East Asian Journal on Applied Mathematics
doi: 10.4208/eajam.2023-128.051023

Vol. **xx**, No. x, pp. 1-33
xxx 2024

Approximation and Generalization of DeepONets for Learning Operators Arising from a Class of Singularly Perturbed Problems

Ting Du[1], Zhongyi Huang[1] and Ye Li[2,*]

[1]*Department of Mathematics Sciences, Tsinghua University, Beijing 100084, China.*
[2]*College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China.*

Received 11 May 2023; Accepted (in revised version) 5 October 2023.

Abstract. Singularly perturbed problems present inherent difficulty due to the presence of thin layers in their solutions. To overcome this difficulty, we propose using deep operator networks (DeepONets), a method previously shown to be effective in approximating nonlinear operators between infinite-dimensional Banach spaces. In this paper, we demonstrate for the first time the application of DeepONets to one-dimensional singularly perturbed problems, achieving promising results that suggest their potential as a robust tool for solving this class of problems. We consider the convergence rate of the approximation error incurred by the operator networks in approximating the solution operator, and examine the generalization gap and empirical risk, all of which are shown to converge uniformly with respect to the perturbation parameter. By utilizing Shishkin mesh points as locations of the loss function, we conduct several numerical experiments that provide further support for the effectiveness of operator networks in capturing the singular layer behavior.

AMS subject classifications: 65M10, 78A48

Key words: Deep operator network, singularly perturbed problem, Shishkin mesh, uniform convergence.

1. Introduction

Singularly perturbed problems (SPP) are widely employed in diverse areas of applied mathematics, such as fluid dynamics, aerodynamics, meteorology, and modeling of semiconductor devices, among others. These problems incorporate a small parameter ε which typically appears before the highest order derivative term and reflects medium properties. As $\varepsilon \to 0$, the derivative (or higher-order derivative) of the solution to the problem becomes infinite, leading to the emergence of boundary or inner layers. In these regions, the

*Corresponding author. *Email addresses:* dt20@mails.tsinghua.edu.cn (T. Du), zhongyih@mail.tsinghua.edu.cn (Z. Huang), yeli20@nuaa.edu.cn (Y. Li)

solution (or its derivative) undergoes significant changes, while away from the layers, the behavior of the solution is regular and slow. Due to the presence of thin layers, conventional methods for solving this class of problems exhibit either high computational complexity or significant computational expense. For a comprehensive analysis and numerical treatment of singularly perturbed problems, refer to the books by Miller *et al.* [24] and Roos *et al.* [28].

Deep neural networks possess the universal approximation property, which allows them to approximate any continuous or measurable finite-dimensional function with arbitrary accuracy. Consequently, they have become a popular tool for solving partial differential equations (PDEs) by serving as a trial space. PDE solvers based on neural networks include Feynman-Kac formula-based methods [2,6,7] and physics-informed neural networks (PINNs) [27]. Moreover, machine learning techniques can be used for operator learning (i.e., mapping one infinite-dimensional Banach space to another) to approximate the underlying solution operator of an equation. To accomplish this, new network structures have been proposed, such as deep operator networks (DeepONets) [21] and neural operators [14,16].

The broad adoption of machine learning techniques in solving PDEs has introduced novel approaches for handling singularly perturbed problems. However, the spectral bias phenomenon [26], which refers to the difficulty of neural networks in capturing sharp fluctuations of the function, poses a significant challenge for directly applying machine learning techniques to solve such problems. In [1], it was observed that the traditional PINNs approach did not produce a satisfactory solution nor capture the singular behavior of the boundary layer. To overcome this problem, a deformation of the traditional PINN based on singular perturbation theory is presented in [1].

Operator learning techniques have received little attention in the context of singularly perturbed problems, leaving an important research gap to be filled. Our focus is on DeepOnet, a simple yet powerful model. It originated from the universal approximation theorem proved by Chen *et al.* [4], where the shallow network was later extended to a deep network by Lu *et al.* [21], resulting in the proposal of DeepONets. This theorem was further extended to more general operators in [15], thereby providing theoretical assurance of the validity of DeepONet to approximate the solution operator. Specifically, for any error bound ϵ, there exists an operator network that can approximate the operator with an accuracy not exceeding ϵ, while also guaranteeing that the size of the network is at most polynomially increasing concerning $1/\epsilon$. Theoretical and experimental results [5,15] support the polynomially increasing property of DeepONets in some cases. DeepONet has demonstrated promising performance in various applications, including fractional derivative operators [21], stochastic differential equations [21], and advection-diffusion equations [5]. To extend the applicability of DeepONet to more general situations, several deformations have been proposed, including physics-informed DeepONet [32], multiple-input DeepONet [10], pre-trained DeepONet for multi-physics systems [3,23], proper orthogonal decomposition-based DeepONet (POD-DeepONet) [22], and Bayesian DeepONet [17].

In this paper, we employ DeepONets to approximate the solution operator of the convection-diffusion problem, with particular attention to the case of Dirichlet boundary conditions. As it is known that for non-homogeneous Dirichlet boundary conditions, the solution

u can be obtained by subtracting a linear function satisfying the original boundary condition to obtain a solution u^* satisfying the homogeneous boundary condition, we consider the following one-dimensional singularly perturbed problem:

$$\begin{aligned} -\varepsilon u'' + pu' + qu &= f, \quad x \in (0,1), \\ u(0) = u(1) &= 0, \end{aligned} \tag{1.1}$$

where $p(x) \geq \alpha > 0$, and both p and q are assumed to be sufficiently differentiable. The solution to this problem typically exhibits a boundary layer near $x = 1$. Define the operator $\mathscr{G} : L^2([0,1]) \to H_0^1([0,1]) \cap H^2([0,1]),\ f \mapsto u$, where u satisfies

$$\varepsilon \int_0^1 u'v'dx + \int_0^1 pu'vdx + \int_0^1 quvdx = \int_0^1 fvdx \quad \text{for all} \quad v \in H_0^1. \tag{1.2}$$

Suppose $\mathscr{G}$ is Lipschitz continuous. We utilize DeepONets to approximate $\mathscr{G}$, which is subject to three sources of error: approximation error, generalization error, and optimization error. We focus our attention on the first two sources of error. For singularly perturbed problems, it is critical to determine whether the error is ε-uniform. We adopt a general framework presented in [15] to estimate the approximation error and analyze the convergence rate of this error. Our analysis shows that the approximation error is ε-uniform. Given the finite nature of the dataset, training a DeepONet is constrained by limited data availability. Hence, the selection of data assumes critical importance in determining the performance of the model. Our numerical experiments demonstrate that the performance of the trained model is significantly affected by the choice of locations in the loss function. Additionally, the generalization ability of the model — i.e. its capacity to perform well on data points beyond the training dataset, is of particular interest in our study. We select the Shishkin mesh points [13] as locations and analyze the generalization gap and empirical risk of our model. The results show that the model exhibits strong generalization capability when applied to boundary layer problems, and we present numerical examples to complement our theoretical findings.

The structure of the paper is organized as follows. In Section 2, we provide a brief introduction to the DeepONets structure and properties of the operator $\mathscr{G}$. Section 3 presents an analysis of the approximation error that arises when the operator network approximates the operator $\mathscr{G}$. In Section 4, we investigate the generalization gap and empirical risk. The Appendix A includes some of the proof procedures. In Section 5, we provide numerical demonstrations of the efficacy of the proposed method, as well as a comparative analysis of the model's performance under different location selections. Finally, we summarize the work of this study and also discuss potential future work.

2. Preliminaries

2.1. DeepONets

A neural network is a composition of affine functions $\mathscr{A}_l : \mathbb{R}^{d_{l-1}} \to \mathbb{R}^{d_l}$ and activation functions $\sigma : \mathbb{R} \to \mathbb{R}$. Given an input $\boldsymbol{x} \in \mathbb{R}^{d_0}$, the output of the l-th layer neural network

is defined as

$$\boldsymbol{y}_l(\boldsymbol{x}) = \sigma \circ \mathscr{A}_l(\boldsymbol{y}_{l-1}) = \sigma\big(\boldsymbol{W}_l \boldsymbol{y}_{\boldsymbol{l-1}}(\boldsymbol{x}) + \boldsymbol{b}_l\big),$$

where $\boldsymbol{W}_l$ and $\boldsymbol{b}_l$ are the weight matrix and bias vector of the affine function. The output of the last layer of the neural network is the output of the whole neural network and can be expressed as follows:

$$\mathscr{N}_\theta(\boldsymbol{x}) = \boldsymbol{y}_L(\boldsymbol{x}) = \mathscr{A}_L \circ \sigma \circ \mathscr{A}_{L-1} \circ \cdots \circ \sigma \circ \mathscr{A}_2 \circ \sigma \circ \mathscr{A}_1(\boldsymbol{x}), \tag{2.1}$$

where $\theta = \{\boldsymbol{W}_l, \boldsymbol{b}_l\}_{1 \le l \le L}$ is the set of learnable parameters obtained by minimizing the loss function during the training of the neural network. The size of a neural network is commonly quantified by its depth L and width $\max_l d_l$.

In deep neural networks, the ReLU, Sigmoid, and Tanh activation functions are commonly used. Other activation functions, such as Sine [29], SReLU [19], and adaptive activation functions [9, 25], have also been proposed.

A DeepONet comprises two sub-networks in the form of (2.1): one to capture the input function's information at a fixed number of points x_i, with $i = 1, \ldots, m$ (i.e., the branch net), and another to encode the output functions' locations (i.e., the trunk net). In this work, we adopt the form presented in [15] for the purpose of theoretical analysis, where DeepONet is decomposed into the three operators — viz.

1. Encoder. For a given constant m and m-dimensional vector $\{x_i\}_{i=1}^m$, $x_i \in D$, Encoder $\mathscr{E}$ is defined as
$$\mathscr{E} : L^2(D) \ \to \ \mathbb{R}^m, \quad f \ \mapsto \ \{f(x_i)\}_{i=1}^m.$$
2. Approximator. Approximator $\mathscr{A}$ is a neural network, defined by Eq. (2.1), takes a specific form given by
$$\mathscr{A} : \mathbb{R}^m \ \to \ \mathbb{R}^p, \quad \{f_j\}_{j=1}^m \ \mapsto \ \{\mathscr{A}_k\}_{k=1}^p.$$
3. Reconstructor. Trunk net $\boldsymbol{\tau}$ is a neural network of the form (2.1) with $d_0 = 1$ and $d_L = p+1$, which can be expressed as $\boldsymbol{\tau}(x) = (\tau_0(x), \tau_1(x), \ldots, \tau_p(x))$, and then the reconstructor $\mathscr{R}$ induced by $\boldsymbol{\tau}$ can be defined as
$$\mathscr{R} = \mathscr{R}_{\boldsymbol{\tau}} : \mathbb{R}^p \ \to \ L^2(U), \quad \{\alpha_k\}_{k=1}^p \ \mapsto \ \tau_0(\cdot) + \sum_{k=1}^p \alpha_k \tau_k(\cdot).$$

Then the DeepONet, as proposed in [21], can be expressed as

$$\mathscr{N} : L^2(D) \ \to \ L^2(U), \quad f \ \mapsto \ \mathscr{R} \circ \mathscr{A} \circ \mathscr{E}(f)(\cdot). \tag{2.2}$$

A diagram representation of the model's structure is shown in Fig. 1.

DeepONet is a neural network model with two inputs: a function f and a location x. Given that neural networks require finite-dimensional inputs, f is evaluated at m-dimensional location vectors through point-wise evaluation $\mathscr{E}$, resulting in an m-dimensional vector that serves as input for the branch net. As a supervised learning model, DeepONet relies on a large dataset of input-output pairs to train the network, a process that is performed offline. Once the network is trained, it can be used as an operator, providing an approximation $\mathscr{N}(f)(x)$ of the equation's solution for any reasonable input function f and location x.

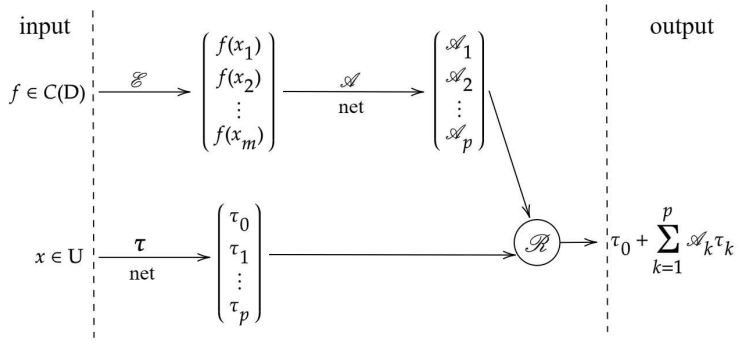


Figure 1: Architecture of DeepONets.

2.2. Prior estimate

We introduce two lemmas that describe the fundamental properties of this operator. Lemma 2.2 provides a sufficient condition for the Lipschitz continuity of $\mathscr{G}$.

Lemma 2.1. *[cf. Kellogg* et al. *[12]] Given $f \in C^\infty$ such that*

$$|f(x,\varepsilon)| \le K\left(1+\varepsilon^{-1}\exp\left(-\frac{\alpha(1-x)}{\varepsilon}\right)\right),$$

the solution $\mathscr{G}(f)$ to Eq. (1.2) *satisfies*

$$|\mathscr{G}(f)(x)| \le C,$$

where C is a constant that depends only on K.

Lemma 2.2. *Under the conditions $0 < \varepsilon \ll 1$, $p(x)$, $q(x) \in C^\infty$, $p(x) \ge \alpha > 0$, $q(x) - p'(x)/2 \ge \beta > 0$, with α and β remaining independent of ε, and $f \in L^2([0,1])$, the weak solution u to Eq.* (1.2) *satisfies the inequality*

$$\int_0^1 u^2 dx \le C\int_0^1 f^2 dx,$$

where C is a constant independent of ε.

Proof. To prove the inequality stated in the theorem, we start by setting $v = u$ in the Eq. (1.2), which gives us

$$\varepsilon\int_0^1 (u')^2 dx + \int_0^1 \left(q(x) - \frac{p'(x)}{2}\right)u^2 dx = \int_0^1 f u dx.$$

We can then use the condition $q(x) - p'(x)/2 \ge \beta > 0$, along with the Cauchy-Schwarz inequality, to obtain the desired inequality. □

3. Approximation Error

In this section, the approximation error generated by the operator network in approximating the solution operator $\mathscr{G}$ is analyzed quantitatively. Given μ as a probability measure on $L^2([0,1])$, referring to the method in [15], we use $L^2(\mu)$ norm to measure the error between the operator $\mathscr{N}$ and the operator $\mathscr{G}$, i.e.

$$\widehat{\mathscr{E}}(\mathscr{N}) = \|\mathscr{G}-\mathscr{N}\|_{L^2(\mu)} = \left(\int_{L^2([0,1])} \|\mathscr{G}(f)-\mathscr{N}(f)\|^2_{L^2([0,1])} d\mu(f)\right)^{1/2}. \tag{3.1}$$

The universal approximation theorem of T. Chen and H. Chen [4] and its generalized version [15] both guarantee the existence of a DeepONet of the form (2.2) that can approximate a given nonlinear operator $\mathscr{G}$ with an error not exceeding a pre-specified bound ϵ. However, these theorems do not provide specific information on determining the configuration of a DeepONet, such as the size of the branch and trunk networks. Our goal is to better understand the performance of DeepONets in approximating nonlinear operators by conducting a quantitative analysis of their efficiency, specifically investigating how variations in network size and structure affect their approximation accuracy and computational efficiency. Lanthaler *et al.* [15] propose a general framework for error estimation, which includes results for general operators and several specific examples. Although operator networks have not yet been applied to singularly perturbed problems, the universal approximation theorem assures the existence of a DeepONet of the form (2.2) that can approximate the solution operators for this class of problems. Our main focus in this section is to quantitatively analyze the efficiency of DeepONets in approximating these operators.

3.1. Error estimate

To estimate (3.1), it is necessary to first identify the underlying space. Let $\mu = \mathscr{N}(\mathbb{E}_\mu, \Gamma)$ be the Gaussian measure on the Hilbert space X, where $\mathbb{E}_\mu = \int_X u d\mu(u)$ is the mean and Γ is the covariance operator satisfying that, for any $k, l \in X$ and $\boldsymbol{x} \sim \mu$,

$$\langle k, \Gamma l\rangle = \mathbb{E}\langle k, \boldsymbol{x}-\mathbb{E}_\mu\rangle\langle \boldsymbol{x}-\mathbb{E}_\mu, l\rangle,$$

which implies that $\Gamma = \mathbb{E}(\boldsymbol{x}-\mathbb{E}_\mu)\otimes(\boldsymbol{x}-\mathbb{E}_\mu)$.

Let $\{\lambda_k, \phi_k\}_{k=1}^{+\infty}$ denote eigenvalue-eigenvector pairs of the operator Γ, where

$$\lambda_1 \geq \lambda_2 \geq \cdots,$$

and the set $\{\phi_k\}_{k=1}^{+\infty}$ is orthogonal. Additionally, let $\{\xi_k\}_{k=1}^{+\infty}$ be a set of independent identically distributed (i.i.d.) random variables with $\xi_1 \sim \mathscr{N}(0,1)$. We can express $\boldsymbol{x} \sim \mu$ using the Karhunen-Loève (K-L) expansion [20], i.e.

$$\boldsymbol{x} = \mathbb{E}_\mu + \sum_{k=1}^{+\infty} \sqrt{\lambda_k}\xi_k\phi_k.$$

This series converges in the quadratic mean uniformly on $[0,1]$.

In this work, we consider the input space to be endowed with a Gaussian measure $\mu = \mathcal{N}(0, \mathscr{C})$, where $\mathscr{C}$ is a covariance operator represented as an integral operator with a kernel, as previously employed in [15, 21]. Specifically, the kernel $k_p(x_1, x_2)$ is the periodization of the radial basis function (RBF) kernel $k_l(x_1, x_2)$, where $k_l(x_1, x_2) = \exp(-|x_1 - x_2|^2/2l^2)$ is a commonly used kernel. The periodization is defined as

$$k_p(x_1, x_2) = \sum\nolimits_{n\in\mathbb{Z}} \exp\left(-\frac{|x_1 - x_2 - n|^2}{2l^2}\right). \tag{3.2}$$

The parameter $l > 0$ in the RBF kernel controls the smoothness of the sampled function, with larger values of l corresponding to smoother functions. The eigenvalues and eigenvectors of the covariance operator $\mathscr{C}$ are characterized by

$$\lambda_k = \sqrt{2\pi} l e^{-2\pi^2 k^2 l^2}, \quad \phi_k(x) = e^{i2\pi kx},$$

respectively, where $k \in \mathbb{Z}$. This can be confirmed by noting that

$$\int_0^1 k_p(x, y)\phi_k(y)dy = \lambda_k \phi_k(x).$$

Then by the K-L expansion, any $f \sim \mu$ has the representation as

$$f = \sum_{k=-\infty}^{+\infty} \sqrt{\lambda_k}\xi_k\phi_k, \tag{3.3}$$

where $\{\xi_k\}_{k=-\infty}^{+\infty}$ is an i.i.d. sequence of $\mathcal{N}(0, 1)$ random variables.

We proceed to analyze the approximation error specified by (3.1). We maintain the notation from Section 2, where m denotes the number of input layer neurons in the branch network $\mathscr{A}$, and p represents the number of output layer neurons in $\mathscr{A}$. Additionally, we assume that both m and p are odd numbers and introduce $K = (p-1)/2$ and $M = (m-1)/2$. The following result emerges, with a rigorous proof provided in the subsequent subsection.

Theorem 3.1. *Assuming the solution operator for Eq.* (1.1) *is Lipschitz continuous, for any fixed, arbitrarily small $\gamma \in (0, 1)$ and $\eta > 0$ (which may depend on p), there exists a ReLU Trunk net $\boldsymbol{\tau}$ with $width(\boldsymbol{\tau}) = \mathcal{O}(\max\{p, (\eta/p)^{1/(\gamma-2)}\})$ and $depth(\boldsymbol{\tau}) = 2$, and a ReLU Branch net $\mathscr{A}$ with $width(\mathscr{A}) = \mathcal{O}(\max\{m, p\})$ and $depth(\mathscr{A}) = 2$. These nets yield an operator network $\mathcal{N}$ such that the following estimate holds:*

$$\widehat{\mathscr{E}}(\mathcal{N}) = \|\mathscr{G} - \mathcal{N}\|_{L^2(\mu)} \lesssim \eta + e^{-\pi^2 l^2 K^2} + e^{-\pi^2 l^2 M^2}.$$

Here, $K = (p-1)/2$, $M = (m-1)/2$, and l represents the parameter in the RBF kernel (3.2).

Remark 3.1. Here, the width refers to the maximum number of neurons across all layers, while the depth denotes the total number of layers, including hidden layers and the output layer, as discussed in Section 2. The conclusion of Theorem 3.1 asserts the existence of an operator network with a specified width and depth that can approximate the solution operator of Eq. (1.1) with high accuracy. However, this does not imply that solely operator networks of this particular size are capable of approximating the solution operator.

Theorem 3.1 shows that different choices of η correspond to different sizes of the trunk network and various error bounds for $\widehat{\mathscr{E}}$. Notably, by setting $\eta = e^{-\pi^2 l^2 K^2}$, the approximation error $\widehat{\mathscr{E}}$ exhibits super-exponential decay with respect to p. However, this comes at the cost of an exponential growth in the size of the trunk net, which is highly undesirable. A preferable scenario is for the size of the trunk net to exhibit polynomial growth with respect to p, which can be achieved by setting $\eta = p^{-n}$, where n is a fixed, positive constant. And for notational simplicity, we adopt $\eta = p^{-n}$ throughout the remainder of this paper. In other cases, identical reasoning leads to the corresponding conclusions. By applying Theorem 3.1 and introducing the notation size(Φ) to represent the number of neurons in neural network Φ, we arrive at the following theorem in a straightforward manner.

Theorem 3.2. *Under the assumption of Lipschitz continuity of the solution operator for the Eq.* (1.2)*, a ReLU Branch net* $\mathscr{A}$ *of* size($\mathscr{A}$) $= \mathscr{O}(\max\{m,p\})$ *and a ReLU Trunk net* $\boldsymbol{\tau}$ *of* size($\boldsymbol{\tau}$) $= \mathscr{O}(p^{(n+1)/(2-\gamma)})$ *can be constructed, for any positive number n and any fixed, arbitrarily small constant* $\gamma \in (0,1)$*. These nets yield an operator network* $\mathscr{N}$ *that satisfies the following estimate:*

$$\widehat{\mathscr{E}}(\mathscr{N}) = \|\mathscr{G} - \mathscr{N}\|_{L^2(\mu)} \lesssim p^{-n} + e^{-\pi^2 l^2 K^2} + e^{-\pi^2 l^2 M^2}.$$

Here, $K = (p-1)/2$*,* $M = (m-1)/2$*, and* l *represents the parameter in the RBF kernel* (3.2).

3.2. Proof of Theorem 3.1

Following the workflow outlined in [15], we introduce here the operators $\mathscr{D}$ and $\mathscr{P}$ as approximations to the inverses of $\mathscr{E}$ and $\mathscr{R}$, and employ $\widehat{\mathscr{E}}_{\mathscr{E}}$, $\widehat{\mathscr{E}}_{\mathscr{R}}$, and $\widehat{\mathscr{E}}_{\mathscr{A}}$ to quantify the differences between $\mathscr{D}\circ\mathscr{E}$ and the identity operator $Id : L^2([0,1]) \to L^2([0,1])$, between $\mathscr{R}\circ\mathscr{P}$ and $Id : L^2([0,1]) \to L^2([0,1])$, and between $\mathscr{A}$ and $\mathscr{P}\circ\mathscr{G}\circ\mathscr{D}$, respectively. The formulae for these errors are shown below

$$\begin{aligned}
\widehat{\mathscr{E}}_{\mathscr{E}} &= \left(\int_{L^2([0,1])} \|\mathscr{D}\circ\mathscr{E}(f) - f\|^2_{L^2([0,1])} d\mu(f)\right)^{1/2},\\
\widehat{\mathscr{E}}_{\mathscr{R}} &= \left(\int_{L^2([0,1])} \|\mathscr{R}\circ\mathscr{P}(u) - u\|^2_{L^2([0,1])} d(\mathscr{G}_{\#}\mu)(u)\right)^{1/2},\\
\widehat{\mathscr{E}}_{\mathscr{A}} &= \left(\int_{\mathbb{R}^m} \|\mathscr{A}(\mathbf{y}) - \mathscr{P}\circ\mathscr{G}\circ\mathscr{D}(\mathbf{y})\|^2_{l^2(\mathscr{R}^p)} d(\mathscr{E}_{\#}\mu)(\mathbf{y})\right)^{1/2}.
\end{aligned}$$

Then we can decompose the approximation error (3.1) into the following terms:

Lemma 3.1. *[cf. Lanthaler* et al. *[15]] The approximation error* (3.1) *associated with* $\mathscr{G}$ *and the operator network* $\mathscr{N} = \mathscr{R}\circ\mathscr{A}\circ\mathscr{E}$ *can be bounded as follows:*

$$\widehat{\mathscr{E}} \le Lip(\mathscr{G}) Lip(\mathscr{R}\circ\mathscr{P})\widehat{\mathscr{E}}_{\mathscr{E}} + Lip(\mathscr{R})\widehat{\mathscr{E}}_{\mathscr{A}} + \widehat{\mathscr{E}}_{\mathscr{R}},$$

where $Lip(\cdot)$ *denotes the Lipschitz norm.*

According to Lemma 3.1, the approximation error $\widehat{\mathscr{E}}$ can be effectively constrained by controlling $\widehat{\mathscr{E}}_{\mathscr{E}}$, $\widehat{\mathscr{E}}_{\mathscr{A}}$, and $\widehat{\mathscr{E}}_{\mathscr{R}}$. We begin by presenting the result for the error term $\widehat{\mathscr{E}}_{\mathscr{E}}$.

Lemma 3.2. *Let $x_j = j/m,\ j = 1,2,3,\ldots,m$ with $m = 2M+1$ be m equidistant points on $[0,1]$. Consider the encoder $\mathscr{E} : L^2([0,1]) \to \mathbb{R}^m$, defined as $\mathscr{E}(f) = (f(x_1),\ldots,f(x_m))$, then, there exists an operator $\mathscr{D} : \mathbb{R}^m \to L^2([0,1])$ such that*

$$\widehat{\mathscr{E}}_{\mathscr{E}} = \|\mathscr{D}\circ\mathscr{E} - Id\|_{L^2(\mu)} \lesssim e^{-\pi^2 l^2 M^2}.$$

Remark 3.2. Here the operator $\mathscr{D}$ is given by

$$\mathscr{D}(f_1,f_2,\ldots,f_m)(x) = \sum_{k=-M}^{M} \widehat{f}_k e^{i2\pi kx},$$

where

$$\widehat{f}_k = \frac{1}{m}\sum_{j=1}^{m} f_j e^{-i2\pi jk/m}.$$

The proof of this result is omitted for brevity, but for a detailed derivation, we refer the reader to the proof of [15, Lemma 3.8]. The use of the symbol '$\lesssim$' in the conclusion of Lemma 3.2 implies that a constant C, which is independent of parameters such as m, p, ε and l, has been omitted from the right-hand side of the inequality. We will adopt this notation without further explanation in subsequent sections.

We now turn to the analysis of the remaining error terms, namely $\widehat{\mathscr{E}}_{\mathscr{R}}$ and $\widehat{\mathscr{E}}_{\mathscr{A}}$. To proceed, we assume that p is an odd integer, specifically $p = 2K+1$. For simplicity, we denote $\mathscr{G}_{\#}\mu$ as ν. We introduce the operator $\widetilde{\mathscr{R}}$, defined as

$$\widetilde{\mathscr{R}}(a_{-K},a_{-K+1},\ldots,a_K) = \mathbb{E}_\nu + \sum_{j=-K}^{K} a_j \widetilde{\tau}_j,$$

where $\{\widetilde{\tau}_j\}_{j=-K}^{K}$ are obtained from $\{\mathscr{G}(\phi_j)\}_{j=-K}^{K}$ (i.e., $\{\mathscr{G}(e^{i2\pi jx})\}_{j=-K}^{K}$) by Gram-Schmidt orthogonalization. Let $\mathscr{P}$ be the operator defined by

$$\mathscr{P}(u) = \big((u-\mathbb{E}_\nu,\widetilde{\tau}_{-K}),\ldots,(u-\mathbb{E}_\nu,\widetilde{\tau}_K)\big),$$

then the $L^2(\nu)$ norm of the difference between $\widetilde{\mathscr{R}}\circ\mathscr{P}$ and Id can be estimated in the following way:

$$\begin{aligned}
&\int_{L^2} \|\widetilde{\mathscr{R}}\circ\mathscr{P}(u) - u\|_{L^2}^2 d\,\nu(u)\\
=&\int_{L^2} \inf_{\widehat{u}\in\, \mathbb{E}_\nu+\mathrm{span}\{\widetilde{\tau}_{-K},\ldots,\widetilde{\tau}_K\}} \|\widehat{u}-u\|_{L^2}^2 d\,\nu(u)\\
=&\int_{L^2} \inf_{\widehat{u}\in\, \mathscr{G}(\mathbb{E}_\mu)+\mathrm{span}\{\mathscr{G}(\phi_{-K}),\ldots,\mathscr{G}(\phi_K)\}} \|\widehat{u}-u\|_{L^2}^2 d\,\nu(u)
\end{aligned}$$

$$
\begin{aligned}
&= \int_{L^2} \inf_{\widehat{f} \in \mathbb{E}_\mu + \mathrm{span}\{\phi_{-K},\ldots,\phi_K\}} \|\mathscr{G}(\widehat{f}) - \mathscr{G}(f)\|_{L^2}^2 d\mu(f) \\
&\le Lip(\mathscr{G})^2 \int_{L^2} \inf_{\widehat{f} \in \mathbb{E}_\mu + \mathrm{span}\{\phi_{-K},\ldots,\phi_K\}} \|\widehat{f} - f\|_{L^2}^2 d\mu(f) \\
&= Lip(\mathscr{G})^2 \int_{L^2} \left\| \mathbb{E}_\mu + \sum_{j=-K}^{K} (f - \mathbb{E}_\mu, \phi_j)\phi_j - f \right\|_{L^2}^2 d\mu(f) \\
&= Lip(\mathscr{G})^2 \sum_{|j|>K} \lambda_j^\mu \le Lip(\mathscr{G})^2 e^{-2\pi^2 l^2 K^2}.
\end{aligned} \tag{3.4}
$$

The bases $\{\widetilde{\tau}_j\}_{j=-K}^K$ in $\widetilde{\mathscr{R}}$ are not represented by neural networks. To obtain the desired operator $\mathscr{R}$ based on $\widetilde{\mathscr{R}}$, it is reasonable to approximate both $\mathbb{E}_\nu$ and $\{\widetilde{\tau}_j\}_{j=-K}^K$ using a set of neural networks. Specifically, we aim to approximate $\mathscr{G}(\mathbb{E}_\mu)$ and $\{\mathscr{G}(e^{i2\pi jx})\}_{j=-K}^K$, respectively. As $\mathbb{E}_\mu = 0$, it follows that $\mathbb{E}_\nu = 0$, allowing us to set $\tau^0 = 0$ as the neural network that approximates $\mathbb{E}_\nu$. We then demonstrate the existence of a set of neural networks, which can approximate $\{\widetilde{\tau}_j\}_{j=-K}^K$ with an error that does not exceed a given bound. This conclusion is supported by the following lemma. For a similar derivation, one can refer to the proof of [33, Lemma 1]. Therefore, the proof for this case is omitted for brevity.

Lemma 3.3. *Consider the Shishkin mesh, a piecewise uniform grid on* $[0,1]$*, defined by*

$$
\Omega = \left\{ x_j : x_j = jh,\ 0 \le j \le \frac{J}{2};\ x_j = 1 - \sigma + \left(j - \frac{J}{2}\right)H,\ \frac{J}{2} \le j \le J \right\}, \tag{3.5}
$$

where $h = 2(1-\sigma)/J$, $H = 2\sigma/J$, $\sigma = \min\{1/2, 2\varepsilon \ln J/\alpha\}$. *Considering a sufficiently smooth function* f*, we define the corresponding solution to Eq.* (1.1) *as* u*. Then for the piecewise linear interpolation function* $u_L(x)$ *on* $\{(x_i, u(x_i))\}_{i=0}^J$*, we have the error estimate*

$$
|u(x) - u_L(x)| \lesssim \begin{cases} \dfrac{1}{J^2}, & x \le 1-\sigma, \\ \dfrac{\ln^2 J}{J^2}, & x > 1-\sigma. \end{cases}
$$

Meanwhile,

$$
u_L = \sum_{n=1}^{J} \left(\big(u(x_n) - u(x_{n-1})\big) \frac{x - x_n}{x_n - x_{n-1}} + u(x_n) \right) 1_{[x_{n-1}, x_n]} = \sum_{n=0}^{J} u(x_n) \mathscr{L}_n(x),
$$

where $\mathscr{L}_n(x)$ is the piecewise linear nodal basis supported on the sub-interval $[x_{n-1}, x_{n+1})$, which can be represented by a neural network of depth 2 and width 3 as shown in [8], i.e.

$$
\begin{aligned}
\mathscr{L}_n(x) = &\frac{1}{x_n - x_{n-1}} \mathrm{ReLU}(x - x_{n-1}) - \left(\frac{1}{x_n - x_{n-1}} + \frac{1}{x_{n+1} - x_n} \right) \mathrm{ReLU}(x - x_n) \\
&+ \frac{1}{x_{n+1} - x_n} \mathrm{ReLU}(x - x_{n+1}).
\end{aligned}
$$

Thus, there exists a neural network τ of width $\mathscr{O}(J)$ and depth 2 such that

$$\|u-\tau\|_{L^\infty} \le \|u-u_L\|_{L^\infty} + \|u_L-\tau\|_{L^\infty} \lesssim \frac{\ln^2 J}{J^2}. \tag{3.6}$$

This inequality holds for all $u \in \{\mathscr{G}(e^{i2\pi jx})\}_{j=-K}^{K}$. By invoking the definition of $\widetilde{\tau}_j$, along with (3.6), it can be inferred that for a given $\delta > 0$ such that $\delta \gtrsim \ln^2 J/J^2$, neural networks τ^0 and $\{\tau_j\}_{j=-K}^{K}$ with a width of $\mathscr{O}(J)$, and a depth of 2, can be found to satisfy

$$\max\left\{\|\tau^0 - \mathbb{E}_\nu\|_{L^\infty}, \max_{j=-K,\dots,K}\|\tau_j - \widetilde{\tau}_j\|_{L^\infty}\right\} < \delta.$$

We define the reconstructor, induced by $\boldsymbol{\tau} = (\tau^0, \tau_{-K}, \tau_{-K+1}, \dots, \tau_K)$, as

$$\mathscr{R} : \mathscr{R}(a_{-K}, a_{-K+1}, \dots, a_K) = \tau^0 + \sum_{j=-K}^{K} a_j \tau_j,$$

which can be used to approximate the operator $\widetilde{\mathscr{R}}$. By combining this with the estimate (3.4), we obtain the following error estimate:

$$\begin{aligned}
\|\mathscr{R}\circ\mathscr{P} - Id\|_{L^2(\nu)} &\le \|\widetilde{\mathscr{R}}\circ\mathscr{P} - Id\|_{L^2(\nu)} + \|\mathscr{R}\circ\mathscr{P} - \widetilde{\mathscr{R}}\circ\mathscr{P}\|_{L^2(\nu)} \\
&\le Lip(\mathscr{G})e^{-\pi^2 l^2 K^2} + \left(\int_{L^2}\left\|\sum_{j=-K}^{K}(u,\widetilde{\tau}_j)(\tau_j-\widetilde{\tau}_j)\right\|_{L^2}^2 d\nu(u)\right)^{1/2} \\
&\le Lip(\mathscr{G})e^{-\pi^2 l^2 K^2} + pLip(\mathscr{G})\Big(\sum_{k\in\mathbb{Z}}\lambda_k^\mu\Big)^{1/2}\max_{j=-K,\dots,K}\|\tau_j-\widetilde{\tau}_j\|_{L^2} \\
&\le Lip(\mathscr{G})\big(e^{-\pi^2 l^2 K^2} + \delta p\big).
\end{aligned}$$

By assuming the Lipschitz continuity of $\mathscr{G}$, and introducing $\eta = p\delta$, we can summarize the above results in the following lemma.

Lemma 3.4. *There exists a set of neural networks $\boldsymbol{\tau} = (\tau^0, \tau_{-K}, \tau_{-K+1}, \dots, \tau_K)$ in the form of* (2.1) *with* $\text{width}(\boldsymbol{\tau}) = \mathscr{O}(\max\{p, (\eta/p)^{1/(\gamma-2)}\})$, $\text{depth}(\boldsymbol{\tau}) = 2$, *that satisfies*

$$\max\left\{\|\tau^0 - \mathbb{E}_\nu\|_{L^\infty}, \max_{j=-K,\dots,K}\|\tau_j - \widetilde{\tau}_j\|_{L^\infty}\right\} < \frac{\eta}{p},$$

where $\{\widetilde{\tau}_j\}_{j=-K}^{K}$ are obtained from $\{\mathscr{G}(e^{i2\pi jx})\}_{j=-K}^{K}$ by Gram-Schmidt orthogonalization, and $\gamma \in (0,1)$ is a fixed, arbitrarily small constant. This set of neural networks $\boldsymbol{\tau}$ can serve as a trunk net in a DeepONet. The reconstructor $\mathscr{R} : \mathbb{R}^p \to L^2([0,1])$ $(p = 2K+1)$ induced by $\boldsymbol{\tau}$ is defined as

$$\mathscr{R}(a_{-K}, a_{-K+1}, \dots, a_K) = \tau^0 + \sum_{j=-K}^{K} a_j \tau_j.$$

Then there exists an operator $\mathscr{P} : L^2([0,1]) \to \mathbb{R}^p$ such that the error term $\widehat{\mathscr{E}}_{\mathscr{R}}$ satisfies

$$\widehat{\mathscr{E}}_{\mathscr{R}} = \|\mathscr{R}\circ\mathscr{P} - Id\|_{L^2(\nu)} \lesssim e^{-\pi^2 l^2 K^2} + \eta.$$

The operator $\mathscr{P}$ (constructed in Lemma 3.4) and the operator $\mathscr{D}$ (constructed in Lemma 3.2) are linear mappings. Hence, $\mathscr{P} \circ \mathscr{G} \circ \mathscr{D}$ is a linear mapping from $\mathbb{R}^m$ to $\mathbb{R}^p$, then there exists a matrix A such that $\mathscr{P} \circ \mathscr{G} \circ \mathscr{D} = A$. The network approximator $\mathscr{A}$, as an approximation of $\mathscr{P} \circ \mathscr{G} \circ \mathscr{D}$, can be taken as $\mathscr{A} : \mathbb{R}^m \to \mathbb{R}^p$, $\boldsymbol{x} \mapsto A\boldsymbol{x}$ or $\boldsymbol{x} \mapsto \mathrm{ReLU}(A\boldsymbol{x}) - \mathrm{ReLU}(-A\boldsymbol{x})$, that is, with respect to the error term $\widehat{\mathscr{E}}_{\mathscr{A}}$, we can easily obtain the following result:

Lemma 3.5. *There exists a neural network* $\mathscr{A}$ *of* $\mathrm{width}(\mathscr{A}) = \mathscr{O}(\max\{m, p\})$, $\mathrm{depth}(\mathscr{A}) = 2$, *such that*

$$\widehat{\mathscr{E}}_{\mathscr{A}} = \left(\int_{\mathbb{R}^m} \| \mathscr{A}(\boldsymbol{x}) - \mathscr{P} \circ \mathscr{G} \circ \mathscr{D}(\boldsymbol{x}) \|^2_{l^2(\mathbb{R}^p)} d(\mathscr{E}_{\#}\mu)(\boldsymbol{x}) \right)^{1/2} = 0.$$

We note that the Lipschitz norm of $\mathscr{R} \circ \mathscr{P}$ is bounded by 2. This observation, coupled with the fact that the Branch net in DeepONet corresponds to $\mathscr{A} \circ \mathscr{E}$, allows us to rigorously establish the validity of Theorem 3.1. The proof is completed through a careful combination of several lemmas, namely, Lemma 3.1, Lemmas 3.2 and 3.4, and finally, Lemma 3.5.

4. Generalization Capability

Acquiring the solution operator $\mathscr{G}$ for partial differential equations (PDEs) directly proves challenging in most cases, rendering the computation of the error term (3.1) equally difficult. Therefore, a proxy measure is necessary to quantify the difference between the operator network and operator $\mathscr{G}$. Numerical techniques are thus implemented to discretize the integration and estimate the error (3.1). For example, the Monte Carlo method is commonly used to approximate the integration over $L^2([0,1])$, while both stochastic and deterministic methods, such as the trapezoidal rule, can be utilized for integration over $[0,1]$.

For problems with a solution that has a bounded gradient, locations can be randomly selected from the domain $[0,1]$ or taken as equidistant points. However, for singularly perturbed problems like (1.1) with relatively small parameter ε, the solution exhibits an exponential boundary layer near $x = 1$. Within this region, the solution's gradient is relatively large, and if the aforementioned sampling methods are utilized, a sufficiently large number of points will be required to capture the boundary layer information accurately. Moreover, in these cases, the discrete form error may depend on $1/\varepsilon$, which is undesirable.

Utilizing the right Riemann sum on a Shishkin mesh [13] allows us to approximate the original integral by partitioning the interval into subintervals with varying widths. Specifically, the Shishkin mesh employs a dense clustering of points near the boundary layer, where the solution changes rapidly, to ensure accurate approximation in this critical region. This technique has been shown to effectively capture the singular layer behavior and has been widely adopted in numerical simulations and modeling.

We consider N samples $F_1, F_2, \ldots, F_N$ drawn independently and identically from a common distribution μ, and we let $\{y_j\}_{j=0}^J$ denote the Shishkin mesh points on the interval $[0,1]$, as defined in (3.5). We approximate two integrals in the error term (3.1) using

Monte Carlo integration and numerical integration, respectively. Then the empirical risk $\mathscr{E}_{N,J}$ can be derived, which takes the following form:

$$\mathscr{E}_{N,J} = \left(\frac{1}{N} \sum_{n=1}^{N} \sum_{j=0}^{J} \omega_j |\mathscr{G}(F_n)(y_j) - \mathscr{N}(F_n)(y_j)|^2 \right)^{1/2}, \tag{4.1}$$

where the weights ω_i are determined as follows: $\omega_0 = 0$, $\omega_i = h$ for $1 \le i \le J/2$, and $\omega_i = H$ for $J/2 < i \le J$, with h and H defined in Lemma 3.3.

4.1. Main results

Ensuring that a machine learning model can generalize well to new, unseen data beyond the training dataset is crucial. In this context, we focus on assessing the generalization ability of our machine learning model, denoted by $\widehat{\mathscr{N}}$, which is learned via a given learning algorithm. To measure the model's generalization performance, we use the concept of the generalization gap (see [11]), which quantifies the difference between the approximation error and the empirical risk on $\widehat{\mathscr{N}}$

$$\mathscr{E}_{gen} = \mathbb{E}\left| \widehat{\mathscr{E}}(\widehat{\mathscr{N}}) - \mathscr{E}_{N,J}(\widehat{\mathscr{N}}) \right| \triangleq \mathscr{E}_{diff}(\widehat{\mathscr{N}}). \tag{4.2}$$

To control this gap, we first introduce the following assumptions.

Assumption 4.1. There exists $q \in [4, +\infty)$ such that $\mathbb{E}_{f\sim\mu} \|\mathscr{G}(f) - \mathscr{N}(f)\|_{L^2}^q \lesssim 1$.

Assumption 4.2. There exists an operator $\widetilde{\mathscr{N}} : L^2([0,1]) \to C^1([0,1])$ such that for any $x \in [0,1]$,

$$\mathbb{E}_{f\sim\mu} \left| \mathscr{N}(f)(x) - \widetilde{\mathscr{N}}(f)(x) \right|^2 \lesssim 1/N^{1/2}, \tag{4.3}$$

$$\mathbb{E}_{f\sim\mu} \left| \frac{d}{dx} \left(\mathscr{G}(f)(x) - \widetilde{\mathscr{N}}(f)(x) \right)^2 \right| \lesssim 1 + \varepsilon^{-1} e^{-\alpha(1-x)/\varepsilon}. \tag{4.4}$$

Remark 4.1. Activation functions like ReLU are not differentiable everywhere, which means that the derivative of $\mathscr{N}(f)(x)$ may be undefined at some points, and present challenges for error estimation. To overcome this issue, we introduce the operator $\widetilde{\mathscr{N}}$, which need not be an operator network, and put forth Assumption 4.2. The incorporation of Assumption 4.1 permits the error estimation for Monte Carlo integration, while the inclusion of Assumption 4.2 enables the error estimation for deterministic numerical integration. The reasonableness of these assumptions can be briefly explained: when $\mathbb{E}_{f\sim\mu} \|f\|_{L^2} \lesssim 1$, it follows that $\mathbb{E}_{f\sim\mu} \|\mathscr{G}(f)\|_{L^2} \lesssim 1$ and for any $x \in [0,1]$,

$$\mathbb{E}_{f\sim\mu} \left| \frac{d}{dx} \mathscr{G}(f)^2 \right| \lesssim 1 + \varepsilon^{-1} e^{-\alpha(1-x)/\varepsilon}.$$

Furthermore, since the operator network $\mathscr{N}$ is an approximation of the target operator $\mathscr{G}$, it is reasonable to make similar assumptions for $\mathscr{G} - \mathscr{N}$. Notably, the operator network constructed in Theorem 3.1 and Theorem 3.2 satisfy these assumptions.

These assumptions allow us to obtain an estimate for the generalization gap, which will be rigorously proved in the next subsection.

Theorem 4.1. *Under Assumptions* 4.1 *and* 4.2*, and assuming the Lipschitz continuity of* $\mathscr{G}$*, a bound for the generalization gap* (4.2) *can be established as follows:*

$$\mathscr{E}_{gen} = \mathbb{E}\left|\widehat{\mathscr{E}}(\widehat{\mathscr{N}}) - \mathscr{E}_{N,J}(\widehat{\mathscr{N}})\right| \lesssim \frac{1}{N^{1/4}} + \frac{\sqrt{\ln J}}{J^{1/2}}. \tag{4.5}$$

Here, N *represents the number of* f *samples, and* J *denotes the number of locations on the interval* $[0,1]$.

Remark 4.2. Both error expressions (3.1) and (4.1) involve a square root symbol on the outside. Therefore, the $1/N^{1/4}$ term in Eq. (4.5) corresponds to the standard Monte Carlo error rate, while $\sqrt{\ln J}/J^{1/2}$ corresponds to the error of the numerical integration method. Theorem 4.1 tells us that increasing the number of sampling points can reduce the generalization gap.

As noted previously, the true error of our operator network $\mathscr{N}$ in approximating $\mathscr{G}$ is unavailable. Instead, we rely on the empirical error as a proxy for assessing the quality of the approximation. Therefore, analyzing this empirical error is crucial. We expect the existence of an operator network in the form of (2.2), such that the corresponding empirical risk exhibits convergence properties with various factors that affect the performance of the neural network, including the dimensions m, p of the input and output spaces, and the number of sample points N, J. Importantly, we hope this error is independent of the negative power of the parameter ε; otherwise, if ε is too small, the corresponding error will become unacceptably large, rendering the operator network unusable for approximating the solution operator of Eq. (1.1). To address this issue, we present the following result, with proof provided in the next subsection.

Theorem 4.2. *Assuming Lipschitz continuity of operator* $\mathscr{G}$ *and* $m \leq p$*, we can construct a ReLU branch network* $\mathscr{A}$ *and a ReLU trunk network* $\boldsymbol{\tau}$*, with* $\text{size}(\mathscr{A}) \lesssim p$ *and* $\text{size}(\boldsymbol{\tau}) \lesssim p^{(n+1)/(2-\gamma)}$ *respectively, for an arbitrarily small constant* $\gamma \in (0,1)$ *and arbitrary constant* $n > 0$*. This results in the corresponding operator network* $\mathscr{N}$ *satisfying the given estimate*

$$\mathbb{E}|\mathscr{E}_{N,J}(\mathscr{N})| \lesssim \frac{\widehat{\mathscr{E}}(\mathscr{N})}{N^{1/4}} + p^{-n} + \frac{C\sqrt{\ln J}}{mJ^{1/2}}.$$

Here, N *and* J *represent the number of* f *samples and locations, respectively,* m *refers to the number of input layer neurons in the branch network, and* C *is a constant related only to* α *and* l*. Additionally,* $\widehat{\mathscr{E}}(\mathscr{N})$ *is provided in Theorem* 3.2.

4.2. Proof overview

This section contains the proofs of Theorems 4.1 and 4.2. To simplify notations, we consider new operators $\Delta_{\mathscr{N}}$ and $\mathscr{I}$ defined by

$$\Delta_{\mathscr{N}}(f)(y) := |\mathscr{G}(f)(y) - \mathscr{N}(f)(y)|^2, \quad \mathscr{I}(g) := \sum_{j=0}^{J} \omega_j g(y_j).$$

Assuming that the operator $\mathscr{N}$ satisfies Assumption 4.2, we can infer the existence of an operator $\widetilde{\mathscr{N}}$ that satisfies inequalities (4.3) and (4.4). Using this, we can control the difference $\mathscr{E}_{diff}(\mathscr{N})$ with the following four terms:

$$\begin{aligned}
\mathscr{E}_{diff}(\mathscr{N}) &= \mathbb{E}\left|\widehat{\mathscr{E}}(\mathscr{N}) - \mathscr{E}_{N,J}(\mathscr{N})\right| \\
&\leq \mathbb{E}\left|\left(\mathbb{E}_{f\sim\mu}\|\Delta_{\mathscr{N}}(f)\|_{L^1}\right)^{1/2} - \left(\frac{1}{N}\sum_{n=1}^{N}\|\Delta_{\mathscr{N}}(F_n)\|_{L^1}\right)^{1/2}\right| \\
&\quad + \mathbb{E}\left|\left(\frac{1}{N}\sum_{n=1}^{N}\|\Delta_{\mathscr{N}}(F_n)\|_{L^1}\right)^{1/2} - \left(\frac{1}{N}\sum_{n=1}^{N}\|\Delta_{\widetilde{\mathscr{N}}}(F_n)\|_{L^1}\right)^{1/2}\right| \\
&\quad + \mathbb{E}\left|\left(\frac{1}{N}\sum_{n=1}^{N}\|\Delta_{\widetilde{\mathscr{N}}}(F_n)\|_{L^1}\right)^{1/2} - \left(\frac{1}{N}\sum_{n=1}^{N}\mathscr{I}\big(\Delta_{\widetilde{\mathscr{N}}}(F_n)\big)\right)^{1/2}\right| \\
&\quad + \mathbb{E}\left|\left(\frac{1}{N}\sum_{n=1}^{N}\mathscr{I}\big(\Delta_{\widetilde{\mathscr{N}}}(F_n)\big)\right)^{1/2} - \left(\frac{1}{N}\sum_{n=1}^{N}\mathscr{I}\big(\Delta_{\mathscr{N}}(F_n)\big)\right)^{1/2}\right| \\
&\triangleq \mathrm{I} + \mathrm{II} + \mathrm{III} + \mathrm{IV}.
\end{aligned} \tag{4.6}$$

This inequality will serve as the basis for proving Theorems 4.1 and 4.2.

4.2.1. Proof of Theorem 4.1

By leveraging this observation, we can now establish a rigorous proof for Theorem 4.1. Let us start with the estimate of the term I in (4.6).

Lemma 4.1. *If $\mathscr{G}$ is the solution operator of the Eq.* (1.1) *and the operator $\mathscr{N}$ satisfies Assumption* 4.1*, then*

$$\mathrm{I} \lesssim \frac{1}{N^{1/4}}.$$

Proof. Let $X_n = \|\mathscr{G}(F_n) - \mathscr{N}(F_n)\|_{L^2}^2$ and $S^N = (\sum_{n=1}^{N} X_n)/N$, we can get

$$\begin{aligned}
\mathrm{I} &= \mathbb{E}\left|\sqrt{S^N} - \sqrt{\mathbb{E}S^N}\right| \leq \mathbb{E}\sqrt{|S^N - \mathbb{E}S^N|} \\
&\leq \left(\mathbb{E}|S^N - \mathbb{E}(S^N)|^2\right)^{1/4} = \frac{1}{\sqrt{N}}\left(\mathbb{E}\left(\sum_{n=1}^{N}(X_N - \mathbb{E}X_N)\right)^2\right)^{1/4} \\
&= \frac{1}{\sqrt{N}}\left(\mathbb{E}\left(\sum_{n=1}^{N}(X_n - \mathbb{E}X_n)^2 + \sum_{m\neq n}(X_n - \mathbb{E}X_n)(X_m - \mathbb{E}X_m)\right)\right)^{1/4} \\
&= \frac{1}{\sqrt{N}}\left(\sum_{n=1}^{N}\mathbb{E}(X_n - \mathbb{E}X_n)^2\right)^{1/4} = \frac{1}{N^{1/4}}\left(\mathbb{E}(X_1 - \mathbb{E}X_1)^2\right)^{1/4}
\end{aligned}$$

$$\leq \frac{1}{N^{1/4}}\left((\mathbb{E}X_1^2)^{1/4}+(\mathbb{E}X_1)^{1/2}\right). \tag{4.7}$$

The boundedness of $\mathbb{E}X_1^2$, as expressed in Assumption 4.1, has already been established. Meanwhile, the result of Theorem 3.2 directly yields the boundedness of the other term, $\mathbb{E}X_1$, then we can directly obtain the desired conclusion from the above inequality (4.7). □

Exploiting the relationship captured by inequality (4.3) between the operators $\widetilde{\mathscr{N}}$ and $\mathscr{N}$, we can arrive at a useful estimate by simply manipulating terms II and IV in Eq. (4.6)

$$\begin{aligned}\mathrm{II}+\mathrm{IV} &\leq \left(\mathbb{E}_{f\sim\mu}\|\mathscr{N}(f)-\widetilde{\mathscr{N}}(f)\|_{L^2}^2\right)^{1/2}+\left(\sum_{j=0}^{J}\omega_j\mathbb{E}_{f\sim\mu}|\mathscr{N}(f)(y_j)-\widetilde{\mathscr{N}}(f)(y_j)|^2\right)^{1/2}\\ &\lesssim \frac{1}{N^{1/4}}.\end{aligned} \tag{4.8}$$

The error denoted by III in (4.6) arises from employing a numerical integration method to estimate the integral over the interval $[0,1]$. This can be bounded by

$$\begin{aligned}\mathrm{III} &= \mathbb{E}\left|\left(\frac{1}{N}\sum_{n=1}^{N}\|\mathscr{G}(F_n)-\widetilde{\mathscr{N}}(F_n)\|_{L^2}^2\right)^{1/2}-\left(\frac{1}{N}\sum_{n=1}^{N}\sum_{j=0}^{J}\omega_j|\mathscr{G}(F_n)(y_j)-\widetilde{\mathscr{N}}(F_n)(y_j)|^2\right)^{1/2}\right|\\ &\leq \left(\mathbb{E}\left|\frac{1}{N}\sum_{n=1}^{N}\left(\|\mathscr{G}(F_n)-\widetilde{\mathscr{N}}(F_n)\|_{L^2}^2-\sum_{j=0}^{J}\omega_j|\mathscr{G}(F_n)(y_j)-\widetilde{\mathscr{N}}(F_n)(y_j)|^2\right)\right|\right)^{1/2}\\ &\leq \left(\mathbb{E}_{f\sim\mu}\left|\|\mathscr{G}(f)-\widetilde{\mathscr{N}}(f)\|_{L^2}^2-\sum_{j=0}^{J}\omega_j|\mathscr{G}(f)(y_j)-\widetilde{\mathscr{N}}(f)(y_j)|^2\right|\right)^{1/2}.\end{aligned} \tag{4.9}$$

To compute this error term, we present a lemma in this regard, and the proof of this lemma is outlined in Appendix A.1.

Lemma 4.2. *Suppose $F: L^2([0,1])\to C^1([0,1])$ is an operator such that*

$$\mathbb{E}_{f\sim\mu}\left|\frac{d}{dx}F(f)(x)\right|\leq C\left(1+\varepsilon^{-1}e^{-\alpha(1-x)/\varepsilon}\right),\quad x\in[0,1]$$

with a constant $\alpha>0$. Assuming a piecewise uniform mesh with $(J+1)$ points given by (3.5) *as $\{x_j\}_{j=0}^{J}$, we have*

$$\mathbb{E}_{f\sim\mu}\left|\int_0^1 F(f)(x)dx-\sum_{j=0}^{J-1}h_jF(f)(x_j)\right|\leq 2C\left(1+\frac{2}{\alpha}\right)\frac{\ln J}{J},$$

where $h_j=x_{j+1}-x_j$.

By setting $F = (\mathscr{G} - \widetilde{\mathscr{N}})^2$ in Lemma 4.2, we can directly obtain an estimate for term III using the conclusion of this lemma, given that inequality (4.4) is satisfied. Specifically, we have

$$\text{III} \lesssim \left(\frac{\ln J}{J}\right)^{1/2}. \tag{4.10}$$

Combining Lemma 4.1 with formulas (4.8), (4.10), we arrive at the conclusion of Theorem 4.1.

4.2.2. Proof of Theorem 4.2

In Section 3, we construct operators in the form of (2.2) and estimate the error incurred by these operators in approximating the target operator $\mathscr{G}$. Of particular interest is the operator defined as

$$\overline{\mathscr{N}}(f) = \mathbb{E}_\nu + \sum_{j=-K}^{K} \big(\mathscr{G} \circ \mathscr{D} \circ \mathscr{E}(f), \widetilde{\tau}_j\big)\tau_j, \tag{4.11}$$

where operator $\mathscr{D}$ is detailed in Remark 3.2, and $\widetilde{\tau}_j$, $j = -K, \ldots, K$ is obtained by orthogonalizing the set of functions $\{\mathscr{G}(\phi_j) = \mathscr{G}(e^{i2\pi kx})\}_{j=-K}^{K}$. Additionally, $\{\tau^0, \{\tau_j\}_{j=-K}^{K}\}$ is a set of neural networks that satisfies the condition

$$\max\left\{\|\tau^0 - \mathbb{E}_\nu\|_{L^\infty}, \max_{j=-K,\ldots,K} \|\tau_j - \widetilde{\tau}_j\|_{L^\infty}\right\} < \delta = p^{-(n+1)},$$

where n is a fixed, arbitrary positive constant.

In the following, we set out to investigate the empirical risk $\mathscr{E}_{N,J}(\overline{\mathscr{N}})$ associated with the operator at hand, and provide proof of Theorem 4.2. While Theorem 3.2 has previously addressed the estimation of $\widehat{\mathscr{E}}(\overline{\mathscr{N}})$, our present focus centers on analyzing $\mathscr{E}_{diff}(\overline{\mathscr{N}})$. Considering that the activation function of $\overline{\mathscr{N}}$ is not always differentiable, we introduce a modified operator $\widetilde{\mathscr{N}}$, which is obtained from $\overline{\mathscr{N}}$ by replacing the bases function. Specifically, $\widetilde{\mathscr{N}}(f)$ is defined as

$$\widetilde{\mathscr{N}}(f) = \tau^0 + \sum_{j=-K}^{K} \big(\mathscr{G} \circ \mathscr{D} \circ \mathscr{E}(f), \widetilde{\tau}_j\big)\widetilde{\tau}_j. \tag{4.12}$$

Substituting $\widetilde{\mathscr{N}}$ into Eq. (4.6), we can control $\mathscr{E}_{diff}(\overline{\mathscr{N}})$ via the four terms associated with $\overline{\mathscr{N}}$ and $\widetilde{\mathscr{N}}$, still denoted as I, II, III, and IV. The estimation of terms I, II, and IV is straightforward, and we present the results in Lemma 4.3 (with the proof given in Appendix A.2).

Lemma 4.3. *Assuming $m \le p$, we can consider the operators $\overline{\mathscr{N}}$ and $\widetilde{\mathscr{N}}$, as defined in Eqs.* (4.11) *and* (4.12)*, respectively. For the terms* I, II*, and* IV *in Eq.* (4.6)*, an estimate for their sum can be derived as follows:*

$$\text{I} + \text{II} + \text{IV} \lesssim \frac{\widehat{\mathscr{E}}(\overline{\mathscr{N}})}{N^{1/4}} + p^{-n},$$

where $\widehat{\mathscr{E}}(\overline{\mathscr{N}})$ is estimated using Theorem 3.2*.*

We now focus on the term III, which can be bounded using inequality (4.9). We define $F(f)=(\mathscr{G}(f)-\widetilde{\mathscr{N}}(f))^2$, and according to Lemma 4.2, estimating term III requires only an estimation of $\mathbb{E}_{f\sim\mu}|(d/dx)F(f)(x)|$. We focus on this quantity next. In fact, when $M\le K$ (that is, when $m\le p$), we have the following simple form for $\mathscr{G}(f)-\widetilde{\mathscr{N}}(f)$:

$$\mathscr{G}(f)(x)-\widetilde{\mathscr{N}}(f)(x)=\mathscr{G}(f)(x)-\mathscr{G}\circ\mathscr{D}\circ\mathscr{E}(f)(x). \tag{4.13}$$

Here, we have used the fact that when $m\le p$,

$$\widetilde{\mathscr{N}}(f)=\sum_{j=-K}^{K}\big(\mathscr{G}\circ\mathscr{D}\circ\mathscr{E}(f),\widetilde{\tau}_j\big)\widetilde{\tau}_j=\mathscr{G}\circ\mathscr{D}\circ\mathscr{E}(f).$$

With this observation, we obtain the estimate for $\mathbb{E}_{f\sim\mu}|(d/dx)F(f)(x)|$ using the following lemma (proof provided in Appendix A.3).

Lemma 4.4. *When $m\le p$, the following inequality holds:*

$$\mathbb{E}_{f\sim\mu}\left|\big(\mathscr{G}(f)(x)-\widetilde{\mathscr{N}}(f)(x)\big)\frac{d}{dx}\big(\mathscr{G}(f)(x)-\widetilde{\mathscr{N}}(f)(x)\big)\right|\le\frac{C}{m^2}\left(1+\frac{1}{\varepsilon e^{-\alpha(1-x)/\varepsilon}}\right).$$

Employing Lemmas 4.2 and 4.4, we derive an upper bound for the term III in the Eq. (4.6), viz.

$$\mathrm{III}\lesssim\frac{\sqrt{\ln J}}{mJ^{1/2}}.$$

Combining this estimate with Lemma 4.3 enables us to complete the proof of Theorem 4.2.

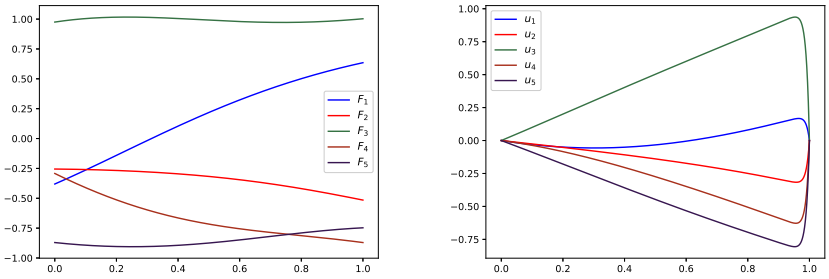
5. Numerical Experiments

In this section, we demonstrate the effectiveness of DeepONets in solving singularly perturbed problems through three concrete examples, thereby contributing to the growing body of theoretical understanding of this model. The model's trainable parameters in Eq. (2.2), denoted by $\boldsymbol{\theta}$, are obtained through the minimization of a loss function defined as follows:

$$\mathscr{L}(\boldsymbol{\theta})=\frac{1}{N(J+1)}\sum_{n=1}^{N}\sum_{j=0}^{J}|u_n(y_j)-\mathscr{N}_{\boldsymbol{\theta}}(F_n)(y_j)|^2. \tag{5.1}$$

Here, u_n represents the ground truth, which is typically obtained through numerical methods of high precision. To obtain $u_n(y_j)$, we utilize an up-winding scheme on the Shishkin mesh — cf. [31], and interpolate accordingly.

N samples, denoted $F_1,F_2,\dots,F_N$, are drawn independently and identically from the common distribution μ. Examples of several pairs (F_i,u_i) are provided in Fig. 2. In general, locations $\{y_i\}$ can be selected randomly from the domain $[0,1]$ or arranged as equidistant points. However, as noted previously, due to the presence of a thin boundary layer, the use of these sampling methods would demand a large number of points to ensure that the



(a) (b)

Figure 2: (a) Examples of random process $F_i \sim \mu$ with parameter $l = 1$, generating by matrix decomposition method [18]. (b) Numerical solutions to Eq. (1.1) with $p(x) = x + 1$, $q(x) = 1$, $\varepsilon = 0.01$, and right-hand side $f = F_i$ as presented in (a).

trained operator network can capture the relevant boundary layer information accurately. Conversely, sampling a large number of points far away from the boundary layer is unnecessary since the solution's behavior in this region is regular and slow. Consequently, we adopt Shishkin mesh points, defined in (3.5), as the locations in (5.1) within the interval $[0,1]$.

Remark 5.1. Although machine learning models are commonly obtained through empirical risk minimization, where $\boldsymbol{\theta}$ is chosen to minimize empirical risk, and an operator network is determined accordingly. Notably, the solution to the singularly perturbed problem (1.1) can vary across different length scales. To address this issue, the loss function (5.1) assigns all weights ω_j, where $j = 0, 1, \ldots, J$ in (4.1), a uniform value of $1/(J+1)$. This rescales the microscale and places data at different scales on the same scale. This approach is expected to mitigate the spectral bias phenomenon. The numerical results presented in the following sections confirm this conjecture.

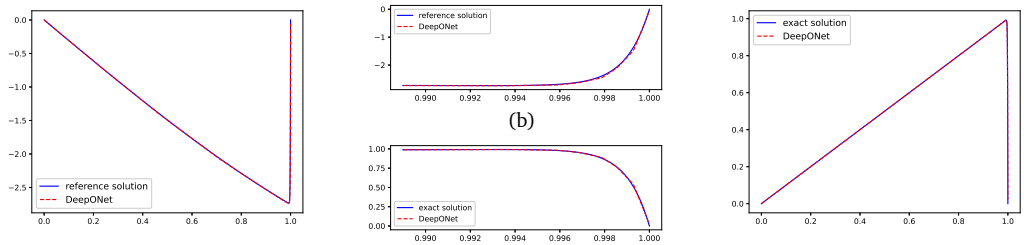
Throughout the following examples, we set the positive lower bound α of the function $p(x)$ to 1. Then the transition point of the Shishkin mesh is $\sigma = \min\{1/2,\ 2\varepsilon \ln J\}$.

Example 5.1. We consider the following boundary layer problem — cf. [31]:

$$\begin{aligned} -\varepsilon u'' + u' &= f, \quad x \in (0,1), \\ u(0) &= u(1) = 0. \end{aligned} \tag{5.2}$$

The Lipschitz continuity of the solution operator that maps $f(x)$ to the ODE solution $u(x)$ is proved in Appendix A.4. Subsequently, we utilize DeepONets to learn this operator. To generate the necessary training dataset, we solve the equation using an up-winding scheme on a Shishkin mesh of 4096 points, and obtain $\{u(y_j)\}_{j=0}^{J}$ via interpolation.

We begin by training a DeepONet using 1000 random f samples and 256 y locations (i.e., $N = 1000$ and $J = 256$), resulting in a training dataset of size 2.56×10^5. The training is conducted over 1000 epochs. Fig. 3 illustrates the predictive capability of the trained model, displaying its ability to predict solutions for two distinct f that lie beyond



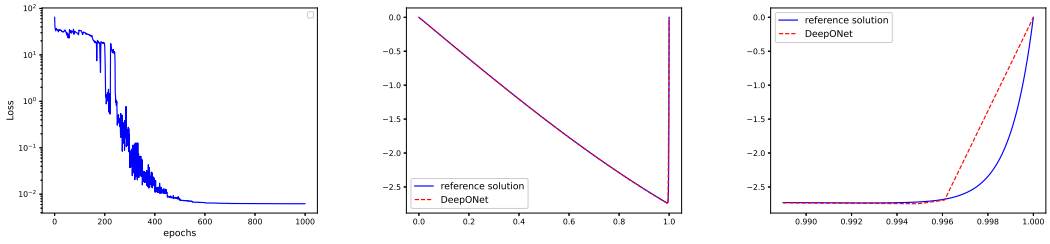
(a) (b) (c) (d)

Figure 3: The trained model's predictions and reference solutions for Eq. (5.2) ($\varepsilon = 0.001$) on Shishkin mesh. (a) Prediction and reference solution on a random $f \sim \mu$ on a grid of 256 points, with the latter using an up-winding scheme on Shishkin mesh. (b) Horizontal magnification of the region $[1-\sigma, 1]$ in (a). (c) Horizontal magnification of the region $[1-\sigma, 1]$ in (d). (d) the model's prediction and exact solution for $f(x) \equiv 1$ on a grid of 256 points, the exact solution is given by $u(x) = x - (e^{-(1-x)/\varepsilon} - e^{-1/\varepsilon})/(1-e^{-1/\varepsilon})$.

our training dataset. The first is randomly sampled from μ with the parameter $l = 1$, while the second is a simple out-of-distribution signal given by $f(x) \equiv 1$.

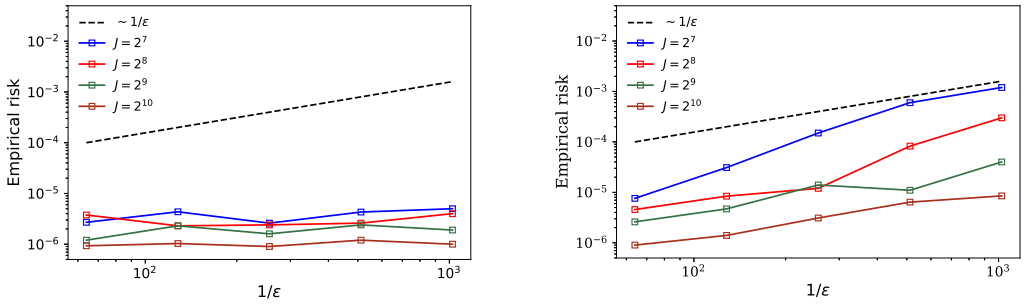
As previously noted, accurate capture of boundary layer information using a trained model necessitates taking a sufficient number of data points, particularly when sampling at equidistant locations. To validate this conjecture, we conducted experimental tests. To begin with, we train a DeepONet leveraging 1000 random f samples and 256 equidistant locations y. In Fig. 4, subplot (a) displays the trajectory of the training process. Subplots (b) and (c) indicate that training the network using equidistant locations is effective in capturing information beyond the boundary layer; however, it is not as effective in dealing with behavior within the boundary layer.

By utilizing the definition of the generalization gap, empirical risk can serve as an effective tool for assessing a model's generalization ability. A comparison of Figs. 3 and 4 reveals that the network trained on equidistant locations is inferior to the network trained on Shishkin mesh points in describing the boundary layer behavior. Fig. 5 provides a more



(a) (b) (c)

Figure 4: Performance of a trained model using equidistant location points ($\varepsilon = 0.001$). (a) The training trajectory of a DeepONet with equidistant location points. Here "epochs" refers to the number of steps that the Adam optimizer performs. (b) Prediction using the trained model and reference solution on a random $f \sim \mu$ on a Shishkin mesh of 256 points. (c) Horizontal magnification of the region $[1-\sigma, 1]$ in (b).



(a) (b)

Figure 5: Models trained on a different number of Shishkin mesh points and equidistant points are compared in terms of empirical risk, evaluated on a 2^{12} location point mesh. The subplots display empirical risk against the parameter (ε) on a $1/\varepsilon$ scale, with the dashed black line running parallel to $1/\varepsilon$. (a) The empirical risk performance of models trained on different numbers of Shishkin mesh points. (b) The performance of models trained on different numbers of equidistant points.

comprehensive analysis, using both 1000 f samples. The comparison between subplots (a) and (b) in Fig. 5 shows that the network trained on Shishkin mesh points exhibits consistent generalization ability across various values of ε. Conversely, the network trained on equidistant points demonstrates a significant correlation with $1/\varepsilon$, indicating a reduction in its generalization ability as ε decreases. Moreover, these figures demonstrate that the model's generalization ability improves with the increase in the number of locations y, i.e. J.

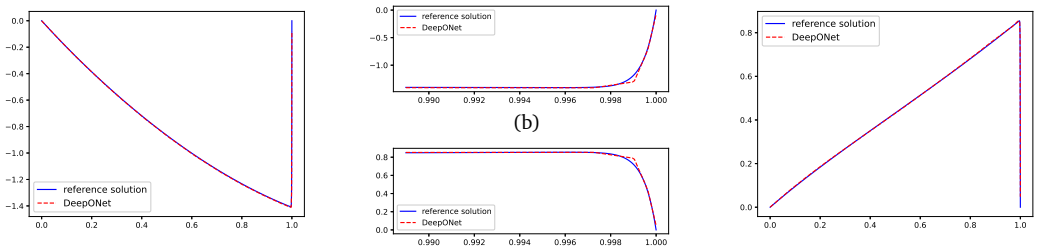
Example 5.2. Consider a singularly perturbed problem with variable coefficients given by

$$\begin{aligned} -\varepsilon u'' + (x+1)u' + u &= f, \quad x \in (0,1), \\ u(0) = u(1) &= 0. \end{aligned} \tag{5.3}$$

The solution operator mapping from f to the solution u is known to be Lipshitz, according to Lemma 2.2. We use DeepONets to approximate this operator. We construct a DeepONet by training it on a dataset comprising 2.56×10^5 triplets (f, y, u) over 1000 epochs. The input data consists of 1000 randomly sampled f and 256 distinct locations y (specifically, a Shishkin mesh of 256 points), while the corresponding output data, u, is obtained using an up-wind scheme on the Shishkin mesh. Fig. 6 demonstrates the network's ability to accurately estimate the output solution u, not only for a randomly sampled input f but also for an out-of-distribution input $f(x) = e^x$.

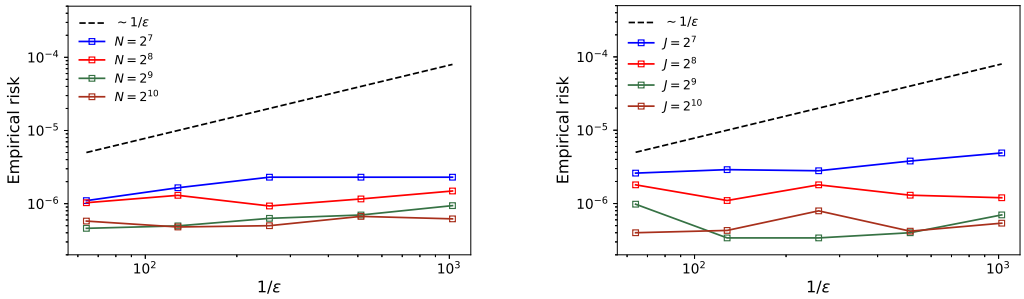
Based on Theorem 4.2, it can be observed that the empirical loss showcases a negative correlation with the quantity of f samples as well as the number of Shishkin locations y. Importantly, this relationship holds irrespective of $1/\varepsilon$. Subsequently, we endeavor to establish the validity of this relationship for Eq. (5.3). The results obtained from pertinent experiments are illustrated in Fig. 7.

Figs. 3 and 6 illustrate DeepONets' effectiveness in capturing the boundary layer behavior near $x = 1$ and performing remarkably accurate out-of-distribution predictions. These



(a) (b) (c) (d)

Figure 6: The trained model's predictions and reference solutions for Eq. (5.3) ($\varepsilon = 0.001$) on Shishkin mesh. (a) Prediction and reference solution on a random $f \sim \mu$ on a grid of 256 points, with the latter using an up-winding scheme on Shishkin mesh. (b) Horizontal magnification of the region $[1-\sigma, 1]$ in (a). (c) Horizontal magnification of the region $[1-\sigma, 1]$ in (d). (d) the model's prediction and reference solution for $f(x) = e^x$ on a grid of 256 points.



(a) (b)

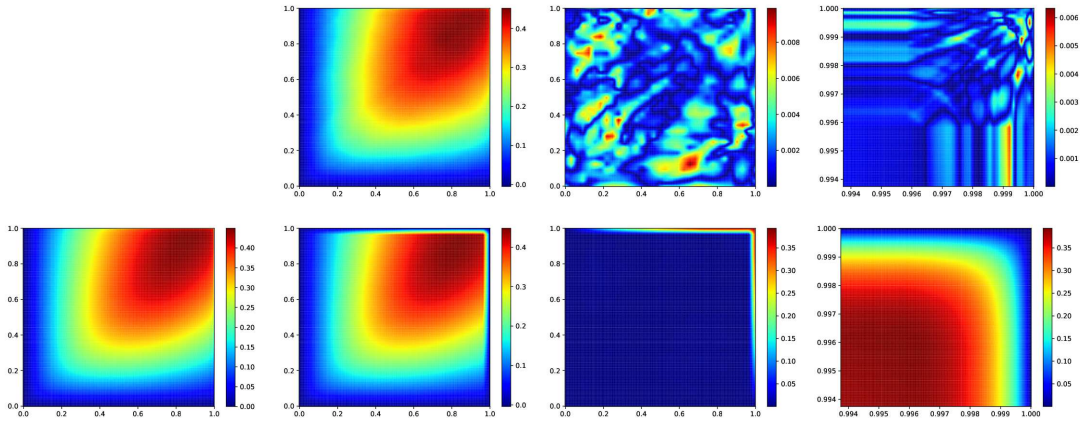
Figure 7: The generalization ability of trained models with varying factors. (a) Impact of varying f sample numbers on model generalization with a fixed set of 512 Shishkin locations. Empirical risk is calculated over a sample of 2^{21} pairs (f, y) consisting of 4096 randomly selected f samples and 512 locations y. (b) Impact of different numbers of locations y on model generalization with a fixed set of 1000 f samples. Empirical risk was calculated over 1000×2^{12} sample pairs (f, y).

experimental findings suggest that DeepONets can serve as a reliable approximator for approximating the solution operator of one-dimensional singular perturbation problems. As we extend our focus to the two-dimensional problem, it is important to note that theoretical results currently exist only for the one-dimensional problem. However, we expect that DeepONet is also suitable for solving the two-dimensional singular perturbation problem, and subsequent numerical results validate this conjecture.

Example 5.3. Consider the 2D singularly perturbed problem

$$\begin{aligned} -\varepsilon \Delta u + u_x + u_y + u &= f, \quad x \in \Omega = (0,1) \times (0,1), \\ u|_{\partial\Omega} &= 0. \end{aligned} \tag{5.4}$$

In this study, we employ DeepONets to learn the solution operator for this problem with $\varepsilon = 0.001$, i.e. to map f to the solution u of Eq. (5.4). As the dimensionality of the problem increases, the size of the training data set needs to be significantly increased due to the



(a) (b) (c) (d)

Figure 8: Comparison of performance between models trained on Shishkin and equidistant locations. (a) The reference solution for a Gaussian random field $f \sim \mathcal{G}(0, k_{RBF})$ achieved through an up-winding scheme on the Shishkin mesh. (b) The predictions of the trained models on f, where the upper subfigure displays the prediction of the model trained on the Shishkin positions and the lower subfigure shows the prediction of the model trained on the equidistant positions. (c) The errors between the predicted results of the trained models and the reference solution. (d) An enlargement of the $[1-\sigma,1]\times[1-\sigma,1]$ region in the corresponding subfigures of (c).

heightened demand for locations. To facilitate better learning of the equation information by the operator network, penalization of certain locations may be necessary, especially when the network cannot be trained on enough data. In our study, we chose to penalize the boundary points in the loss function, giving rise to a loss function of the following form:

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{NJ_r}\sum_{n=1}^{N}\sum_{j=1}^{J_r}|u_n(\boldsymbol{y}_j) - \mathcal{N}_{\boldsymbol{\theta}}(F_n)(\boldsymbol{y}_j)|^2 + \frac{\lambda}{NJ_b}\sum_{n=1}^{N}\sum_{j=1}^{J_b}|u_n(\boldsymbol{y}_j) - \mathcal{N}_{\boldsymbol{\theta}}(F_n)(\boldsymbol{y}_j)|^2,$$

where λ is the positive penalizing parameter.

Subsequently, we proceed to train DeepONets utilizing a dataset of $1000 \times 64 \times 64$ triples $(f, \boldsymbol{y}, u)$, consisting of 1000 Gaussian random fields $f \sim \mathcal{G}(0, k_{RBF})$ and 64×64 locations $\boldsymbol{y}$, with k_{RBF} denoting the radial basis function kernel. The training process is conducted over 2000 epochs, during which the penalizing parameter is set to $\lambda = 0.1$. We conduct training on both equidistant and Shishkin locations, with a comparison of their performance shown in Fig. 8. Our findings indicate that the network trained on equidistant locations effectively captures the behavior of the solution outside the boundary layer, but falls short in characterizing the behavior inside the boundary layer. In contrast, the network trained on Shishkin locations proves adept at capturing the behavior of the solution within the boundary layer.

6. Conclusion and Future Directions

In this paper, we present a novel application of DeepONets for solving singular perturbation problems, specifically the one-dimensional convection-diffusion equation. We conduct

a thorough analysis of the approximation error generated by the operator network when approximating the target solution operator and provide proof of its convergence rate with respect to the input and output dimensions of the branch net. Importantly, we demonstrate that this error is independent of ε, which is a notable finding. Since this error cannot be calculated directly, the empirical risk serves as a surrogate. Notably, the Shishkin mesh points are employed as samples on the interval $[0,1]$. We analyze the empirical risk and the corresponding generalization gap, both of which exhibit ε-uniform convergence with respect to the number of samples. Additionally, the Shishkin mesh points are utilized as locations for the loss function, and numerical experiments showcase the effectiveness of DeepONets in capturing the boundary layer behavior of solutions to singular perturbation problems. Our findings provide a novel approach to addressing this type of problem.

We examine one-dimensional steady-state singular perturbation problems, anticipating that our methodology could extend to higher-dimensional problems or those with time dependence. Additionally, employing ε as an input and utilizing DeepONets to obtain a more general operator, which requires an input (f,ε) and produces the corresponding solution u, presents a valuable avenue for further exploration. Despite their effectiveness as purely data-driven models, DeepONets fail to fully exploit the information encapsulated within the governing equation, leading to inefficiencies. A promising avenue for further development is to incorporate singular perturbation theory, as demonstrated in previous works such as [1]. Such improvements have the potential to enhance the performance of traditional models significantly.

Appendix A

A.1 Proof of Lemma 4.2

Utilizing the condition

$$\mathbb{E}_{f\sim\mu}\left|\frac{d}{dx}F(f)(x)\right| \le C\left(1+\frac{1}{\varepsilon}e^{-\alpha(1-x)/\varepsilon}\right),$$

we derive

$$\begin{aligned}
&\mathbb{E}_{f\sim\mu}\left|\int_0^1 F(f)(x)dx-\sum_{j=1}^{J}h_{j-1}F(f)(x_j)\right|\\
&=\mathbb{E}_{f\sim\mu}\left|\sum_{j=1}^{J}\int_{x_{j-1}}^{x_j}F(f)(x)-F(f)(x_j)dx\right|\\
&=\mathbb{E}_{f\sim\mu}\left|\sum_{j=1}^{J}\int_{x_{j-1}}^{x_j}\int_{x}^{x_j}\frac{d}{ds}F(f)(s)dsdx\right|\\
&\le\sum_{j=1}^{J}\int_{x_{j-1}}^{x_j}\int_{x}^{x_j}\mathbb{E}_{f\sim\mu}\left|\frac{d}{ds}F(f)(s)\right|dsdx
\end{aligned}$$

$$\leq C\sum_{j=1}^{J}\int_{x_{j-1}}^{x_j}\int_{x}^{x_j}\left(1+\frac{1}{\varepsilon}e^{-\alpha(1-s)/\varepsilon}\right)dsdx.$$

We consider two cases for specifying σ, based on its definition. If $\sigma = 2\varepsilon\ln J/\alpha$, i.e. $2\varepsilon\ln J/\alpha \leq 1/2$, we obtain the following result for $x_j \leq 1-\sigma$:

$$\begin{aligned}
&\int_{x_{j-1}}^{x_j}\int_{x}^{x_j}\left(1+\frac{1}{\varepsilon}e^{-\alpha(1-s)/\varepsilon}\right)dsdx\\
&=\frac{h_{j-1}^2}{2}-\frac{\varepsilon}{\alpha^2}\left(e^{-\alpha(1-x_j)/\varepsilon}-e^{-\alpha(1-x_{j-1})/\varepsilon}\right)+\frac{h_{j-1}}{\alpha}e^{-\alpha(1-x_j)/\varepsilon}\\
&\leq\frac{h_{j-1}^2}{2}+\frac{h_{j-1}}{\alpha}e^{-\alpha(1-x_j)/\varepsilon}\\
&\leq\frac{h_{j-1}^2}{2}+\frac{h_{j-1}}{\alpha J^2}\leq 2\left(1+\frac{1}{\alpha}\right)\frac{1}{J^2}.
\end{aligned}$$

For $x_{j-1} > 1-\sigma$, we have

$$\int_{x_{j-1}}^{x_j}\int_{x}^{x_j}\left(1+\frac{1}{\varepsilon}e^{-\alpha(1-s)/\varepsilon}\right)dsdx\leq\frac{2}{\varepsilon}\int_{x_{j-1}}^{x_j}\int_{x}^{x_j}dsdx=\frac{h_{j-1}^2}{\varepsilon}\leq\frac{4\ln J}{\alpha J^2},$$

which implies that

$$C\sum_{j=1}^{J}\int_{x_{j-1}}^{x_j}\int_{x}^{x_j}\left(1+\frac{1}{\varepsilon}e^{-\alpha(1-s)/\varepsilon}\right)dsdx\leq 2C\left(1+\frac{2}{\alpha}\right)\frac{\ln J}{J}.$$

The case $\sigma = 1/2$ is straightforward, and we directly obtain that

$$\begin{aligned}
&C\sum_{j=1}^{J}\int_{x_{j-1}}^{x_j}\int_{x}^{x_j}\left(1+\frac{1}{\varepsilon}e^{-\alpha(1-s)/\varepsilon}\right)dsdx\\
&\leq\frac{2C}{\varepsilon}\sum_{j=1}^{J}\int_{x_{j-1}}^{x_j}\int_{x}^{x_j}dsdx=\frac{C}{\varepsilon J}<2C\left(1+\frac{2}{\alpha}\right)\frac{\ln J}{J}.
\end{aligned}$$

The lemma is proven. □

A.2 Proof of Lemma 4.3

To facilitate our proof procedure, we first establish a lemma that is essential to our subsequent analysis.

Lemma A.1 (cf. Stuart [30]). *If $\mathcal{N}(0,\mathscr{C})$ is a Gaussian measure on a Hilbert space $\mathscr{H}$, then for any integer q there is a constant $C = C_q \geq 0$ such that for $x \sim \mathcal{N}(0,\mathscr{C})$, we have*

$$\mathbb{E}\|x\|^{2q}\leq C_q\left(\mathbb{E}\|x\|^2\right)^q.$$

Let $X_n = \|\mathscr{G}(F_n) - \overline{\mathscr{N}}(F_n)\|_{L^2}^2$, and define $S^N = (\sum_{n=1}^N X_n)/N$. Using basic inequalities such as the Hölder inequality, we obtain

$$\mathrm{I} = \mathbb{E}\left|\sqrt{S^N} - \sqrt{\mathbb{E}S^N}\right| \le \left(\mathbb{E}|S^N - \mathbb{E}(S^N)|^2\right)^{1/4} = \frac{1}{N^{1/4}}\left(\mathbb{E}(X_1 - \mathbb{E}X_1)^2\right)^{1/4}. \tag{A.1}$$

The term $\mathbb{E}X_1$ is closely related to the approximation error for $\overline{\mathscr{N}}$, viz. $\mathbb{E}X_1 = \widehat{\mathscr{E}}(\overline{\mathscr{N}})^2$. Besides,

$$\mathbb{E}X_1^2 = \int_{L^2} \|\mathscr{G}(f) - \overline{\mathscr{N}}(f)\|_{L^2}^4 d\mu(f) = \int_{L^2} \|f\|_{L^2}^4 d(\mathscr{G} - \overline{\mathscr{N}})_{\#}\mu(f),$$

where $(\mathscr{G} - \overline{\mathscr{N}})_{\#}\mu$ is the push-forward measure of μ under $\mathscr{G} - \overline{\mathscr{N}}$. As $\mathscr{G}$ and $\overline{\mathscr{N}}$ are bounded linear operators, $(\mathscr{G} - \overline{\mathscr{N}})_{\#}\mu$ is a Gaussian measure. Applying Lemma A.1 to the above equation and combining it with (A.1) gives

$$\mathrm{I} \lesssim \frac{\widehat{\mathscr{E}}(\overline{\mathscr{N}})}{N^{1/4}}.$$

The term II can be estimates as follows:

$$\begin{aligned}
\mathrm{II} &\le \mathbb{E}\left(\frac{1}{N}\sum_{n=1}^N \|\overline{\mathscr{N}}(F_n) - \widetilde{\mathscr{N}}(F_n)\|_{L^2}^2\right)^{1/2} \\
&= \mathbb{E}\left(\frac{1}{N}\sum_{n=1}^N \left\|\sum_{q=-K}^K \left(\mathscr{G}\circ\mathscr{D}\circ\mathscr{E}(F_n), \widetilde{\tau}_q\right)\left(\tau_q - \widetilde{\tau}_q\right)\right\|_{L^2}^2\right)^{1/2} \\
&\le \eta\mathbb{E}\left(\frac{1}{N}\sum_{n=1}^N \|\mathscr{G}\circ\mathscr{D}\circ\mathscr{E}(F_n)\|_{L^2}^2\right)^{1/2} \\
&\le \eta Lip(\mathscr{G})\left(\mathbb{E}\|\mathscr{D}\circ\mathscr{E}(F_1)\|_{L^2}^2\right)^{1/2}.
\end{aligned}$$

Note that

$$\mathscr{D}\circ\mathscr{E}(e^{2\pi ikx}) = e^{2\pi ikx}\big(|k| \le M\big),$$

where P_M is the projection operator onto $\mathrm{span}\{e^{2\pi ikx} : |k| \le M\}$. Thus, we have $\mathscr{D}\circ\mathscr{E}\circ P_M = P_M$, and since $\mathscr{D}\circ\mathscr{E}$ is linear, we get

$$\mathscr{D}\circ\mathscr{E} = \mathscr{D}\circ\mathscr{E}\circ P_M + \mathscr{D}\circ\mathscr{E}\circ P_M^{\perp} = P_M + \mathscr{D}\circ\mathscr{E}\circ P_M^{\perp}.$$

Hence,

$$\begin{aligned}
\left(\mathbb{E}\|\mathscr{D}\circ\mathscr{E}(F_1)\|_{L^2}^2\right)^{1/2} &\le \left(\mathbb{E}\|P_M F_1\|_{L^2}^2\right)^{1/2} + \left(\mathbb{E}\|\mathscr{D}\circ\mathscr{E}\circ(P_M^{\perp}F_1)\|_{L^2}^2\right)^{1/2} \\
&\le \left(\mathbb{E}\|F_1\|_{L^2}^2\right)^{1/2} + \sqrt{\sum_{|k|>M}\lambda_k}
\end{aligned}$$

$$= \sqrt{\sum_{k\in\mathbb{Z}} \lambda_k} + \sqrt{\sum_{|k|>M} \lambda_k}$$
$$\leq 2\sqrt{\sum_{k\in\mathbb{Z}} \lambda_k} \leq 2.$$

Combining the above results gives $\mathrm{II} \lesssim \eta = p^{-n}$.

Similarly, the estimate of IV follows the same procedure as for II, yielding

$$\mathrm{IV} \lesssim \eta = p^{-n}.$$

The final result follows from the above estimates for I, II and IV. □

A.3 Proof of Lemma 4.4

The proof of this theorem relies on a series of lemmas, which we will now present along with their corresponding proofs.

Lemma A.2. *Let f drawn from the measure μ, and let $\widehat{f_k}$ and c_k denote the Fourier coefficients and discrete Fourier coefficients of f, respectively. Specifically, we have,*

$$\widehat{f_k} = \int_0^1 f(x)e^{-i2\pi kx}dx, \quad c_k = \frac{1}{m}\sum_{j=1}^{m} f(x_j)e^{-\frac{i2\pi jk}{m}}.$$

For $0 \leq k \leq M$, it can be shown that

$$\left(\mathbb{E}_{f\sim\mu}|\widehat{f_k}|^2\right)^{1/2} \leq Ce^{-\pi^2 l^2 k^2},$$
$$\left(\mathbb{E}_{f\sim\mu}|c_k|^2\right)^{1/2} \leq Ce^{-\pi^2 l^2 k^2},$$
$$\left(\mathbb{E}_{f\sim\mu}|\widehat{f_k} - c_k|^2\right)^{1/2} \leq \frac{C}{m}e^{-\pi^2 l^2 k^2},$$

where C is a constant that solely depends on l.

Proof. Using the K-L expansion, we represent $f \sim \mu$ as

$$f(x) = \sum_{n=-\infty}^{\infty} \sqrt{\lambda_n}\xi_n\phi_n(x),$$

where $\lambda_n = \sqrt{2\pi}le^{-2\pi^2 n^2 l^2}$, $\phi_n(x) = e^{i2\pi nx}$ and $\xi_n \sim \mathcal{N}(0,1)$ are i.i.d. Gaussian random variables. Then

$$\left(\mathbb{E}_{f\sim\mu}|\widehat{f_k}|^2\right)^{1/2} = \left(\mathbb{E}|\sqrt{\lambda_k}\xi_k|^2\right)^{1/2} = \sqrt{\lambda_k} = \left(\sqrt{2\pi}l\right)^{1/2} e^{-\pi^2 k^2 l^2}.$$

On the other hand, we can obtain

$$\begin{aligned}\left(\mathbb{E}_{f\sim\mu}|\widehat{f}_k - c_k|^2\right)^{1/2} &= \left(\mathbb{E}_{f\sim\mu}\left|\widehat{f}_k - \frac{1}{m}\sum_{j=1}^{m} f(x_j)e^{-i2\pi jk/m}\right|^2\right)^{1/2}\\ &= \left(\mathbb{E}_{f\sim\mu}\left|\widehat{f}_k - \frac{1}{m}\sum_{j=1}^{m}\sum_{n=-\infty}^{\infty}\widehat{f}_n e^{i2\pi jn/m}e^{-i2\pi jk/m}\right|^2\right)^{1/2}\\ &= \left(\mathbb{E}_{f\sim\mu}\left|\widehat{f}_k - \sum_{n=-\infty}^{\infty}\widehat{f}_{k+nm}\right|^2\right)^{1/2}\\ &\lesssim l^{1/2}\sum_{n=1}^{\infty} e^{-\pi^2 l^2(k+nm)^2} + e^{-\pi^2 l^2(k-nm)^2},\end{aligned}$$

where

$$\sum_{n=1}^{\infty} e^{-\pi^2 l^2(k+nm)^2} \le \int_0^{+\infty} e^{-\pi^2 l^2(k+xm)^2}dx = \frac{1}{\pi lm}\int_{k\pi l}^{+\infty} e^{-y^2}dy \lesssim \frac{1}{ml}e^{-\pi^2 l^2 k^2},$$

and considering that $0 \le k \le M$, we have

$$\sum_{n=1}^{\infty} e^{-\pi^2 l^2(k-nm)^2} \le e^{-\pi^2 l^2 k^2}\sum_{n=1}^{\infty} e^{-\pi^2 l^2 nm} \le e^{-\pi^2 l^2 k^2}\int_0^{+\infty} e^{-\pi^2 l^2 xm}dx = \frac{1}{\pi^2 l^2 m}e^{-\pi^2 l^2 k^2},$$

thus

$$\left(\mathbb{E}_{f\sim\mu}|\widehat{f}_k - c_k|^2\right)^{1/2} \lesssim \frac{1}{m}\left(l^{-1/2} + l^{-3/2}\right)e^{-\pi^2 l^2 k^2}.$$

Furthermore, c_k can be bounded by

$$(\mathbb{E}_{f\sim\mu}|c_k|^2)^{1/2} \le \left(\mathbb{E}_{f\sim\mu}|\widehat{f}_k - c_k|^2\right)^{1/2} + (\mathbb{E}_{f\sim\mu}|\widehat{f}_k|^2)^{1/2} \le Ce^{-\pi^2 l^2 k^2},$$

where C is a constant that depends only on l. □

The next result follows directly from Lemma A.2.

Corollary A.1. *Under the conditions of Lemma* A.2*, we have*

$$\begin{aligned}\left(\mathbb{E}_{f\sim\mu}\left|f - \sum_{q=-k}^{k} c_q\phi_q(x)\right|^2\right)^{1/2} &\le \frac{C}{k},\\ \left(\mathbb{E}_{f\sim\mu}\left|\mathscr{G}\left(f - \sum_{q=-k}^{k} c_q\phi_q(x)\right)\right|^2\right)^{1/2} &\le \frac{C}{k},\end{aligned}$$

where C depends solely on l.

Proof. To establish the first inequality, we proceed as follows:

$$\begin{aligned}
&\left(\mathbb{E}_{f\sim\mu}\left|f-\sum_{q=-k}^{k}c_q\phi_q(x)\right|^2\right)^{1/2}\\
&=\left(\mathbb{E}_{f\sim\mu}\left|\sum_{q=-\infty}^{\infty}\widehat{f}_q\phi_q-\sum_{|q|\le k}c_q\phi_q\right|^2\right)^{1/2}\\
&=\left(\mathbb{E}_{f\sim\mu}\left|\sum_{|q|\le k}(\widehat{f}_q-c_q)\phi_q+\sum_{|q|>k}\widehat{f}_q\phi_q\right|^2\right)^{1/2}\\
&\le\sum_{|q|\le k}\left(\mathbb{E}_{f\sim\mu}|\widehat{f}_q-c_q|^2\right)^{1/2}+\sum_{|q|>k}\left(\mathbb{E}_{f\sim\mu}|\widehat{f}_q|^2\right)^{1/2}\\
&\le\frac{C}{m}\sum_{|q|\le k}e^{-\pi^2l^2q^2}+C\sum_{|q|>k}e^{-\pi^2l^2q^2}\le\frac{C}{k}.
\end{aligned}$$

Based on Lemma 2.1, it follows that $|\mathscr{G}(\phi_k)|\lesssim 1$. Moreover, by repeating the aforementioned proof, we can derive the second inequality stated in this corollary. □

We present our final lemma, the proof of which closely follows that of [12].

Lemma A.3. *Let $u=\mathscr{G}(f)$ be the solution to the equation*

$$\begin{aligned}
&-\varepsilon u''+p(x)u'+q(x)u=f,\quad x\in(0,1),\\
&u(0)=u(1)=0,
\end{aligned}$$

and $h=f-qu$. Then

$$|u'(1)|\lesssim\frac{1}{\varepsilon^2}\int_0^1\int_t^1 e^{-\alpha(s-t)/\varepsilon}|h(s)|dsdt.$$

Proof. The function u satisfies

$$u(x)=u_p(x)+K_1+K_2\int_x^1 e^{-(\int_t^1 p(s)ds)/\varepsilon}dt, \tag{A.2}$$

where $K_1=u(1)$, $K_2=u'(1)$, and

$$u_p(x)=-\int_x^1\int_t^1\frac{1}{\varepsilon}e^{-(\int_t^s p(w)dw)/\varepsilon}h(s)dsdt.$$

Choosing $x=0$ in (A.2) yields

$$u(0)=u_p(0)+K_1+K_2\int_0^1 e^{-(\int_t^1 p(s)ds)/\varepsilon}dt.$$

Since $u(0)=u(1)=0$ and $p(x)\geq\alpha>0$ is bounded on $[0,1]$, we have

$$\varepsilon|K_2|\lesssim|K_2|\int_0^1 e^{-c(1-t)/\varepsilon}dt\leq|u_p(0)|\leq\int_0^1\int_t^1\frac{1}{\varepsilon}e^{-\alpha(s-t)/\varepsilon}|h(s)|dsdt,$$

which implies that

$$|u'(1)|=|K_2|\lesssim\frac{1}{\varepsilon^2}\int_0^1\int_t^1 e^{-\alpha(s-t)/\varepsilon}|h(s)|dsdt. \qquad \square$$

Using the above lemmas, we can now provide the proof of Lemma 4.4.

Proof. Let

$$u(x)=\mathscr{G}\big(f-\mathscr{D}\circ\mathscr{E}(f)\big)(x),\quad h(x)=f-\mathscr{D}\circ\mathscr{E}(f)-q(x)u.$$

Then $z(x)=du/dx$ satisfies

$$z(x)=z(1)e^{-(\int_x^1 p(s)ds)/\varepsilon}+\frac{1}{\varepsilon}\int_x^1 h(t)e^{-(\int_x^t p(s)ds)/\varepsilon}dt.$$

Employing the Eq. (4.13), Lemma A.3, and the estimate $p(x)\geq\alpha>0$ gives

$$\begin{aligned}
&\mathbb{E}_{f\sim\mu}\left|\big(\mathscr{G}(f)(x)-\widetilde{\mathscr{N}}(f)(x)\big)\frac{d}{dx}\big(\mathscr{G}(f)(x)-\widetilde{\mathscr{N}}(f)(x)\big)\right|\\
&=\mathbb{E}_{f\sim\mu}\left|\mathscr{G}\big(f-\mathscr{D}\circ\mathscr{E}(f)\big)(x)\left[z(1)e^{-(\int_x^1 p(s)ds)/\varepsilon}+\frac{1}{\varepsilon}\int_x^1 h(t)e^{-(\int_x^t p(s)ds)/\varepsilon}dt\right]\right|\\
&\leq e^{-(\int_x^1 p(s)ds)/\varepsilon}\mathbb{E}_{f\sim\mu}|\mathscr{G}(f-\mathscr{D}\circ\mathscr{E}(f))(x)z(1)|\\
&\quad+\frac{1}{\varepsilon}\int_x^1 e^{-(\int_x^t p(s)ds)/\varepsilon}\mathbb{E}_{f\sim\mu}|\mathscr{G}(f-\mathscr{D}\circ\mathscr{E}(f))(x)h(t)|\,dt\\
&\lesssim\frac{1}{\varepsilon^2}e^{-\alpha(1-x)/\varepsilon}\int_0^1\int_t^1 e^{-\alpha(s-t)/\varepsilon}\mathbb{E}_{f\sim\mu}|\mathscr{G}(f-\mathscr{D}\circ\mathscr{E}(f))(x)h(s)|\,dsdt\\
&\quad+\frac{1}{\varepsilon}\int_x^1 e^{-\alpha(t-x)/\varepsilon}\mathbb{E}_{f\sim\mu}|\mathscr{G}(f-\mathscr{D}\circ\mathscr{E}(f))(x)h(t)|\,dt. \qquad (A.3)
\end{aligned}$$

Substituting the expression for $h(x)$ into $\mathbb{E}_{f\sim\mu}|\mathscr{G}(f-\mathscr{D}\circ\mathscr{E}(f))(x)h(s)|$ and using the boundedness of $q(x)$ on $[0,1]$ and Corollary A.1 leads to the estimate

$$\mathbb{E}_{f\sim\mu}|\mathscr{G}(f-\mathscr{D}\circ\mathscr{E}(f))(x)h(s)|\leq\frac{C}{m^2}.$$

After substituting this into (A.3), the required inequality can be obtained via a straightforward integration calculation. $\square$

A.4 Lipschitz continuity of the operator in Example 5.1

Lemma A.4. *Let $\mathscr{G} : f \mapsto u$ denote the solution operator for the following boundary value problem:*

$$-\varepsilon u''(x) + u'(x) = f(x), \quad x \in (0,1),$$
$$u(0) = u(1) = 0,$$

if $0 < \varepsilon \leq 1/2$, then the operator $\mathscr{G}$ is Lipschitz continuous.

Proof. As $\mathscr{G}$ is linear, it suffices to show that $\|u\|_{L^2} \lesssim \|f\|_{L^2}$. To achieve this, we substitute $v = e^{-x}u$ into (1.2). Utilizing $uu' = (u^2)'/2$ and $u(0) = u(1) = 0$, we obtain

$$\int_0^1 \varepsilon (u')^2 e^{-x} dx + \frac{1-\varepsilon}{2} \int_0^1 e^{-x} u^2 dx = \int_0^1 e^{-x} u f \, dx.$$

Next we obtain the inequality

$$\frac{1-\varepsilon}{2e} \int_0^1 u^2 dx \leq \|u\|_{L^2} \|f\|_{L^2}.$$

Notably, since $0 < \varepsilon \ll 1$, we can reasonably posit that $\varepsilon \leq 1/2$ and thereby complete the proof of the lemma. □

Acknowledgments

Z.Y. Huang was partially supported by the NSFC Projects No. 12025104 and 81930119. Y. Li was partially supported by the NSFC Project No. 62106103 and by the Fundamental Research Funds for the Central Universities No. ILA22023.

References

[1] A. Arzani, K.W. Cassel and R.M. D'Souza, *Theory-guided physics-informed neural networks for boundary layer problems with singular perturbation*, J. Comput. Phys. **473**, Paper No. 111768 (2023).

[2] C. Beck, S. Becker, P. Grohs, N. Jaafari and A. Jentzen, *Solving stochastic differential equations and Kolmogorov equations by means of deep learning*, arXiv:1806.00421 (2018).

[3] S. Cai, Z. Wang, L. Lu, T.A. Zaki and G.E. Karniadakis, *DeepM&Mnet: Inferring the electroconvection multiphysics fields based on operator approximation by neural networks*, J. Comput. Phys. **436**, Paper No. 110296 (2021).

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

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

[6] W. E, J. Han and A. Jentzen, *Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations*, Commun. Math. Stat. **5**, 349–380 (2017).

[7] 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).
[8] J. He, L. Li, J. Xu and C. Zheng, *ReLU deep neural networks and linear finite elements*, J. Comput. Math. **38**, 502-–527 (2020).
[9] A.D. Jagtap, K. Kawaguchi and G.E. Karniadakis, *Adaptive activation functions accelerate convergence in deep and physics-informed neural networks*, J. Comput. Phys. **404**, Paper No. 109136 (2020).
[10] P. Jin, S. Meng and L. Lu, *MIONet: Learning multiple-input operators via tensor product*, SIAM J. Sci. Comput. **44**, A3490–A3514 (2022).
[11] K. Kawaguchi, L.P. Kaelbling and Y. Bengio, *Generalization in deep learning*, arXiv:1710.05468 (2017).
[12] R.B. Kellogg and A. Tsan, *Analysis of some difference approximations for a singular perturbation problem without turning points*, Math. Comp. **32**, 1025–1039 (1978).
[13] N. Kopteva and E. O'Riordan, *Shishkin meshes in the numerical solution of singularly perturbed differential equations*, Int. J. Numer. Anal. Model. **7**, 393–415 (2010).
[14] N. Kovachki, Z. Li, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. Stuart and A. Anandkumar, *Neural operator: Learning maps between function spaces*, arXiv:2108.08481 (2021).
[15] S. Lanthaler, S. Mishra and G.E. Karniadakis, *Error estimates for DeepONets: A deep learning framework in infinite dimensions*, Trans. Math. Appl. **6**, tnac001 (2022)
[16] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart and A. Anandkumar, *Fourier neural operator for parametric partial differential equations*, arXiv:2010.08895 (2020).
[17] G. Lin, C. Moya and Z. Zhang, *Accelerated replica exchange stochastic gradient Langevin diffusion enhanced Bayesian DeepONet for solving noisy parametric PDEs*, arXiv:2111.02484 (2021).
[18] Y. Liu, J. Li, S. Sun and B. Yu, *Advances in Gaussian random field generation: A review*, Comput. Geosci. **23**, 1011–1047 (2019).
[19] Z. Liu, W. Cai and Z.Q.J. Xu, *Multi-scale deep neural network (MscaleDNN) for solving Poisson-Boltzmann equation in complex domains*, Commun. Comput. Phys. **28**, 1970–2001 (2020).
[20] M. Loève, *Probability Theory II*, Springer-Verlag (1978).
[21] L. Lu, P. Jin, G. Pang, Z. Zhang and G.E. Karniadakis, *Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators*, Nat. Mach. Intell. **3**, 218–229 (2021).
[22] L. Lu, X. Meng, S. Cai, Z. Mao, S. Goswami, Z. Zhang and G.E. Karniadakis, *A comprehensive and fair comparison of two neural operators (with practical extensions) based on fair data*, Comput. Methods Appl. Mech. Engrg. **393**, Paper No. 114778 (2022).
[23] Z. Mao, L. Lu, O. Marxen, T.A. Zaki and G.E. Karniadakis, *DeepM&Mnet for hypersonics: Predicting the coupled flow and finite-rate chemistry behind a normal shock using neural-network approximation of operators*, J. Comput. Phys. **447**, Paper No. 110698 (2021).
[24] J.J.H. Miller, E. O'Riordan and G.I. Shishkin, *Fitted Numerical Methods for Singular Perturbation Problems: Error Estimates in the Maximum Norm for Linear Problems in One and Two Dimensions*, World Scientific (1996).
[25] S. Qian, H. Liu, C. Liu, S. Wu and H.S. Wong, *Adaptive activation functions in convolutional neural networks*, Neurocomputing **272**, 204–212 (2018).
[26] N. Rahaman, A. Baratin, D. Arpit, F. Draxler, M. Lin, F. Hamprecht, Y. Bengio and A. Courville, *On the spectral bias of neural networks*, in: *International Conference on Machine Learning*, pp. 5301–5310, PMLR (2019).
[27] 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).

[28] H.G. Roos, M. Stynes and L. Tobiska, *Robust Numerical Methods for Singularly Perturbed Differential Equations: Convection-Diffusion-Reaction and Flow Problems*, Springer-Verlag (2008).
[29] V. Sitzmann, J. Martel, A. Bergman, D. Lindell and G. Wetzstein, *Implicit neural representations with periodic activation functions*, in: *Advances in Neural Information Processing Systems*, pp. 7462–7473, Morgan Kaufmann (2020).
[30] A.M. Stuart, *Inverse problems: A Bayesian perspective*, Acta Numer. **19**, 451–559 (2010).
[31] M. Stynes, *Steady-state convection-diffusion problems*, Acta Numer. **14**, 445–508 (2005).
[32] S. Wang, H. Wang and P. Perdikaris, *Learning the solution operator of parametric partial differential equations with physics-informed DeepONets*, Sci. Adv. **7**, eabi8605 (2021).
[33] A.I. Zadorin, *Refined-mesh interpolation method for functions with a boundary-layer component*, Comput. Math. Math. Phys. **48**, 1634–1645 (2008).