsymbol{u}_k\|_1=1, \forall k}\sum_{k=1}^m \left|a_k\|\boldsymbol{w}_k\|_1^3\right|\left|\sum_{i=1}^n \tau_i\boldsymbol{u}_k^\intercal \boldsymbol{A}(\boldsymbol{x}_i)\boldsymbol{u}_k\sigma''(\boldsymbol{u}_k^\intercal \boldsymbol{x}_i)\right|\right] \\
&= \mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{\theta}\|_{\mathcal{P}}\leq Q, \|\boldsymbol{u}\|_1=1}\sum_{k=1}^m |a_k|\|\boldsymbol{w}_k\|_1^3\left|\sum_{i=1}^n \tau_i\boldsymbol{u}^\intercal \boldsymbol{A}(\boldsymbol{x}_i)\boldsymbol{u}\sigma''(\boldsymbol{u}^\intercal \boldsymbol{x}_i)\right|\right] \\
&\leq Q\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{u}\|_1\leq 1, \|\boldsymbol{p}\|_1\leq 1, \|\boldsymbol{q}\|_1\leq 1}\left|\sum_{i=1}^n \tau_i\boldsymbol{p}^\intercal \boldsymbol{A}(\boldsymbol{x}_i)\boldsymbol{q}\sigma''(\boldsymbol{u}^\intercal \boldsymbol{x}_i)\right|\right] \\
&= Q\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{u}\|_1\leq 1, \|\boldsymbol{p}\|_1\leq 1, \|\boldsymbol{q}\|_1\leq 1}\left|\boldsymbol{p}^\intercal\left(\sum_{i=1}^n \tau_i\boldsymbol{A}(\boldsymbol{x}_i)\sigma''(\boldsymbol{u}^\intercal \boldsymbol{x}_i)\right)\boldsymbol{q}\right|\right] \\
&= Q\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{u}\|_1\leq 1, \|\boldsymbol{p}\|_1\leq 1, \|\boldsymbol{q}\|_1\leq 1}\sum_{\alpha,\beta=1}^d |p_\alpha||q_\beta|\left|\sum_{i=1}^n \tau_i A_{\alpha\beta}(\boldsymbol{x}_i)\sigma''(\boldsymbol{u}^\intercal \boldsymbol{x}_i)\right|\right] \\
&\leq Q\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{u}\|_1\leq 1}\max_{\alpha,\beta\in[d]}\left|\sum_{i=1}^n \tau_i A_{\alpha\beta}(\boldsymbol{x}_i)\sigma''(\boldsymbol{u}^\intercal \boldsymbol{x}_i)\right|\right] \\
&\leq Q\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{u}\|_1\leq 1}\sum_{\alpha,\beta=1}^d\left|\sum_{i=1}^n \tau_i A_{\alpha\beta}(\boldsymbol{x}_i)\sigma''(\boldsymbol{u}^\intercal \boldsymbol{x}_i)\right|\right] \\
&\leq Q\mathbb{E}_{\boldsymbol{\tau}}\left[\sum_{\alpha,\beta=1}^d\sup_{\|\boldsymbol{u}\|_1\leq 1}\left|\sum_{i=1}^n \tau_i A_{\alpha\beta}(\boldsymbol{x}_i)\sigma''(\boldsymbol{u}^\intercal \boldsymbol{x}_i)\right|\right] \\
&= Q\sum_{\alpha,\beta=1}^d\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{u}\|_1\leq 1}\left|\sum_{i=1}^n \tau_i A_{\alpha\beta}(\boldsymbol{x}_i)\sigma''(\boldsymbol{u}^\intercal \boldsymbol{x}_i)\right|\right].
\end{aligned} \tag{5.4}$$

Note that $\sigma''(\boldsymbol{u}^\intercal \boldsymbol{x}_i) = 0$ for $\boldsymbol{u} = 0$ and for any $\boldsymbol{x}_i$. For any $\alpha, \beta \in [d]$, we have

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{u}\|_1 \le 1}\left|\sum_{i=1}^n \tau_i A_{\alpha\beta}(\boldsymbol{x}_i)\sigma''(\boldsymbol{u}^\intercal \boldsymbol{x}_i)\right|\right] &\le \mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{u}\|_1 \le 1}\sum_{i=1}^n \tau_i A_{\alpha\beta}(\boldsymbol{x}_i)\sigma''(\boldsymbol{u}^\intercal \boldsymbol{x}_i)\right] \\
&\quad + \mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{u}\|_1 \le 1}\sum_{i=1}^n -\tau_i A_{\alpha\beta}(\boldsymbol{x}_i)\sigma''(\boldsymbol{u}^\intercal \boldsymbol{x}_i)\right] \\
&= 2\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{u}\|_1 \le 1}\sum_{i=1}^n \tau_i A_{\alpha\beta}(\boldsymbol{x}_i)\sigma''(\boldsymbol{u}^\intercal \boldsymbol{x}_i)\right]. \qquad (5.5)
\end{aligned}$$

Applying Lemma 2.1 with $\psi_i(y_i) = A_{\alpha\beta}(\boldsymbol{x}_i)\sigma''(y_i)$ for $i \in [n]$, whose Lipschitz constant is M, we have for all $\alpha, \beta \in [d]$

$$\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{u}\|_1 \le 1}\sum_{i=1}^n \tau_i A_{\alpha\beta}(\boldsymbol{x}_i)\sigma''(\boldsymbol{u}^\intercal \boldsymbol{x}_i)\right] \le M\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{u}\|_1 \le 1}\sum_{i=1}^n \tau_i \boldsymbol{u}^\intercal \boldsymbol{x}_i\right]. \qquad (5.6)$$

Therefore, combining (5.4), (5.5), and (5.6), we obtain

$$\begin{aligned}
I_1 &\le 2MQd^2\mathbb{E}_{\boldsymbol{\tau}}\left[\sup_{\|\boldsymbol{u}\|_1 \le 1}\sum_{i=1}^n \tau_i \boldsymbol{u}^\intercal \boldsymbol{x}_i\right] \\
&\le 2MQd^2\sqrt{n}\sqrt{2\log(2d)},
\end{aligned}$$

where the last inequality comes from the Rademacher bound for linear predictors in Lemma 2.2.

For I_2 and I_3, we note that $\sigma(z) = \frac{1}{6}z^2\sigma''(z)$ and $\sigma'(z) = \frac{1}{2}z\sigma''(z)$. Then by similar arguments, we have

$$\begin{aligned}
I_2 &\le MQd\sqrt{n}\sqrt{2\log(2d)}, \\
I_3 &\le \frac{1}{3}MQ\sqrt{n}\sqrt{2\log(2d)}.
\end{aligned}$$

These estimates for I_1, I_2, I_3 combined with (5.3) complete the proof. □

5.2 Proofs of Generalization Bounds

In the proofs of this section, we will first show in Proposition 5.1 that two-layer neural networks $f(\boldsymbol{x};\boldsymbol{\theta})$ in (4.2) can approximate Barron-type functions with an approximation error $O\left(\frac{\|f\|_{\mathcal{B}}^2}{m}\right)$. Second, for an arbitrary $f(\boldsymbol{x};\boldsymbol{\theta}) = \mathcal{L}\phi(\boldsymbol{x};\boldsymbol{\theta})$, we show its a posteriori generalization bound $|R_{\mathcal{D}}(\boldsymbol{\theta}) - R_S(\boldsymbol{\theta})| \le O\left(\frac{\|\boldsymbol{\theta}\|_{\mathcal{P}}^2 \log\|\boldsymbol{\theta}\|_{\mathcal{P}}}{\sqrt{n}}\right)$ in Theorem 3.2. Finally, the a priori generalization bound $R_{\mathcal{D}}(\boldsymbol{\theta}_{S,\lambda}) \le O\left(\frac{\|f\|_{\mathcal{B}}^2}{m} + \frac{\|f\|_{\mathcal{B}}^2 \log\|f\|_{\mathcal{B}}}{\sqrt{n}}\right)$ is proved in Theorem 3.3, where the first and second terms comes from the approximation error bound and the a posteriori generalization bound.

First, the approximation capacity of two-layer neural networks $f(\boldsymbol{x};\boldsymbol{\theta})$ can be characterized by Proposition 5.1 below.

Proposition 5.1 (Approximation Error)**.** *For any $f \in \mathcal{B}(\Omega)$, there exists a two-layer neural network $f(\boldsymbol{x};\tilde{\boldsymbol{\theta}})$ of width m with $\|\tilde{\boldsymbol{\theta}}\|_{\mathcal{P}} \le 2\|f\|_{\mathcal{B}}$,*

$$R_{\mathcal{D}}(\tilde{\boldsymbol{\theta}}) := \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\tfrac{1}{2}(f(\boldsymbol{x},\tilde{\boldsymbol{\theta}}) - f(\boldsymbol{x}))^2 \le \frac{6M^2\|f\|_{\mathcal{B}}^2}{m},$$

where M introduced in (3.1) *controls the upper bound of the differential operator and m is the width of the neural network.*

Proof. Without loss of generality, let ρ be the best representation, i.e., $\|f\|_{\mathcal{B}}^2 = \mathbb{E}_{(a,\boldsymbol{w})\sim\rho}|a|^2\|\boldsymbol{w}\|_1^6$. We set $\bar{\boldsymbol{\theta}} = \{\frac{1}{m}a_k, \boldsymbol{w}_k\}_{k=1}^m$, where $(a_k, \boldsymbol{w}_k)$, $k = 1, \cdots, m$ are independent sampled from ρ. Let

$$f_{\bar{\boldsymbol{\theta}}}(\boldsymbol{x}) = \frac{1}{m}\sum_{k=1}^m a_k[\boldsymbol{w}_k^\intercal \boldsymbol{A}(\boldsymbol{x})\boldsymbol{w}_k\sigma''(\boldsymbol{w}_k^\intercal\boldsymbol{x}) + \boldsymbol{b}^\intercal(\boldsymbol{x})\boldsymbol{w}_k\sigma'(\boldsymbol{w}_k^\intercal\boldsymbol{x}) + c(\boldsymbol{x})\sigma(\boldsymbol{w}_k^\intercal\boldsymbol{x})].$$

Recall the definition $R_{\mathcal{D}}(\bar{\boldsymbol{\theta}}) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\frac{1}{2}|f_{\bar{\boldsymbol{\theta}}}(\boldsymbol{x}) - f(\boldsymbol{x})|^2$. Then

$$\begin{aligned}
&2\mathbb{E}_{\bar{\boldsymbol{\theta}}}R_{\mathcal{D}}(\bar{\boldsymbol{\theta}}) \\
&= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\mathbb{E}_{\bar{\boldsymbol{\theta}}}|f_{\bar{\boldsymbol{\theta}}}(\boldsymbol{x}) - f(\boldsymbol{x})|^2 \\
&= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\mathrm{Var}_{\{(a_k,\boldsymbol{w}_k)\}\text{i.i.d.}\sim\rho}\left(\frac{1}{m}\sum_{k=1}^m a_k[\boldsymbol{w}_k^\intercal \boldsymbol{A}(\boldsymbol{x})\boldsymbol{w}_k\sigma''(\boldsymbol{w}_k^\intercal\boldsymbol{x}) + \boldsymbol{b}^\intercal(\boldsymbol{x})\boldsymbol{w}_k\sigma'(\boldsymbol{w}_k^\intercal\boldsymbol{x}) + c(\boldsymbol{x})\sigma(\boldsymbol{w}_k^\intercal\boldsymbol{x})]\right) \\
&= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\frac{1}{m}\mathrm{Var}_{(a,\boldsymbol{w})\sim\rho}\left(a[\boldsymbol{w}^\intercal \boldsymbol{A}(\boldsymbol{x})\boldsymbol{w}\sigma''(\boldsymbol{w}^\intercal\boldsymbol{x}) + \boldsymbol{b}^\intercal(\boldsymbol{x})\boldsymbol{w}\sigma'(\boldsymbol{w}^\intercal\boldsymbol{x}) + c(\boldsymbol{x})\sigma(\boldsymbol{w}^\intercal\boldsymbol{x})]\right) \\
&\le \frac{1}{m}\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\mathbb{E}_{(a,\boldsymbol{w})\sim\rho}\left(a[\boldsymbol{w}^\intercal \boldsymbol{A}(\boldsymbol{x})\boldsymbol{w}\sigma''(\boldsymbol{w}^\intercal\boldsymbol{x}) + \boldsymbol{b}^\intercal(\boldsymbol{x})\boldsymbol{w}\sigma'(\boldsymbol{w}^\intercal\boldsymbol{x}) + c(\boldsymbol{x})\sigma(\boldsymbol{w}^\intercal\boldsymbol{x})]\right)^2 \\
&\le \frac{1}{m}\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\mathbb{E}_{(a,\boldsymbol{w})\sim\rho}|a|^2\left(M\|\boldsymbol{w}\|_1^3 + \tfrac{1}{2}M\|\boldsymbol{w}\|_1^3 + \tfrac{1}{6}M\|\boldsymbol{w}\|_1^3\right)^2 \\
&\le \frac{4M^2}{m}\mathbb{E}_{(a,\boldsymbol{w})\sim\rho}|a|^2\|\boldsymbol{w}\|_1^6 \\
&= \frac{4M^2\|f\|_{\mathcal{B}}^2}{m}.
\end{aligned}$$

Also, we have

$$\begin{aligned}
\mathbb{E}_{\bar{\boldsymbol{\theta}}}\|\bar{\boldsymbol{\theta}}\|_{\mathcal{P}} &= \mathbb{E}_{\{(a_k,\boldsymbol{w}_k)\}\text{i.i.d.}\sim\rho}\frac{1}{m}\sum_{k=1}^m|a_k|\|\boldsymbol{w}_k\|_1^3 \\
&= \mathbb{E}_{(a,\boldsymbol{w})\sim\rho}|a|\|\boldsymbol{w}\|_1^3 \\
&\le \|f\|_{\mathcal{B}}.
\end{aligned}$$

Define two events $E_1 := \{R_{\mathcal{D}}(\bar{\boldsymbol{\theta}}) < \frac{6M^2\|f\|_{\mathcal{B}}^2}{m}\}$ and $E_2 := \{\|\bar{\boldsymbol{\theta}}\|_{\mathcal{P}} < 2\|f\|_{\mathcal{B}}\}$. By Markov inequality, we have

$$\begin{aligned}
\mathbb{P}(E_1) &= 1 - \mathbb{P}\left(R_{\mathcal{D}}(\bar{\boldsymbol{\theta}}) \ge \frac{6M^2\|f\|_{\mathcal{B}}^2}{m}\right) \ge 1 - \frac{\mathbb{E}_{\bar{\boldsymbol{\theta}}}R_{\mathcal{D}}(\bar{\boldsymbol{\theta}})}{6M^2\|f\|_{\mathcal{B}}^2/m} \ge \frac{2}{3}, \\
\mathbb{P}(E_2) &= 1 - \mathbb{P}(\|\bar{\boldsymbol{\theta}}\|_{\mathcal{P}} \ge 2\|f\|_{\mathcal{B}}) \ge 1 - \frac{\mathbb{E}_{\bar{\boldsymbol{\theta}}}\|\bar{\boldsymbol{\theta}}\|_{\mathcal{P}}}{2\|f\|_{\mathcal{B}}} \ge \frac{1}{2}.
\end{aligned}$$

Thus

$$\mathbb{P}(E_1 \cap E_2) \ge \mathbb{P}(E_1) + \mathbb{P}(E_2) - 1 \ge \frac{2}{3} + \frac{1}{2} - 1 > 0.$$

□

Second, we use Theorem 2.1 with $\mathcal{F} = \mathcal{H}_Q := \{\ell(f(\boldsymbol{x}), f_{\boldsymbol{\theta}}(\boldsymbol{x})) \mid \|\boldsymbol{\theta}\|_{\mathcal{P}} \le Q\}$ and $\mathcal{Z} = \Omega$ to show the a posteriori generalization bound in Theorem 3.2.

Proof of Theorem 3.2. Let $\mathcal{H}_Q := \{\ell(f(\boldsymbol{x}), f_{\boldsymbol{\theta}}(\boldsymbol{x})) \mid \|\boldsymbol{\theta}\|_{\mathcal{P}} \le Q\}$, then $\mathcal{H} = \cup_{Q=1}^{\infty}\mathcal{H}_Q$. Note that

$$\begin{aligned}
\sup_{\boldsymbol{x}\in\Omega}|f_{\boldsymbol{\theta}}(\boldsymbol{x})| &= \sup_{\boldsymbol{x}\in\Omega}\left|\sum_{k=1}^{m} a_k[\boldsymbol{w}_k^{\intercal}\boldsymbol{A}(\boldsymbol{x})\boldsymbol{w}_k\sigma''(\boldsymbol{w}_k^{\intercal}\boldsymbol{x}) + \boldsymbol{b}^{\intercal}(\boldsymbol{x})\boldsymbol{w}_k\sigma'(\boldsymbol{w}_k^{\intercal}\boldsymbol{x}) + c(\boldsymbol{x})\sigma(\boldsymbol{w}_k^{\intercal}\boldsymbol{x})]\right| \\
&\le \sum_{k=1}^{m}|a_k|\|\boldsymbol{w}_k\|_1^3\left[M + \frac{1}{2}M + \frac{1}{6}M\right] \\
&\le \frac{5}{3}M\|\boldsymbol{\theta}\|_{\mathcal{P}}.
\end{aligned}$$

Therefore, for functions in $\mathcal{H}_Q$, since $|f(x)| \le 1$ by assumption, we have

$$\begin{aligned}
0 \le \ell(f(\boldsymbol{x}), f_{\boldsymbol{\theta}}(\boldsymbol{x})) &\le \frac{1}{2}(1 + |f_{\boldsymbol{\theta}}(\boldsymbol{x})|)^2 \\
&\le \frac{1}{2}\left(1 + \frac{5}{3}M\|\boldsymbol{\theta}\|_{\mathcal{P}}\right)^2 \\
&\le \frac{32}{9}M^2Q^2 \le 4M^2Q^2
\end{aligned}$$

for all $\boldsymbol{x} \in \Omega$ and all $Q \ge 1$. For $\|\boldsymbol{\theta}\|_{\mathcal{P}} \le Q$, we note that $\ell(y, \cdot)$ is a Lipschitz function with a Lipschitz constant which is no larger than $\sup_{\boldsymbol{x}\in\Omega}|f_{\boldsymbol{\theta}}(\boldsymbol{x})| \le \frac{5}{3}M\|\boldsymbol{\theta}\|_{\mathcal{P}} + 1$. Let S' be an arbitrary set of n samples of Ω, then

$$\mathrm{Rad}_{S'}(\mathcal{H}_Q) \le (\frac{5}{3}M\|\boldsymbol{\theta}\|_{\mathcal{P}} + 1)\mathrm{Rad}_{S'}(\mathcal{F}_Q) \le (\frac{5}{3}MQ + 1)\mathrm{Rad}_{S'}(\mathcal{F}_Q).$$

Let us assume $MQ \ge \frac{3}{5}$ without loss of generality. By Lemma 5.1 and Theorem 2.1, for any δ given in Theorem 3.2 and any positive integer Q with probability at least $1 - \delta_Q$ over S with $\delta_Q = \frac{6\delta}{\pi^2Q^2}$, we have

$$\begin{aligned}
\sup_{\|\boldsymbol{\theta}\|_{\mathcal{P}}\le Q}|R_{\mathcal{D}}(\boldsymbol{\theta}) - R_S(\boldsymbol{\theta})| &\le (\frac{5}{3}MQ + 1)2\mathbb{E}_{S'}\mathrm{Rad}_{S'}(\mathcal{F}_Q) + 4M^2Q^2\sqrt{\frac{\log(2/\delta_Q)}{2n}} \\
&\le 27M^2Q^2d^2\sqrt{\frac{2\log(2d)}{n}} + 4M^2Q^2\sqrt{\frac{\log(\pi^2Q^2/3\delta)}{2n}}.
\end{aligned}$$

For any $\boldsymbol{\theta} \in \mathbb{R}^{m(d+1)}$ given in Theorem 3.2, choose the integer Q such that $\|\boldsymbol{\theta}\|_{\mathcal{P}} \le Q \le \|\boldsymbol{\theta}\|_{\mathcal{P}}+1$. Then we have

$$\begin{aligned}
|R_{\mathcal{D}}(\boldsymbol{\theta}) - R_S(\boldsymbol{\theta})| &\le 27M^2Q^2d^2\sqrt{\frac{2\log(2d)}{n}} + 4M^2Q^2\sqrt{\frac{\log(\pi^2Q^2/3\delta)}{2n}} \\
&\le 27M^2(\|\boldsymbol{\theta}\|_{\mathcal{P}} + 1)^2d^2\sqrt{\frac{2\log(2d)}{n}} + 4M^2(\|\boldsymbol{\theta}\|_{\mathcal{P}} + 1)^2\sqrt{\frac{\log\pi(\|\boldsymbol{\theta}\|_{\mathcal{P}} + 1)}{n} + \frac{\log(1/3\delta)}{2n}} \\
&\le 27M^2(\|\boldsymbol{\theta}\|_{\mathcal{P}} + 1)^2d^2\sqrt{\frac{2\log(2d)}{n}} + 4M^2(\|\boldsymbol{\theta}\|_{\mathcal{P}} + 1)^2\left\{\frac{\log[\pi(\|\boldsymbol{\theta}\|_{\mathcal{P}} + 1)]}{\sqrt{n}} + \sqrt{\frac{\log(1/3\delta)}{2n}}\right\} \\
&\le \frac{(\|\boldsymbol{\theta}\|_{\mathcal{P}} + 1)^2}{\sqrt{n}}2M^2(14d^2\sqrt{2\log(2d)} + \log[\pi(\|\boldsymbol{\theta}\|_{\mathcal{P}} + 1)] + \sqrt{2\log(1/3\delta)}),
\end{aligned}$$

where we have used the facts that $\sqrt{a+b} \le \sqrt{a} + \sqrt{b}$ for $a, b > 0$ and that $\sqrt{a} \le a$ for $a \ge 1$.

The bound just above holds with probability $1-\delta_Q$ for any pair $(\boldsymbol{\theta}, Q)$ as long as $\|\boldsymbol{\theta}\|_{\mathcal{P}} \le Q$. By the definition $\delta_Q = \frac{6\delta}{\pi^2Q^2}$, we have $\sum_{Q=1}^{\infty}\delta_Q = \delta$. Therefore, for any $\boldsymbol{\theta} \in \mathbb{R}^{m(d+1)}$ given in Theorem 3.2, the above bound holds with probability $1 - \delta$, which finishes the proof of Theorem 3.2. □

Finally, based on the approximation bound in Proposition 5.1 and the a posteriori generalization bound in Theorem 3.2, we show the a priori generalization bound in Theorem 3.3.

Proof of Theorem 3.3. Note that

$$R_{\mathcal{D}}(\boldsymbol{\theta}_{S,\lambda}) = R_{\mathcal{D}}(\tilde{\boldsymbol{\theta}}) + [R_{\mathcal{D}}(\boldsymbol{\theta}_{S,\lambda}) - J_{S,\lambda}(\boldsymbol{\theta}_{S,\lambda})] + [J_{S,\lambda}(\boldsymbol{\theta}_{S,\lambda}) - J_{S,\lambda}(\tilde{\boldsymbol{\theta}})] + [J_{S,\lambda}(\tilde{\boldsymbol{\theta}}) - R_{\mathcal{D}}(\tilde{\boldsymbol{\theta}})].$$

By definition, $J_{S,\lambda}(\boldsymbol{\theta}_{S,\lambda}) - J_{S,\lambda}(\tilde{\boldsymbol{\theta}}) \leq 0$. By Proposition 5.1, there exists $\tilde{\boldsymbol{\theta}}$ such that $R_{\mathcal{D}}(\tilde{\boldsymbol{\theta}}) \leq \frac{6M^2\|f\|_{\mathcal{B}}^2}{m}$. Therefore,

$$R_{\mathcal{D}}(\boldsymbol{\theta}_{S,\lambda}) \leq \frac{6M^2\|f\|_{\mathcal{B}}^2}{m} + [R_{\mathcal{D}}(\boldsymbol{\theta}_{S,\lambda}) - J_{S,\lambda}(\boldsymbol{\theta}_{S,\lambda})] + [J_{S,\lambda}(\tilde{\boldsymbol{\theta}}) - R_{\mathcal{D}}(\tilde{\boldsymbol{\theta}})]. \tag{5.7}$$

By Theorem 3.2, we have with probability at least $1-\delta/2$,

$$\begin{aligned}
R_{\mathcal{D}}(\boldsymbol{\theta}_{S,\lambda}) - J_{S,\lambda}(\boldsymbol{\theta}_{S,\lambda}) &= R_{\mathcal{D}}(\boldsymbol{\theta}_{S,\lambda}) - R_S(\boldsymbol{\theta}_{S,\lambda}) - \frac{\lambda}{\sqrt{n}}\|\boldsymbol{\theta}_{S,\lambda}\|_{\mathcal{P}}^2 \log[\pi(\|\boldsymbol{\theta}_{S,\lambda}\|_{\mathcal{P}}+1)] \\
&\leq \frac{1}{\sqrt{n}} 2M^2(\|\boldsymbol{\theta}_{S,\lambda}\|_{\mathcal{P}}+1)^2\{\log[\pi(\|\boldsymbol{\theta}_{S,\lambda}\|_{\mathcal{P}}+1)] + 14d^2\sqrt{2\log(2d)} + \sqrt{2\log(2/3\delta)}\} \\
&\quad - \frac{\lambda}{\sqrt{n}}\|\boldsymbol{\theta}_{S,\lambda}\|_{\mathcal{P}}^2 \log[\pi(\|\boldsymbol{\theta}_{S,\lambda}\|_{\mathcal{P}}+1)] \\
&\leq \frac{1}{\sqrt{n}} 4M^2(\|\boldsymbol{\theta}_{S,\lambda}\|_{\mathcal{P}}^2+1)\{\log[\pi(\|\boldsymbol{\theta}_{S,\lambda}\|_{\mathcal{P}}+1)] + 14d^2\sqrt{2\log(2d)} + \sqrt{2\log(2/3\delta)}\} \\
&\quad - \frac{\lambda}{\sqrt{n}}\|\boldsymbol{\theta}_{S,\lambda}\|_{\mathcal{P}}^2 \log[\pi(\|\boldsymbol{\theta}_{S,\lambda}\|_{\mathcal{P}}+1)] \\
&\leq \frac{1}{\sqrt{n}}\|\boldsymbol{\theta}_{S,\lambda}\|_{\mathcal{P}}^2 \log[\pi(\|\boldsymbol{\theta}_{S,\lambda}\|_{\mathcal{P}}+1)]\left\{4M^2[1+14d^2\sqrt{2\log(2d)} + \sqrt{2\log(2/3\delta)}] - \lambda\right\} \\
&\quad + \frac{4M^2}{\sqrt{n}}\log[\pi(\|\boldsymbol{\theta}_{S,\lambda}\|_{\mathcal{P}}+1)] + \frac{1}{\sqrt{n}}4M^2(14d^2\sqrt{2\log(2d)} + \sqrt{2\log(2/3\delta)}) \\
&\leq \frac{1}{\sqrt{n}}\|\boldsymbol{\theta}_{S,\lambda}\|_{\mathcal{P}}^2 \log[\pi(\|\boldsymbol{\theta}_{S,\lambda}\|_{\mathcal{P}}+1)]\left\{4M^2[2+14d^2\sqrt{2\log(2d)} + \sqrt{2\log(2/3\delta)}] - \lambda\right\} \\
&\quad + \frac{1}{\sqrt{n}}4M^2\left[\log(2\pi) + 14d^2\sqrt{2\log(2d)} + \sqrt{2\log(2/3\delta)}\right] \\
&\leq \frac{1}{\sqrt{n}}4M^2\left[\log(2\pi) + 14d^2\sqrt{2\log(2d)} + \sqrt{2\log(2/3\delta)}\right],
\end{aligned} \tag{5.8}$$

where we have used the facts that $(a+b)^2 \leq 2a^2+2b^2$ for all $a,b \geq 0$ and that $\lambda \geq 4M^2[2 + 14d^2\sqrt{2\log(2d)} + \sqrt{2\log(2/3\delta)}]$ in the second and last inequalities, respectively. By Theorem 3.2 again, with probability at least $1-\delta/2$, we have

$$\begin{aligned}
J_{S,\lambda}(\tilde{\boldsymbol{\theta}}) - R_{\mathcal{D}}(\tilde{\boldsymbol{\theta}}) &\leq \frac{1}{\sqrt{n}} 2M^2(\|\tilde{\boldsymbol{\theta}}\|_{\mathcal{P}}+1)^2\{\log[\pi(\|\tilde{\boldsymbol{\theta}}\|_{\mathcal{P}}+1)] + 14d^2\sqrt{2\log(2d)} + \sqrt{2\log(2/3\delta)}\} \\
&\quad + \frac{\lambda}{\sqrt{n}}\|\tilde{\boldsymbol{\theta}}\|_{\mathcal{P}}^2 \log[\pi(\|\tilde{\boldsymbol{\theta}}\|_{\mathcal{P}}+1)] \\
&\leq \frac{1}{\sqrt{n}} 4M^2(\|\tilde{\boldsymbol{\theta}}\|_{\mathcal{P}}^2+1)\{\log[\pi(\|\tilde{\boldsymbol{\theta}}\|_{\mathcal{P}}+1)] + 14d^2\sqrt{2\log(2d)} + \sqrt{2\log(2/3\delta)}\} \\
&\quad + \frac{\lambda}{\sqrt{n}}\|\tilde{\boldsymbol{\theta}}\|_{\mathcal{P}}^2 \log[\pi(\|\tilde{\boldsymbol{\theta}}\|_{\mathcal{P}}+1)].
\end{aligned} \tag{5.9}$$

Note that, by Proposition 5.1, we have $\|\tilde{\boldsymbol{\theta}}\|_{\mathcal{P}} \leq 2\|f\|_{\mathcal{B}}$. Hence, the inequality (5.9) becomes

$$\begin{aligned} J_{S,\lambda}(\tilde{\boldsymbol{\theta}}) - R_{\mathcal{D}}(\tilde{\boldsymbol{\theta}}) \leq & \frac{1}{\sqrt{n}} 4M^2(4\|f\|_{\mathcal{B}}^2 + 1)\{\log[\pi(2\|f\|_{\mathcal{B}} + 1)] + 14d^2\sqrt{2\log(2d)} + \sqrt{2\log(2/3\delta)}\} \\ & + \frac{4\lambda}{\sqrt{n}}\|f\|_{\mathcal{B}}^2 \log[\pi(2\|f\|_{\mathcal{B}} + 1)]. \end{aligned} \tag{5.10}$$

Adding the estimates in (5.7), (5.8), and (5.9) together completes the proof. □

6 Conclusion

In this paper, we theoretically analyzed the optimization problem arising in deep learning-based PDE solvers for second-order linear PDEs and two-layer neural networks under the assumption of over-parametrization (i.e., the network width is sufficiently large). In particular, we show that gradient descent can identify a global minimizer of the least-squares optimization problem for solving second-order linear PDEs. Note that we have fixed the samples in the least-squares optimization, while practical algorithms would randomly sample the PDE domain and its boundaries in every iteration of gradient descent. Hence, there is still a gap between the optimization problem analyzed in this paper and the practical algorithm. This gap can be filled by studying the convergence behavior of stochastic gradient descent, which will be left as future work.

We have also analyzed the generalization error of deep learning-based PDE solvers for second-order linear PDEs and two-layer neural networks, when the right-hand-side function of the PDE is in a Barron-type space and the least-squares optimization is regularized with a Barron-type norm, without the over-parametrization assumption. The Barron-type space and norm are adaptive to PDE problems and are different from those for regression problems. The global minimizer of the regularized least-squares problem can generalize well with a scaling of order $\frac{1}{m} + \frac{1}{\sqrt{n}}$, where m is the number of neurons and n is the number of data samples. Note that whether gradient descent methods can identify a global minimizer of the regularized least-squares problem is still unknown. This is left as interesting future work.

Acknowledgments. H. Y. was partially supported by the US National Science Foundation under award DMS-1945029.

References

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

[2] J. Berg and K. Nyström. A Unified Deep Artificial Neural Network Approach to Partial Differential Equations in Complex Geometries. *Neurocomputing*, 317:28 – 41, 2018.

[3] Julius Berner, Philipp Grohs, and Arnulf Jentzen. Analysis of the generalization error: Empirical risk minimization over deep artificial neural networks overcomes the curse of dimensionality in the numerical approximation of black-scholes partial differential equations. *CoRR*, abs/1809.03062, 2018.

[4] Yuan Cao and Quanquan Gu. Generalization bounds of stochastic gradient descent for wide and deep neural networks. *CoRR*, abs/1905.13210, 2019.

[5] G. Carleo and M. Troyer. Solving the Quantum Many-body Problem with Artificial Neural Networks. *Science*, 355:602–606, 2017.

[6] Liang Chen and Congwei Wu. A note on the expressive power of deep rectified linear unit networks in high-dimensional spaces. *Mathematical Methods in the Applied Sciences*, 42(9):3400–3404, 2019.

[7] Zixiang Chen, Yuan Cao, Difan Zou, and Quanquan Gu. How much over-parameterization is sufficient to learn deep relu networks? *CoRR*, arXiv:1911.12360, 2019.

[8] M. W. M. G. Dissanayake and N. Phan-Thien. Neural-network-based Approximations for Solving Partial Differential Equations. *Comm. Numer. Methods Engrg.*, 10:195–201, 1994.

[9] Simon S. Du, Jason D. Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. *CoRR*, abs/1811.03804, 2018.

[10] Simon S. Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations*, 2019.

[11] W. E and B. Yu. The Deep Ritz Method: a Deep Learning-based Numerical Algorithm for Solving Variational Problems. *Commun. Math. Stat.*, 6:1–12, 2018.

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

[13] Weinan E, Chao Ma, and Qingcan Wang. A priori estimates of the population risk for residual networks. 2019.

[14] Weinan E, Chao Ma, and Lei Wu. A priori estimates of the population risk for two-layer neural networks. *Communications in Mathematical Sciences*, 17(5):1407 – 1425, 2019.

[15] Weinan E, Chao Ma, and Lei Wu. A comparative analysis of optimization and generalization properties of two-layer neural network and random feature models under gradient descent dynamics. *Science China Mathematics*, 63(7):1235–1258, Jan 2020.

[16] Weinan E and Qingcan Wang. Exponential convergence of the deep neural network approximation for analytic functions. *CoRR*, abs/1807.00297, 2018.

[17] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning.* MIT Press, Cambridge, 2016.

[18] Yiqi Gu, Chunmei Wang, and Haizhao Yang. Structure probling neural network deflation. *CoRR*, 2020.

[19] Yiqi Gu, Haizhao Yang, and Chao Zhou. Selectnet: Self-paced learning for high-dimensional partial differential equations. *CoRR*, abs/2001.04860, 2020.

[20] J. Han, A. Jentzen, and W. E. Solving High-dimensional Partial Differential Equations Using Deep Learning. *Proc. Natl. Acad. Sci. USA*, 115:8505–8510, 2018.

[21] Jiequn Han and Jihao Long. Convergence of the deep bsde method for coupled fbsdes. *ArXiv*, abs/1811.01165, 2018.

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[23] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *CoRR*, abs/1608.06993, 2016.

[24] Jianguo Huang, Haoqin Wang, and Haizhao Yang. Int-deep: A deep learning initialized iterative method for nonlinear problems. *Journal of Computational Physics*, page 109675, 2020.

[25] M. Hutzenthaler, A. Jentzen, Th. Kruse, and T. A. Nguyen. A proof that rectified deep neural networks overcome the curse of dimensionality in the numerical approximation of semilinear heat equations. Technical Report 2019-10, Seminar for Applied Mathematics, ETH Zürich, Switzerland, 2019.

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

[27] Y. Khoo, J. Lu, and L. Ying. Solving for High-dimensional Committor Functions Using Artificial Neural Networks. *Res. Math. Sci.*, 6:1–13, 2019.

[28] I.E. Lagaris, A. Likas, and D. I. Fotiadis. Artificial Neural Networks for Solving Ordinary and Partial Differential Equations. *IEEE Trans. Neural Networks*, 9:987–1000, 1998.

[29] Shiyu Liang and R. Srikant. Why deep neural networks? *CoRR*, abs/1610.04161, 2016.

[30] Y. Liao and P. Ming. Deep Nitsche method: deep Ritz method with essential boundary conditions. *arXiv e-prints*, arXiv:1912.01309, 2019.

[31] Jianfeng Lu, Zuowei Shen, Haizhao Yang, and Shijun Zhang. Deep Network Approximation for Smooth Functions. *arXiv e-prints*, page arXiv:2001.03040, January 2020.

[32] Yiping Lu, Chao Ma, Yulong Lu, Jianfeng Lu, and Lexing Ying. A mean-field analysis of deep resnet and beyond: Towards provable optimization via overparameterization from depth. *CoRR*, abs/2003.05508, 2020.

[33] Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018.

[34] Hadrien Montanelli and Qiang Du. New error bounds for deep relu networks using sparse grids. *SIAM Journal on Mathematics of Data Science*, 1(1):78–92, 2019.

[35] Hadrien Montanelli and Haizhao Yang. Error bounds for deep relu networks using the kolmogorov–arnold superposition theorem. *Neural Networks*, 129:1 – 6, 2020.

[36] Hadrien Montanelli, Haizhao Yang, and Qiang Du. Deep ReLU networks overcome the curse of dimensionality for bandlimited functions. 2019.

[37] Joost A.A. Opschoor, Christoph Schwab, and Jakob Zech. Exponential relu dnn expression of holomorphic maps in high dimension. Technical report, Zurich, 2019-07.

[38] T. Poggio, H. N. Mhaskar, L. Rosasco, B. Miranda, and Q. Liao. Why and when can deep—but not shallow—networks avoid the curse of dimensionality: A review. *International Journal of Automation and Computing*, 14:503–519, 2017.

[39] M. Raissi, P. Perdikaris, and G.E. Karniadakis. Physics-informed Neural Networks: a Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations. *J. Comput. Phys.*, 378:686 – 707, 2019.

[40] K. Rudd and S. Ferrari. A Constrained Integration (CINT) Approach to Solving Partial Differential Equations Using Artificial Neural Networks. *Neurocomputing*, 155:277 – 285, 2015.

[41] S. Shalev-Shwartz and S. Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[42] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Nonlinear approximation via compositions. *Neural Networks*, 119:74 – 84, 2019.

[43] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Deep network approximation characterized by number of neurons. *Communications in Computational Physics*, 28(5):1768–1811, 2020.

[44] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Neural network approximation: Three hidden layers are enough. *arXive:2010.14075*, 2020.

[45] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Deep network approximation with discrepancy being reciprocal of width to power of depth. *Neural Computation*, To appear.

[46] Yeonjong Shin, Jerome Darbon, and George Em Karniadakis. On the convergence of physics informed neural networks for linear second-order elliptic and parabolic type pdes, 2020.

[47] J. Sirignano and K. Spiliopoulos. DGM: a Deep Learning Algorithm for Solving Partial Differential Equations. *J. Comput. Phys.*, 375:1339 – 1364, 2018.

[48] E. Weinan, Chao Ma, and Lei Wu. Barron spaces and the compositional function spaces for neural network models. *ArXiv*, abs/1906.08039, 2019.

[49] Yunfei Yang and Yang Wang. Approximation in shift-invariant spaces with deep ReLU neural networks. *arXiv e-prints*, page arXiv:2005.11949, May 2020.

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

[51] Dmitry Yarotsky and Anton Zhevnerchuk. The phase diagram of approximation rates for deep neural networks. *arXiv e-prints*, page arXiv:1906.09477, June 2019.

[52] Y. Zhang, Z.-Q. J. Xu, T. Luo, and Z. Ma. A type of generalization error induced by initialization in deep neural networks. *arXiv e-prints*, arXiv:1905.07777, 2019.