Computers and Mathematics with Applications 113 (2022) 103–116

Contents lists available at ScienceDirect

Computers and Mathematics with Applications

www.elsevier.com/locate/camwa

Adaptive two-layer ReLU neural network: II. Ritz approximation to elliptic PDEs ☆

Min Liu [a], Zhiqiang Cai [b,*]

[a] School of Mechanical Engineering, Purdue University, 585 Purdue Mall, West Lafayette, IN 47907-2088, United States of America
[b] Department of Mathematics, Purdue University, 150 N. University Street, West Lafayette, IN 47907-2067, United States of America

ARTICLE INFO

Keywords:
Adaptivity
A posteriori estimator
Diffusion-reaction problem
Neural network
Ritz method

ABSTRACT

In the companion paper [1], we introduce adaptive network enhancement (ANE) method for the best least-squares approximation to a target function by using two-layer ReLU neural networks (NNs). In this paper, we apply the ANE method for solving self-adjoint second-order elliptic partial differential equations (PDEs). The underlying PDE is discretized by the Ritz method using a two-layer spline neural network based on either the primal or dual formulations that minimize the respective energy or complimentary functionals. Essential boundary conditions are imposed weakly through the functionals with proper norms. It is proved that the Ritz approximation is the best approximation in the energy norm; moreover, effect of numerical integration for the Ritz approximation is analyzed as well. Two estimators for adaptive neuron enhancement method are introduced, one is the so-called recovery estimator and the other is the least-squares estimator. Finally, numerical results for diffusion problems with either corner or intersecting interface singularities are presented.

1. Introduction

Recent success of neural networks (NNs) for many artificial intelligence tasks has led wide applications to other fields, including recent studies of using neural network models to numerically solve partial differential equations (PDEs) (see, e.g., [2–6]). Because neural network functions are nonlinear functions of the parameters, discretization of a PDE is set up as an optimization problem through either the natural minimization or manufactured least-squares (LS) principles. Hence, existing methods consist of (1) the deep Ritz method [4] and (2) the deep LS method such as the deep Galerkin method (DGM) [6], the physics-informed neural networks (PINN) [5], the deep LS and FOSLS methods [3], etc. The former has the least variables, requires the least smoothness, but is applicable to a single class of problems. The latter is applicable to a large class of PDEs, but either has additional variables such as the FOSLS or requires additional smoothness like the LS.

Neural networks produce a new class of functions through compositions of linear transformations and activation functions. This class of functions is extremely rich. For example, it contains piecewise polynomials, which are the footing of spectral elements, and continuous and discontinuous finite element methods. It approximates polynomials of any degree with exponential efficiency, even using simple activation functions like ReLU. Despite many efforts, it is widely accepted that approximation properties of NNs are not yet well-understood. As a consequence, design of network structures for approximating the solution of a PDE within the prescribed accuracy remains open and is mainly done by time consuming trial-and-error.

To address the issue on how to design a minimal network model required to approximate a function/PDE within the prescribed accuracy, in terms of width, depth, and the number of parameters, we propose and study the adaptive neuron enhancement (ANE) method for approximating a function in the companion paper [1] and for solving a self-adjoint second-order elliptic PDE in this paper. Specifically, for a given tolerance $\epsilon > 0$, the ANE method generates a two-layer spline neural network such that the approximation accuracy is within the prescribed tolerance. The key ingredient of the method is the neuron enhancement strategy which determines how many new neurons to be added, when the current approximation is not within the given accuracy. This is done through local error indicators collected on the physical subdomains, and a proper neuron initialization.

The underlying PDE is discretized by the Ritz method using a two-layer spline neural network based on either the primal or dual formulations. The primal problem minimizes the energy functional with the (essential) Dirichlet boundary condition imposed weakly [3]. Another way to impose

☆ This work was supported in part by the National Science Foundation under grant DMS-2110571.
* Corresponding author.
E-mail addresses: liu66@purdue.edu (M. Liu), caiz@purdue.edu (Z. Cai).

https://doi.org/10.1016/j.camwa.2022.03.010
Received 8 March 2021; Received in revised form 18 January 2022; Accepted 5 March 2022

the Dirichlet boundary condition is the well-known Nitsche's method that requires the stabilization constant is sufficient large (see [7] in the context of neural networks). When the dual (flux) variable is important for the underlying application, we may maximize the complementary functional directly; in this case, the Neumann boundary condition becomes essential and is again enforced weakly through the functional with a proper norm.

Even though the set of neural network functions does not form a space, we show that the Ritz approximation based on either the primal or the dual formulation is the best approximation in the energy norm. Moreover, we are able to analyze the effect of numerical integration as well. As expected, the error in the energy norm by the Ritz approximation with numerical integration is bounded by the approximation error of the neural network and the error of the numerical integration. This result may be considered as the extension of the first Strang's lemma (see, e.g., [8]) from the Galerkin approximation over a subspace to the Ritz approximation over a set.

A posteriori error estimation is important for the ANE method. In this paper, we consider two estimators: the recovery and the least-squares estimators. For the primal formulation, the recovery estimator is defined as a weighted L^2 norm of difference between the numerical and the recovered fluxes. When the recovered flux is more accurate than the numerical flux (i.e., the so-called saturation assumption [9]), the recovery estimator is proved to be reliable and efficient. The least-squares estimator [10] adds an additional term, the L^2 norm of the residual, to the recovery estimator.

The paper is organized as follows. The primal and dual formulations of the diffusion-reaction problem and their well-posedness are discussed in section 2. The Ritz approximation using neural networks are described in section 3. Error estimates of the Ritz approximations and effect of numerical integration are obtained in sections 3 and 4, respectively. *A posteriori* error estimators, the ANE method, and initialization at each stage of the ANE method are introduced in the respective section 5, 6, and 7. Finally, we present numerical results for problems with either corner or intersecting interface singularities in section 8.

In this paper, we will use the standard notation and definitions for the Sobolev space $H^s(\Omega)$ and $H^s(\Gamma)$ for a subset Γ in $\partial\Omega$. The standard associated inner product and norms are denoted by $(\cdot,\cdot)_{s,\Omega}$ and $(\cdot,\cdot)_{s,\Gamma}$ and by $\|\cdot\|_{s,\Omega}$ and $\|\cdot\|_{s,\Gamma}$, respectively. When $s=0$, $H^0(\Omega)$ coincides with $L^2(\Omega)$. Denote the corresponding norms on product space $H^s(\Omega)^d$ by $\|\cdot\|_{s,\Omega,d}$ and $|\cdot|_{s,\Omega,d}$. When there is no ambiguity, the subscript Ω and d in the designation of norms will be suppressed.

2. Diffusion-reaction problem

Let Ω be a bounded domain in $\mathbb{R}^d$ with Lipschitz boundary $\partial\Omega = \bar{\Gamma}_D \cup \bar{\Gamma}_N$. Consider the following self-adjoint second-order scalar elliptic partial differential equation:

$$\begin{cases} -\mathrm{div}\,(A\nabla u) + c\,u &= f, \quad \text{in } \Omega \subset \mathbb{R}^d, \\ u &= g_D, \quad \text{on } \Gamma_D, \\ -\mathbf{n}\cdot A\nabla u &= g_N, \quad \text{on } \Gamma_N, \end{cases} \tag{2.1}$$

where $f \in L^2(\Omega)$, $c \in L^2(\Omega)$, $g_D \in H^{1/2}(\Gamma_D)$, $g_N \in H^{-1/2}(\Gamma_N)$; $A(x)$ is a $d\times d$ symmetric matrix-valued function in $L^2(\Omega)^{d\times d}$; and $\mathbf{n}$ is the outward unit vector normal to the boundary. We assume that A is uniformly positive definite and that $c \geq 0$.

The natural optimization problem of (2.1) is the so-called primal problem that minimizes the energy functional over $H^1_{g,D}(\Omega) = \{v \in H^1(\Omega) : v = g_D \text{ on } D\}$. Since it is difficult for neural network functions to satisfy boundary conditions (see [4]), as in [3], we enforce the Dirichlet (essential) boundary condition weakly through the energy functional. To this end, define the energy functional by

$$\begin{aligned} J(v) &= \frac{1}{2}\left\{\|A^{1/2}\nabla v\|^2_{0,\Omega} + \|c^{1/2}v\|^2_{0,\Omega} + \gamma_D\|v - g_D\|^2_{1/2,\Gamma_D}\right\} - \Big((f,v) + (g_N,v)_{0,\Gamma_N}\Big) \\ &= \frac{1}{2}a(v,v) - f(v) + \frac{\gamma_D}{2}\|g_D\|^2_{1/2,\Gamma_D}, \end{aligned} \tag{2.2}$$

where $\gamma_D > 0$ is a constant and the quadratic form $a(\cdot,\cdot)$ and the linear form $f(\cdot)$ are given by

$$a(v,v) = \|A^{1/2}\nabla v\|^2_{0,\Omega} + \|c^{1/2}v\|^2_{0,\Omega} + \gamma_D\|v\|^2_{1/2,\Gamma_D}$$

and

$$f(v) = (f,v) + (g_N,v)_{0,\Gamma_N} + \gamma_D(g_D,v)_{1/2,\Gamma_D},$$

respectively. Then the primal problem is to find $u \in V := H^1(\Omega)$ such that

$$J(u) = \min_{v\in V} J(v). \tag{2.3}$$

Remark 2.1. Another way to enforce the Dirichlet boundary condition is the well-known Nitsche's method usually stated in the Galerkin formulation. Its equivalent form for the Ritz formulation is to transform the Dirichlet boundary condition to the Robin boundary condition by penalization *(see [7] in the context of neural networks)*, and the penalization constant is usually required to be large.

Proposition 2.2. *Problem (2.3) has a unique solution $u \in V$. Moreover, the solution u satisfies the following a priori estimate:*

$$\|u\|_{1,\Omega} \leq C\left(\|f\|_{-1,\Omega} + \|g_D\|_{1/2,\Gamma_D} + \|g_N\|_{-1/2,\Gamma_N}\right). \tag{2.4}$$

Proof. By the assumptions on A and c and the trace theorem, the bilinear form $a(\cdot,\cdot)$ is $H^1(\Omega)$ elliptic, i.e., there exists a positive constant α such that

$$\alpha\|v\|_{1,\Omega}^2 \le a(v,v), \quad \forall\, v \in V. \tag{2.5}$$

It is easy to see that the bilinear form $a(\cdot,\cdot)$ and the linear form $f(\cdot)$ are continuous in $V\times V$ and V, respectively. Then the Lax-Milgram lemma implies that problem (2.3) has one and only one solution in V.

The solution $u\in V$ of problem (2.3) may be characterized by the relation

$$a(u,v) = f(v), \quad \forall\, v \in V.$$

Now, the *a priori* estimate in (2.4) is a direct consequence of (2.5) and the fact that

$$|f(u)| \le C\left(\|f\|_{-1,\Omega} + \|g_D\|_{1/2,\Gamma_D} + \|g_N\|_{-1/2,\Gamma_N}\right)\|u\|_{1,\Omega}.$$

This completes the proof of the proposition. □

Another optimization problem of (2.1) is the so-called dual problem that maximizes the complementary functional for the dual variable $\boldsymbol{\sigma} = -A\nabla u$ over the dual space

$$\Sigma := H(\mathrm{div};\Omega) = \{\boldsymbol{\tau}\in L^2(\Omega)^d \,:\, \nabla\cdot\boldsymbol{\tau}\in L^2(\Omega)\}.$$

For simplicity of presentation, assume that the reaction coefficient $c(\mathbf{x})$ is positive. The negative of the complementary functional is given by

$$\begin{aligned} J^*(\boldsymbol{\tau}) &= \frac{1}{2}\left\{\|A^{-1/2}\boldsymbol{\tau}\|_{0,\Omega}^2 + \|\frac{1}{\sqrt{c}}(\nabla\cdot\boldsymbol{\tau} - f)\|_{0,\Omega}^2 + \gamma_N\|\boldsymbol{\tau}\cdot\mathbf{n} + g_N\|_{-1/2,\Gamma_N}^2\right\} + \int_{\Gamma_D} g_D(\boldsymbol{\tau}\cdot\mathbf{n})ds \\ &= \frac{1}{2}a^*(\boldsymbol{\tau},\boldsymbol{\tau}) - f^*(\boldsymbol{\tau}) + \frac{1}{2}\left(\|c^{-1/2}f\|_{0,\Omega}^2 + \gamma_N\|g_N\|_{-1/2,\Gamma_N}^2\right), \end{aligned} \tag{2.6}$$

where $\gamma_N > 0$ is a constant and the quadratic form $a^*(\cdot,\cdot)$ and the linear form $f^*(\cdot)$ are given by

$$a(\boldsymbol{\tau},\boldsymbol{\tau}) = \|A^{-1/2}\boldsymbol{\tau}\|_{0,\Omega}^2 + \|c^{-1/2}\nabla\cdot\boldsymbol{\tau}\|_{0,\Omega}^2 + \gamma_N\|\boldsymbol{\tau}\cdot\mathbf{n}\|_{-1/2,\Gamma_N}^2$$

and

$$f^*(\boldsymbol{\tau}) = (f, c^{-1}\nabla\cdot\boldsymbol{\tau}) - (g_D, \boldsymbol{\tau}\cdot\mathbf{n})_{0,\Gamma_D} - \gamma_N(g_N, \boldsymbol{\tau}\cdot\mathbf{n})_{-1/2,\Gamma_N},$$

respectively. The Neumann boundary condition becomes essential for the dual problem and is enforced weakly through the complementary functional defined in (2.6). The dual problem is then to seek $\boldsymbol{\sigma}\in\Sigma$ such that

$$J^*(\boldsymbol{\sigma}) = \min_{\boldsymbol{\tau}\in\Sigma} J^*(\boldsymbol{\tau}). \tag{2.7}$$

Proposition 2.3. *Problem (2.7) has a unique solution $\boldsymbol{\sigma}\in\Sigma$. Moreover, the solution $\boldsymbol{\sigma}$ satisfies the following a priori estimate:*

$$\|\boldsymbol{\sigma}\|_{0,\Omega} \le C\left(\|f\|_{-1,\Omega} + \|g_D\|_{1/2,\Gamma_D} + \|g_N\|_{-1/2,\Gamma_N}\right).$$

Proof. The proposition may be proved in a similar fashion as that of Proposition 2.2. □

3. Neural network methods

A two-layer neural network (NN) consists of an input and output layers. The output layer usually has no activation function. Let τ_k be the spline activation function of the form:

$$\tau_k(t) = \max\{0, t^k\} = \begin{cases} 0, & t<0, \\ t^k, & t\ge 0, \end{cases}$$

for a fixed integer $k>0$. It is clear that $\tau_k(t)$ is a piece-wise polynomial of degree k in $C^{k-1}(\mathbb{R})$ with one breaking point $t=0$. When $k=1$, the activation function $\tau_1(t)$ is the popular rectified linear unit (ReLU). A two-layer spline NN with n neurons generates the following functional class from $\mathbb{R}^d\to\mathbb{R}^o$:

$$\mathcal{M}_n^o(\tau_k) = \left\{\mathbf{c}_0 + \sum_{i=1}^n \mathbf{c}_i\tau_k(\boldsymbol{\omega}_i\cdot\mathbf{x} - b_i) \,:\, b_i\in\mathbb{R},\ \boldsymbol{\omega}_i\in S^{d-1},\ \mathbf{c}_i\in\mathbb{R}^o\right\}, \tag{3.1}$$

where d and o are the respective dimension of the input and output; $\boldsymbol{\omega}_i\in S^{d-1}$ and $b_i\in\mathbb{R}$ are the respective weights and bias of the input layer; $\mathbf{c}_i\in\mathbb{R}^o$ and $\mathbf{c}_0\in\mathbb{R}^o$ are the respective weights and bias of the output layer; and S^{d-1} is the unit sphere in $\mathbb{R}^d$. The total number of parameters of $\mathcal{M}_n^o(\tau_k)$ is

$$M_d(n) = (d+o)n + o,$$

where $\{\mathbf{c}_i\}_{i=0}^n$ are linear parameters and $\{\boldsymbol{\omega}_i, b_i\}_{i=1}^n$ are nonlinear parameters.

Remark 3.1. Since τ_k is not a polynomial, it has been proven (see, e.g., [11,12]) that the linear space

$$\mathcal{M}(\tau_k) = \left\{ v(\mathbf{x}) \in \mathcal{M}_n^1(\tau_k) : n \in \mathbb{Z}_+ \right\}$$

is dense in $C(K)$, the space of all continuous functions defined on a compact set $K \in \mathbb{R}^d$. Note that $\mathcal{M}_n^o(\tau_k)$ is a subset, but not a subspace, of $\mathcal{M}^o(\tau_k) = \underbrace{\mathcal{M}(\tau_k) \times \cdots \times \mathcal{M}(\tau_k)}_{o}$.

Even though results on approximation order are still scarce, there are two noticeable results for target functions in Sobolev space in the $L^2(\Omega)$ norm by Petrushev *[13]* and in spectral Barron space in the Sobolev norm by Siegel and Xu *[14]*. It is not clear if the former has been extended to the $H^1(\Omega)$ norm. For the latter, solutions of very few partial differential equations have been shown in the spectral Barron space *[15]*.

To approximate the solution of (2.1) using neural network functions, the Ritz method is to minimize the energy functional over the set $\mathcal{M}_n^1(\tau_k)$, i.e., finding $u_n \in \mathcal{M}_n^1(\tau_k) \subset H^1(\Omega)$ such that

$$J(u_n) = \min_{v \in \mathcal{M}_n^1(\tau_k)} J(v). \tag{3.2}$$

Since $\mathcal{M}_n^1(\tau_k)$ is not convex, problem (3.2) has many solutions.

Theorem 3.2. *Let $u \in H^1(\Omega)$ be the solution of problem (2.3), and let $u_n \in \mathcal{M}_n^1(\tau_k)$ be a solution of (3.2). Then we have*

$$\|u - u_n\|_a = \inf_{v \in \mathcal{M}_n^1(\tau_k)} \|u - v\|_a, \tag{3.3}$$

where $\|v\|_a := \sqrt{a(v,v)}$ is the energy norm for the primal variable.

Proof. The proof of the theorem follows easily from the standard error estimate for the Ritz approximation: for any $v \in \mathcal{M}_n^1(\tau_k)$,

$$\|u - u_n\|_a^2 = 2\left(J(u_n) - J(u)\right) \le 2\left(J(v) - J(u)\right) = \|u - v\|_a^2,$$

which implies the validity of (3.3). □

When the flux variable $\boldsymbol{\sigma} = -A\nabla u$ is important for the underlying application, we may approximate it directly through the dual problem: finding $\boldsymbol{\sigma}_n \in \mathcal{M}_n^d(\tau_k)$ such that

$$J^*(\boldsymbol{\sigma}_n) = \min_{\boldsymbol{\tau} \in \mathcal{M}_n^d(\tau_k)} J^*(\boldsymbol{\tau}). \tag{3.4}$$

Theorem 3.3. *Let $\boldsymbol{\sigma} \in H(div;\Omega)$ be the solution of problem (2.7), and let $\boldsymbol{\sigma}_n \in \mathcal{M}_n^d(\tau_k)$ be a solution of (3.4). Then we have*

$$\|\boldsymbol{\sigma} - \boldsymbol{\sigma}_n\|_{a^*} = \inf_{\boldsymbol{\tau} \in \mathcal{M}_n^d(\tau_k)} \|\boldsymbol{\sigma} - \boldsymbol{\tau}\|_{a^*}, \tag{3.5}$$

where $\|\boldsymbol{\tau}\|_{a^*} := \sqrt{a^*(\boldsymbol{\tau},\boldsymbol{\tau})}$ is the energy norm for the dual variable.

Proof. The theorem may be proved in a similar fashion. For any $v \in \mathcal{M}_n^1(\tau_k)$,

$$\|\boldsymbol{\sigma} - \boldsymbol{\sigma}_n\|_{a^*}^2 = 2\left(J^*(\boldsymbol{\sigma}_n) - J^*(\boldsymbol{\sigma})\right) \le 2\left(J^*(\boldsymbol{\tau}) - J^*(\boldsymbol{\sigma})\right) = \|\boldsymbol{\sigma} - \boldsymbol{\tau}\|_{a^*},$$

which implies the validity of (3.5). □

4. Effect of numerical integration

In practice, the integrals of the loss functional are computed numerically. For example, we proposed and implemented the composite mid-point quadrature rule in [3]. To understand the effect of numerical integration, we extend the first Strang lemma for the Galerkin approximation over a subspace (see, e.g., [8]) to the Ritz approximation over a subset.

To this end, denote by $a_\tau(\cdot,\cdot)$ and $f_\tau(\cdot)$ the discrete counterparts of $a(\cdot,\cdot)$ and $f(\cdot)$ through numerical integration. Similarly, $a_\tau^*(\cdot,\cdot)$ and $f_\tau^*(\cdot)$ are the discrete counterparts of $a^*(\cdot,\cdot)$ and $f^*(\cdot)$. Then approximations to the primal and dual variables with numerical integration are seeking $u_\tau \in \mathcal{M}_n^1(\tau_k)$ such that

$$J_\tau(u_\tau) = \min_{v \in \mathcal{M}_n^1(\tau_k)} J_\tau(v), \quad \text{where } J_\tau(v) = \frac{1}{2} a_\tau(v,v) - f_\tau(v) \tag{4.1}$$

and $\boldsymbol{\sigma}_\tau \in \mathcal{M}_n^d(\tau_k)$ such that

$$J_\tau^*(\boldsymbol{\sigma}_\tau) = \min_{\boldsymbol{\tau} \in \mathcal{M}_n^d(\tau_k)} J_\tau^*(\boldsymbol{\tau}), \quad \text{where } J_\tau^*(v) = \frac{1}{2} a_\tau(\boldsymbol{\tau},\boldsymbol{\tau}) - f_\tau(\boldsymbol{\tau}), \tag{4.2}$$

respectively.

Theorem 4.1. *Assume that there exists a positive constant α independent of $\mathcal{M}_{2n}^1(\tau_k)$ such that*

$$\alpha \|v\|_a^2 \le a_{\mathcal{T}}(v,v), \quad \forall\, v \in \mathcal{M}_{2n}^1(\tau_k). \tag{4.3}$$

Let u and $u_{\mathcal{T}}$ be the solutions of (2.3) and (4.1), respectively. Then there exists a positive constant C such that

$$\begin{aligned} &\|u - u_{\mathcal{T}}\|_a \\ &\le C\left(\inf_{v\in\mathcal{M}_n^1(\tau_k)}\left\{\|u-v\|_a + \sup_{\phi\in\mathcal{M}_{2n}^1(\tau_k)}\frac{|a(v,\phi)-a_{\mathcal{T}}(v,\phi)|}{\|\phi\|_a}\right\} + \sup_{\phi\in\mathcal{M}_{2n}^1(\tau_k)}\frac{|f(\phi)-f_{\mathcal{T}}(\phi)|}{\|\phi\|_a}\right). \end{aligned} \tag{4.4}$$

Proof. For any $v \in \mathcal{M}_n^1(\tau_k)$, it is easy to see that $u_{\mathcal{T}} - v \in \mathcal{M}_{2n}^1(\tau_k) \subset V$. By the assumption in (4.3), the definition of $J_{\mathcal{T}}(\cdot)$, and the relations:

$$J_{\mathcal{T}}(u_{\mathcal{T}}) \le J_{\mathcal{T}}(v) \quad \text{and} \quad a(u, u_{\mathcal{T}} - v) = f(u_{\mathcal{T}} - v),$$

we have

$$\begin{aligned} \frac{\alpha}{2}\|u_{\mathcal{T}} - v\|_a^2 &\le \frac{1}{2}a_{\mathcal{T}}(u_{\mathcal{T}} - v, u_{\mathcal{T}} - v) = J_{\mathcal{T}}(u_{\mathcal{T}}) - J_{\mathcal{T}}(v) + f_{\mathcal{T}}(u_{\mathcal{T}} - v) - a_{\mathcal{T}}(v, u_{\mathcal{T}} - v) \\ &\le f_{\mathcal{T}}(u_{\mathcal{T}} - v) - a_{\mathcal{T}}(v, u_{\mathcal{T}} - v) \\ &= \Big(f_{\mathcal{T}}(u_{\mathcal{T}} - v) - f(u_{\mathcal{T}} - v)\Big) + \Big(a(v, u_{\mathcal{T}} - v) - a_{\mathcal{T}}(v, u_{\mathcal{T}} - v)\Big) + a(u - v, u_{\mathcal{T}} - v) \end{aligned}$$

which, together with the Cauchy-Schwarz inequality, implies

$$\|u_{\mathcal{T}} - v\|_a^2 \le C\left(\|u-v\|_a^2 + \sup_{\phi\in\mathcal{M}_{2n}^1(\tau_k)}\frac{|a(v,\phi)-a_{\mathcal{T}}(v,\phi)|}{\|\phi\|_a} + \sup_{\phi\in\mathcal{M}_{2n}^1(\tau_k)}\frac{|f(\phi)-f_{\mathcal{T}}(\phi)|}{\|\phi\|_a}\right).$$

Combining the above inequality with the triangle inequality

$$\|u - u_{\mathcal{T}}\|_a \le \|u - v\|_a + \|v - u_{\mathcal{T}}\|_a$$

and taking the infimum over all $v \in \mathcal{M}_n^1(\tau_k)$ yield (4.4). This completes the proof of the theorem. □

Remark 4.2. For any $\phi_1, \phi_2 \in \mathcal{M}_n^1(\tau_k)$, since $\mathcal{M}_n^1(\tau_k)$ is not a subspace, $\phi_1 - \phi_2$ is generally not in $\mathcal{M}_n^1(\tau_k)$. But it is easy to see that $\phi_1 - \phi_2 \in \mathcal{M}_{2n}^1(\tau_k)$. This is why the assumption in *(4.3)* and the supremum in *(4.4)* are over $\mathcal{M}_{2n}^1(\tau_k)$ but not $\mathcal{M}_n^1(\tau_k)$.

Theorem 4.3. *Assume that there exists a positive constant α^* independent of $\mathcal{M}_{2n}^d(\tau_k)$ such that*

$$\alpha^* \|\boldsymbol{\tau}\|_{a^*}^2 \le a_{\mathcal{T}}^*(\boldsymbol{\tau}, \boldsymbol{\tau}), \quad \forall\, \boldsymbol{\tau} \in \mathcal{M}_{2n}^d(\tau_k). \tag{4.5}$$

Let $\boldsymbol{\sigma}$ and $\boldsymbol{\sigma}_{\mathcal{T}}$ be the solutions of (2.7) and (4.2), respectively. Then there exists a positive constant C such that

$$\begin{aligned} &\|\boldsymbol{\sigma} - \boldsymbol{\sigma}_{\mathcal{T}}\|_{a^*} \\ &\le C\left(\inf_{\boldsymbol{\tau}\in\mathcal{M}_n^d(\tau_k)}\left\{\|\boldsymbol{\sigma}-\boldsymbol{\tau}\|_{a^*} + \sup_{\mathbf{v}\in\mathcal{M}_{2n}^d(\tau_k)}\frac{|a(\boldsymbol{\tau},\mathbf{v})-a_{\mathcal{T}}(\boldsymbol{\tau},\mathbf{v})|}{\|\mathbf{v}\|_{a^*}}\right\} + \sup_{\mathbf{v}\in\mathcal{M}_{2n}^d(\tau_k)}\frac{|f(\mathbf{v})-f_{\mathcal{T}}(\mathbf{v})|}{\|\mathbf{v}\|_{a^*}}\right). \end{aligned} \tag{4.6}$$

Proof. The theorem may be proved in a similar fashion as that of Theorem 4.1. □

5. Estimators

Like adaptive finite element method, *a posteriori* error estimation plays a crucial role for the ANE method. The *a posteriori* error estimator is used for determining if the current approximation is within the target accuracy and *a posteriori* error indicators for determining how many new neurons to be added, where to refine the integration mesh, and how to initialize parameters of new neurons. By following ideas of the *a posteriori* error estimation for the finite element method (see, e.g., [16]), we study the recovery and least-squares estimators in this section.

Let $u_{\mathcal{T}} \in \mathcal{M}_n^1(\tau_k)$ be a solution of (4.1). Define the recovered flux $\hat{\boldsymbol{\sigma}}_{\mathcal{T}} \in \mathcal{M}_n^d(\tau_k)$ satisfying

$$\|D^{-1/2}\big(\hat{\boldsymbol{\sigma}}_{\mathcal{T}} + A\nabla u_{\mathcal{T}}\big)\|_{0,\Omega} = \inf_{\boldsymbol{\tau}\in\mathcal{M}_n^d(\tau_k)}\|D^{-1/2}\big(\boldsymbol{\tau} + A\nabla u_{\mathcal{T}}\big)\|_{0,\Omega}, \tag{5.1}$$

where D is either the identity I, A, or A^2. Then the estimator and indicators are given by

$$\xi = \|D^{-1/2}\big(\hat{\boldsymbol{\sigma}}_{\mathcal{T}} + A\nabla u_{\mathcal{T}}\big)\|_{0,\Omega} \quad \text{and} \quad \xi_K = \|D^{-1/2}\big(\hat{\boldsymbol{\sigma}}_{\mathcal{T}} + A\nabla u_{\mathcal{T}}\big)\|_{0,K}, \quad \forall\, K \in \mathcal{K}_n, \tag{5.2}$$

where $\mathcal{K}_n = \{K\}$ is the physical partition of the domain Ω for the current approximation $u_{\mathcal{T}}$ (see the subsequent section). To analyze the estimator ξ, we make the standard saturation assumption (see, e.g., [9]): there exists a positive constant $\gamma \in [0,1)$ such that

$$\|D^{-1/2}\big(\hat{\boldsymbol{\sigma}}_{\mathcal{T}} + A\nabla u\big)\|_{0,\Omega} \le \gamma\, \|D^{-1/2}A\nabla\big(u - u_{\mathcal{T}}\big)\|_{0,\Omega}. \tag{5.3}$$

Theorem 5.1. *Under the assumption in (5.3), we have*

$$\frac{1}{1+\gamma}\,\xi \le \|D^{-1/2}A\nabla(u-u_{\mathcal{T}})\|_{0,\Omega} \le \frac{1}{1-\gamma}\,\xi. \tag{5.4}$$

Proof. The first inequality in (5.4) is a direct consequence of the triangle inequality and (5.3):

$$\xi \le \|D^{-1/2}(\hat{\boldsymbol{\sigma}}_{\mathcal{T}} + A\nabla u)\|_{0,\Omega} + \|D^{-1/2}A\nabla(u-u_{\mathcal{T}})\|_{0,\Omega} \le (1+\gamma)\,\|D^{-1/2}A\nabla(u-u_{\mathcal{T}})\|_{0,\Omega}.$$

The second inequality in (5.4) may be proved in a similar fashion. □

The first and second inequalities in (5.4) are the so-called global efficiency and reliability bounds, respectively. The reliability bound is used for terminating the adaptive procedure. For the ANE method, the global efficiency bound is sufficient for determining the number of new neurons to be added. This is different from the adaptive finite element method in which a local efficiency bound is preferred.

Another estimator of the recovery type is the least-squares (or dual) estimator defined as follows:

$$\eta = \left(\sum_{K\in\mathcal{K}_n}\eta_K^2\right)^{1/2} = \left(\|A^{-1/2}(\hat{\boldsymbol{\sigma}}_{\mathcal{T}} + A\nabla u_{\mathcal{T}})\|_{0,\Omega}^2 + \|c^{-1/2}(\nabla\cdot\hat{\boldsymbol{\sigma}}_{\mathcal{T}} + cu_{\mathcal{T}} - f)\|_{0,\Omega}^2\right)^{1/2}, \tag{5.5}$$

where η_K is the local indicator given by

$$\eta_K = \left(\|A^{-1/2}(\hat{\boldsymbol{\sigma}}_{\mathcal{T}} + A\nabla u_{\mathcal{T}})\|_{0,K}^2 + \|c^{-1/2}(\nabla\cdot\hat{\boldsymbol{\sigma}}_{\mathcal{T}} + cu_{\mathcal{T}} - f)\|_{0,K}^2\right)^{1/2} \tag{5.6}$$

for all $K\in\mathcal{K}_n$. Let u and $\boldsymbol{\sigma} = -A\nabla u$ be the solutions of (2.3) and (2.7), respectively. Denote the errors by

$$e = u - u_{\mathcal{T}} \quad \text{and} \quad \mathbf{e} = \boldsymbol{\sigma} - \hat{\boldsymbol{\sigma}}_{\mathcal{T}}.$$

It is easy to see that

$$\eta^2 = \|A^{-1/2}(\mathbf{e} + A\nabla e)\|_{0,\Omega}^2 + \|c^{-1/2}(\nabla\cdot\mathbf{e} + c\,e)\|_{0,\Omega}^2,$$

which, together with integration by parts, yields

$$|||e|||^2 + |||\mathbf{e}|||_*^2 = \eta^2 + 2\int_{\partial\Omega} e\,(\mathbf{e}\cdot\mathbf{n})\,ds, \tag{5.7}$$

where $|||\cdot|||$ and $|||\cdot|||_*$ are norms given by

$$|||v|||^2 = \|A^{1/2}\nabla v\|_{0,\Omega}^2 + \|c^{1/2}v\|_{0,\Omega}^2 \quad \text{and} \quad |||\boldsymbol{\tau}|||_*^2 = \|A^{-1/2}\boldsymbol{\tau}\|_{0,\Omega}^2 + \|c^{-1/2}\nabla\cdot\boldsymbol{\tau}\|_{0,\Omega}^2.$$

Theorem 5.2. *Let $\hat{\eta}^2 = \eta^2 + \|u_{\mathcal{T}} - g_D\|_{1/2,\Gamma_D}^2 + \|\hat{\boldsymbol{\sigma}}_{\mathcal{T}}\cdot\mathbf{n} + g_N\|_{-1/2,\Gamma_N}^2$. There exists a constant $C_r > 0$ such that*

$$|||e||| \le C_r\,\hat{\eta} \quad \textit{and} \quad |||\mathbf{e}|||_* \le C_r\,\hat{\eta}. \tag{5.8}$$

Moreover, if $\|u_{\mathcal{T}} - g_D\|_{1/2,\Gamma_D}$ and $\|\hat{\boldsymbol{\sigma}}_{\mathcal{T}}\cdot\mathbf{n} + g_N\|_{-1/2,\Gamma_N}$ are higher order comparing to $|||e|||$ and $|||\mathbf{e}|||_*$, then we have

$$|||e||| \le \eta + h.o.t \quad \textit{and} \quad |||\mathbf{e}|||_* \le \eta + h.o.t. \tag{5.9}$$

Proof. By the definition of the negative norm and the trace inequality, we have

$$\left|\int_{\Gamma_D} e\,(\mathbf{e}\cdot\mathbf{n})\,ds\right| \le \|e\|_{1/2,\Gamma_D}\|\mathbf{e}\cdot\mathbf{n}\|_{-1/2,\Gamma_D} \le C\,\|e\|_{1/2,\Gamma_D}|||\mathbf{e}|||_* \le \frac{1}{4}|||\mathbf{e}|||_*^2 + 2C^2\|e\|_{1/2,\Gamma_D}^2,$$

where C is a constant depending on the diffusion and reaction coefficients A and c. In a similar fashion, we have

$$\left|\int_{\Gamma_N} e\,(\mathbf{e}\cdot\mathbf{n})\,ds\right| \le \frac{1}{4}|||e|||^2 + 2C^2\|\mathbf{e}\cdot\mathbf{n}\|_{-1/2,\Gamma_N}^2.$$

Hence, we have

$$\left|2\int_{\partial\Omega} e\,(\mathbf{e}\cdot\mathbf{n})\,ds\right| \le \frac{1}{2}\left(|||e|||^2 + |||\mathbf{e}|||_*^2\right) + C\left(\|u_{\mathcal{T}} - g_D\|_{1/2,\Gamma_D}^2 + \|\hat{\boldsymbol{\sigma}}_{\mathcal{T}}\cdot\mathbf{n} + g_N\|_{-1/2,\Gamma_N}^2\right)$$

which, together with (5.7), implies (5.8).

If $\|u_{\mathcal{T}} - g_D\|_{1/2,\Gamma_D}$ and $\|\hat{\boldsymbol{\sigma}}_{\mathcal{T}}\cdot\mathbf{n} + g_N\|_{-1/2,\Gamma_N}$ are higher order comparing to $|||e|||$ and $|||\mathbf{e}|||_*$, so is $\left|2\int_{\partial\Omega} e\,(\mathbf{e}\cdot\mathbf{n})\,ds\right|$. Now, (5.9) is a direct consequence of (5.7). This completes the proof of the theorem. □

Remark 5.3. The assumption in *Theorem 5.2* on the boundary errors is made based on numerical results for test problems in *Section 8*. Currently, we are not able to verify them theoretically.

6. Adaptive neuron enhancement (ANE) method

Let $u(\mathbf{x})$ and $u_{\mathcal{T}}(\mathbf{x},\theta^*_{\mathcal{T}}) \in \mathcal{M}_n^1(\tau_k)$ be the solutions of (2.3) and (4.1), respectively. For a given tolerance $\epsilon > 0$, this section describes the ANE method (see [1] for the best least-squares approximation) to generate a two-layer spline NN such that the approximation accuracy is within the prescribed tolerance, i.e.,

$$\|u - u_{\mathcal{T}}\|_a \le \epsilon \|u_{\mathcal{T}}\|_a. \tag{6.1}$$

For simplicity of presentation, assume that the numerical integration based on a partition $\mathcal{T}$ is sufficiently accurate (see [1] on how to adaptively generating a numerical integration mesh).

The procedure of the ANE method is similar to the widely used adaptive mesh refinement (AMR) method for traditional, well-studied mesh-based numerical methods. Unlike the mesh-based methods, the NN method is based on the NN structure determined by the number of neurons in the case of two-layer NNs. This observation suggests that the key question for developing the ANE method is: how many new neurons will be added at each adaptive step?

To address this question, we introduce the concept of the physical partition of the domain Ω for a function in $\mathcal{M}_n^1(\tau_k)$. To this end, let

$$u_{\mathcal{T}}(\mathbf{x},\theta^*_{\mathcal{T}}) = c_0 + \sum_{i=1}^{n} c_i \tau_k(\boldsymbol{\omega}_i \cdot \mathbf{x} - b_i).$$

The physical partition for $u_{\mathcal{T}}(\mathbf{x},\theta^*_{\mathcal{T}})$ is determined by the n hyper-planes $\{\boldsymbol{\omega}_i \cdot \mathbf{x} - b_i = 0\}_{i=1}^n$ plus the boundary of the domain Ω, and is denoted by $\mathcal{K}_n = \{K\}$. Clearly, $\mathcal{K}_n$ forms a partition of the domain Ω; i.e., the union of all subdomains in $\mathcal{K}_n$ equals the whole domain Ω and that any two distinct subdomains of $\mathcal{K}_n$ have no intersection.

For each element $K \in \mathcal{K}_n$, denote by ξ_K the local indicator defined in either (5.2) or (5.5). We then define a subset $\hat{\mathcal{K}}_n$ of $\mathcal{K}_n$ by using either the following average marking strategy:

$$\hat{\mathcal{K}}_n = \left\{ K \in \mathcal{K}_n \,:\, \xi_K \ge \frac{1}{\#\mathcal{K}_n} \sum_{K \in \mathcal{K}_n} \xi_K \right\}, \tag{6.2}$$

where $\#\mathcal{K}_n$ is the number of elements of $\mathcal{K}_n$, or the bulk marking strategy: finding a minimal subset $\hat{\mathcal{K}}_n$ of $\mathcal{K}_n$ such that

$$\sum_{K \in \hat{\mathcal{K}}_n} \xi_K^2 \ge \gamma_1 \sum_{K \in \mathcal{K}_n} \xi_K^2 \quad \text{for} \quad \gamma_1 \in (0,1). \tag{6.3}$$

With the subset $\hat{\mathcal{K}}_n$, the number of new neurons to be added to the NN is equal to the number of elements in $\hat{\mathcal{K}}_n$. With an accurate numerical integration, the ANE method is defined in Algorithm 6.1.

Algorithm 6.1 Adaptive Neuron Enhancement Method.

Given a tolerance $\epsilon > 0$ and a numerical integration mesh $\mathcal{T}$, starting with a two-layer spline NN with a small number of neurons,

(1) solve the optimization problem in (4.1);
(2) estimate the total error by computing $\xi = \left(\sum_{K \in \mathcal{K}} \xi_K^2 \right)^{1/2}$;
(3) if $\xi < \epsilon \|u_{\mathcal{T}}\|_a$, then stop; otherwise, mark $\hat{\mathcal{K}}_n$ using (6.2) or (6.3), go to Step (4);
(4) add $\#\hat{\mathcal{K}}_n$ neurons to the network, then go to Step (1).

7. Initialization

The high dimensional, non-convex optimization problem in (4.1) is often solved by iterative optimization methods such as gradient descent (GD), Stochastic GD, Adam, etc. (see, e.g., [17] for a review paper in 2018 and references therein). Usually nonlinear optimizations have many solutions, and the desired one is obtained only if we start from a close enough first approximation. The ANE method provides a natural process for obtaining a good initialization.

We employ the initialization approach introduced for the best least-squares approximation in [1]. For readers' convenience, we briefly describe it here. First, we specify the size of the initial NN and its input and output weights and bias. Input weights and bias are chosen so that the corresponding hyper-planes form a uniform partition of the domain Ω. The output weights and bias is chosen as the solution of the system of linear equations to be given in (7.1).

When the NN is enhanced by adding new neurons in Step (4) of Algorithm 6.1, clearly, parameters corresponding to old neurons will use the current approximation as their initial. Each new neuron is associated with a subdomain $K \in \hat{\mathcal{K}}_n$ and its initial is chosen so that the corresponding hyper-plane passes through the centroid of K and orthogonal to the direction vector with the smallest variance of quadrature points in K. This direction vector may be computed by the Principal Component Analysis method (or PCA [18]). For output weights and biases corresponding to new neurons, a simple initial is to set them zero. This means that the initial of the approximation is the current approximation. A better way is to solve problem (7.1) for all output weights and bias.

In the remainder of this section, we describe the system of algebraic equations, that determines the initial of the output weights and bias when the corresponding hyper-planes are fixed. Denote by $\boldsymbol{\omega}^0 = (\boldsymbol{\omega}_1^0, \ldots, \boldsymbol{\omega}_n^0)$ and $\mathbf{b}^0 = (b_1^0, \ldots, b_n^0)$ the initial of the input weights and bias, respectively. Let

$$u_{\mathcal{T}}^0(\mathbf{x}) = c_0^0 + \sum_{i=1}^{n} c_i^0 \tau_k(\boldsymbol{\omega}_i^0 \cdot \mathbf{x} - b_i^0) \equiv \sum_{i=0}^{n} c_i^0 \varphi_i(\mathbf{x}; \boldsymbol{\omega}_i^0, b_i^0)$$

be the initial approximation to the solution, $u_{\mathcal{T}}(\mathbf{x}) \in \mathcal{M}_n^1(\tau_k)$, of (4.1). Then $\mathbf{c}^0 = (c_0^0, c_1^0, \ldots, c_n^0)$ is the solution of the following algebraic equations

$$a_{\mathcal{T}}(u_{\mathcal{T}}^0, \varphi_i) = f_{\mathcal{T}}(\varphi_i) \quad \text{for } i = 0, 1, ..., n. \tag{7.1}$$

Lemma 7.1. *Assume that the hyper-planes* $\{\boldsymbol{\omega}_i^0 \cdot \mathbf{x} = b_i^0\}_{i=1}^n$ *are distinct. Then the stiffness matrix* $\mathbf{K} = \left(a_{\mathcal{T}}(\varphi_i(\cdot; \boldsymbol{\omega}_i^0, b_i^0), \varphi_j(\cdot; \boldsymbol{\omega}_j^0, b_j^0))\right)_{(n+1)\times(n+1)}$ *is symmetric, and positive definite.*

Proof. Clearly, $\mathbf{K}$ is symmetric. For any $\mathbf{v} = (v_0, v_1, ..., v_n)^t$, we have

$$\mathbf{v}^t \mathbf{K} \mathbf{v} = a_{\mathcal{T}}(v, v),$$

where $v(\mathbf{x}) = \sum_{i=0}^{n} v_i \varphi_i(\mathbf{x}; \boldsymbol{\omega}_i^0, b_i^0)$. Since $\{\varphi_i\}_{i=0}^n$ are linearly independent (see Lemma 2.1 in [1]) when the hyper-planes $\{\boldsymbol{\omega}_i^0 \cdot \mathbf{x} = b_i^0\}_{i=1}^n$ are distinct, $a_{\mathcal{T}}(v, v)$ is positive for any nonzero $\mathbf{v}$, which, in turn, implies that $\mathbf{K}$ is positive definite. □

Remark 7.2. If there are two hyper-planes are almost linearly dependent, then the stiffness matrix $\mathbf{K}$ is ill-conditioned even though it is symmetric, positive definite. This is because basis function $\varphi_i(\mathbf{x}; \boldsymbol{\omega}_i^0, b_i^0) = \tau_k(\boldsymbol{\omega}_i^0 \cdot \mathbf{x} - b_i^0)$ has a non-local support.

In one dimension, this difficulty may be overcome by transforming the non-local basis functions to the local nodal basis functions and solving (7.1) using the local basis function. More specifically, assume that $b_0^0 < b_1^0 < \cdots < b_n^0$. Denote by $h_i = b_{i+1} - b_i$ the length of subinterval $[b_i, b_{i+1}]$, and set

$$l_i(\mathbf{x}) = h_i^{-1}\varphi_i - \left(h_i^{-1} + h_{i+1}^{-1}\right)\varphi_{i+1} + h_{i+1}^{-1}\varphi_{i+2} \quad \text{for } i = 1, ..., n-2,$$

$$l_{n-1}(\mathbf{x}) = h_{n-1}^{-1}\left(\varphi_{n-1} - \varphi_n\right), \quad l_n(\mathbf{x}) = h_n^{-1}\varphi_n, \quad \text{and} \quad l_0(\mathbf{x}) = \varphi_0 - \sum_{i=1}^{n-1} l_i(\mathbf{x}).$$

Let $\hat{\mathbf{c}} = (\hat{c}_0, \hat{c}_1, ..., \hat{c}_n)^t$ be the solution of

$$\sum_{i=0}^{n} \hat{c}_j a_{\mathcal{T}}(l_j, l_i) = f_{\mathcal{T}}(l_i) \quad \text{for } i = 0, 1, ..., n,$$

and let T be the linear transformation from $\boldsymbol{\varphi} = (\varphi_0, ..., \varphi_n)^t$ to $\mathbf{l} = (l_0, ..., l_n)^t$. Then

$$\mathbf{c}^0 = (c_0^0, ..., c_n^0)^t = T^t \hat{\mathbf{c}}. \tag{7.2}$$

8. Numerical experiments

In this section, we present our numerical results on using the adaptive neuron enhancement (ANE) method to solve diffusion problems based on the Ritz approximation. In all experiments, the minimization problems are iteratively solved by the Adam optimizer [19]. The integrals of the energy functionals are computed numerically using composite mid-point quadrature rules with uniformly distributed quadrature points. For each run during the adaptive enhancement process, the training stops when the value of the energy functional decreases within 0.1% in the last 2000 iterations. And the ANE process stops when the user specified accuracy tolerance ε is obtained, where the error is estimated using the relative recovery error estimator $\xi_{\text{rel}} = \xi / \|\hat{\boldsymbol{\sigma}}_{\mathcal{T}}\|_0$, where $\|\hat{\boldsymbol{\sigma}}_{\mathcal{T}}\|_0 \approx \|u_{\mathcal{T}}\|_a$.

8.1. One-dimensional Poisson equation

The first test problem (see [20,3,1]) is a one-dimensional Poisson equation with homogeneous Dirichlet boundary condition defined on the unit interval $\Omega = (0, 1)$. For $f(x) = -40000(x^3 - 2x^2/3 + 173x/1800 + 1/300)e^{-100(x-1/3)^2}$, the exact solution of the test problem is given by

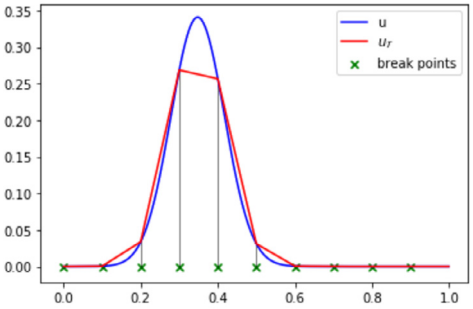
$$u(x) = x\left(e^{-(x-\frac{1}{3})^2/0.01} - e^{-\frac{4}{9}/0.01}\right).$$

A fixed numerical integration partition $\mathcal{T}$ of 1000 uniformly distributed quadrature points is utilized to calculate the energy functional in (2.2) with $\gamma_D = 2000$. For training (optimizing) this primary problem in (2.3), the learning rate of the Adam optimizer is fixed at 0.002.

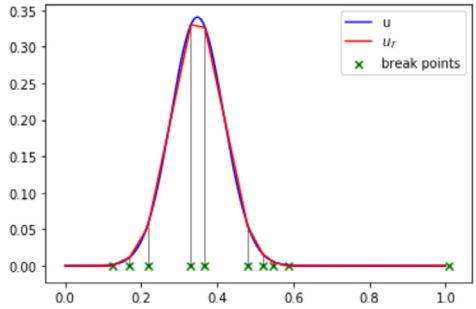
We start from 10 neurons with their breakpoints distributed uniformly and then solve the linear system in (7.1) for the initial of the output weights and bias; the initial NN model of $u_{\mathcal{T}}$ is depicted in Fig. 1(a). After optimizing all the parameters in the network, the 10 breakpoints move themselves and the NN outputs an optimized model of non-uniformly distributed breakpoints as shown in Fig. 1(b).

Local error indicator ξ_K is calculated using the recovered $\sigma_{\mathcal{T}}$ from $-u_{\mathcal{T}}'$ (see Fig. 1(c) and 1(e) for a graphical illustration). Elements with large errors are marked by the average marking strategy (see (5.4) in [1]) and the corresponding neurons are added with proper initialization. This adaptive process repeats itself three runs until our target approximation accuracy $\varepsilon = 0.08$ is reached. Fig. 1(d) shows the final approximation of adaptive two-layer ReLU NN with 25 neurons.

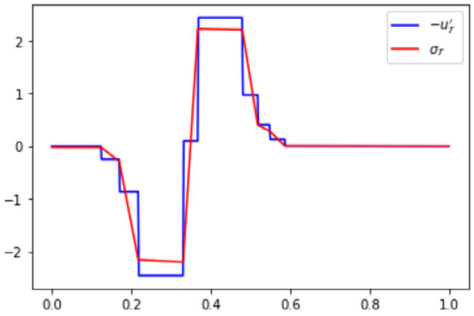
For comparison, we also report numerical results using fixed two-layer ReLU NNs with 25 and 50 neurons. Table 1 clearly shows that the accuracy of the adaptive ReLU NN is about the same as that of the fixed NN with twice parameters. The approximation of the fixed NN with 25 neurons is depicted in Fig. 1(f); the fact that, only 17 out of 25 neurons contribute to the approximation, explains why the fixed NN is not as accurate as the adaptive NN. Finally, we report numerical results from our previous paper [3] using an over-parametrized DNN of four layers in the last row of Table 1. Even though the over-parametrized DNN is powerful in approximation, attainable approximation may not be as accurate as that of a proper adaptive/fixed NN with significant less parameters due to the difficulty of non-convex optimization.



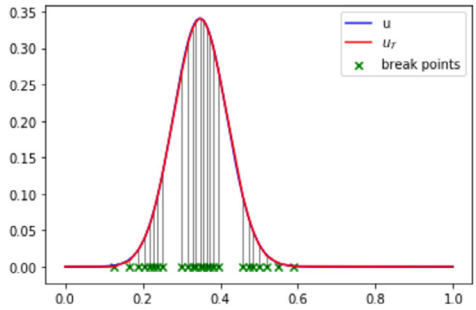
(a) Initial model $u_{\mathcal{T}}$ with 10 neurons $\frac{\|u'-u'_{\mathcal{T}}\|_0}{\|u'\|_0}$=0.522380



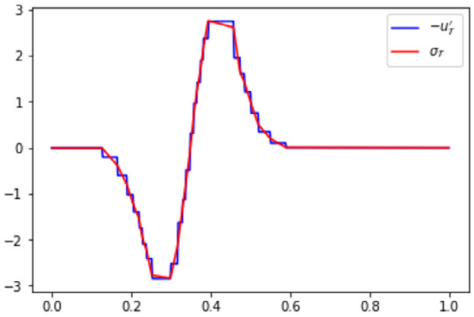
(b) Optimized model $u_{\mathcal{T}}$ with 10 neurons, $\frac{\|u'-u'_{\mathcal{T}}\|_0}{\|u'\|_0}$=0.229533



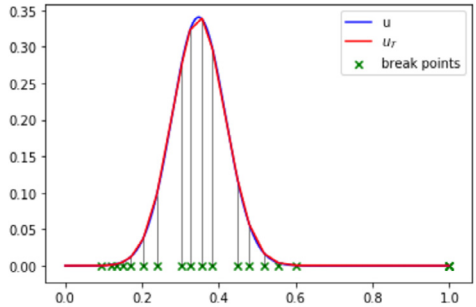
(c) Recovered flux $\sigma_{\mathcal{T}}$ and the calculated $-u'_{\mathcal{T}}$ of 10 neurons, $\frac{\|\sigma_{\mathcal{T}}+u'_{\mathcal{T}}\|_0}{\|\sigma_{\mathcal{T}}\|_0}$=0.278647



(d) Adaptive model $u_{\mathcal{T}}$ with 25 neurons, $\frac{\|u'-u'_{\mathcal{T}}\|_0}{\|u'\|_0}$=0.075847



(e) Recovered flux $\sigma_{\mathcal{T}}$ and the calculated $-u'_{\mathcal{T}}$ of 25 neurons, $\frac{\|\sigma_{\mathcal{T}}+u'_{\mathcal{T}}\|_0}{\|\sigma_{\mathcal{T}}\|_0}$=0.076366



(f) A fixed model $u_{\mathcal{T}}$ with 25 neurons, $\frac{\|u'-u'_{\mathcal{T}}\|_0}{\|u'\|_0}$=0.151279

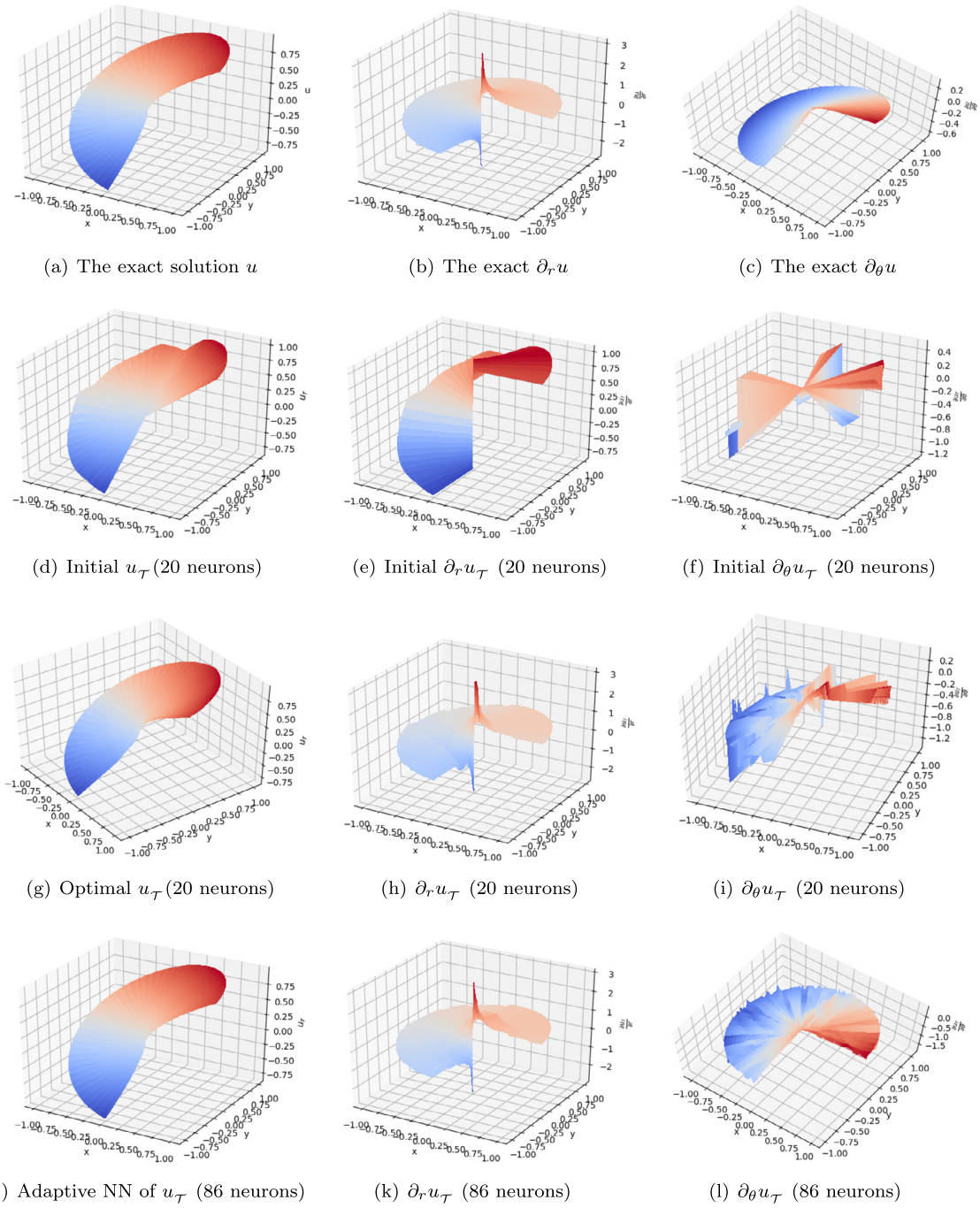
Fig. 1. Poisson equation approximation results using energy functional as the loss function.

Table 1
Poisson equation: comparing adaptive network with fixed networks using Energy functional.

NN (hidden layer neurons)	#Parameters	$\frac{\|u-u_\tau\|_0}{\|u\|_0}$	$\frac{\|u'-u'_\tau\|_0}{\|u'\|_0}$	$\xi_{\text{rel}}=\frac{\|\sigma_\tau+u'_\tau\|_0}{\|\sigma_\tau\|_0}$
Fixed 2-layer (25)	51	0.012943	0.149020	0.164645
Fixed 2-layer (50)	101	**0.006108**	0.089470	0.095394
Adaptive 2-layer (25)	51	0.007794	**0.075847**	0.076366
Fixed 4-layer (24-14-14) [3]	623	0.029161	0.160666	-

8.2. Two-dimensional Poisson equation with re-entrant corner

The second test problem is a two-dimensional Poisson equation with pure Dirichlet boundary condition defined on a domain with re-entrant corner $\Omega=\{(r,\theta)|\, r\in(0,1),\ \theta\in(0,\frac{3\pi}{2})\}$. The exact solution



(a) The exact solution u (b) The exact $\partial_r u$ (c) The exact $\partial_\theta u$

(d) Initial $u_{\mathcal{T}}$(20 neurons) (e) Initial $\partial_r u_{\mathcal{T}}$ (20 neurons) (f) Initial $\partial_\theta u_{\mathcal{T}}$ (20 neurons)

(g) Optimal $u_{\mathcal{T}}$(20 neurons) (h) $\partial_r u_{\mathcal{T}}$ (20 neurons) (i) $\partial_\theta u_{\mathcal{T}}$ (20 neurons)

(j) Adaptive NN of $u_{\mathcal{T}}$ (86 neurons) (k) $\partial_r u_{\mathcal{T}}$ (86 neurons) (l) $\partial_\theta u_{\mathcal{T}}$ (86 neurons)

Fig. 2. Poisson equation with re-entrant corner: Exact solution and results of adaptive two-layer ReLU NN from 20 to 86 neurons.

$$u(r,\theta) = r^{\frac{2}{3}} \sin(\frac{2\theta+\pi}{3}),$$

is harmonic, i.e., $\Delta u = 0$.

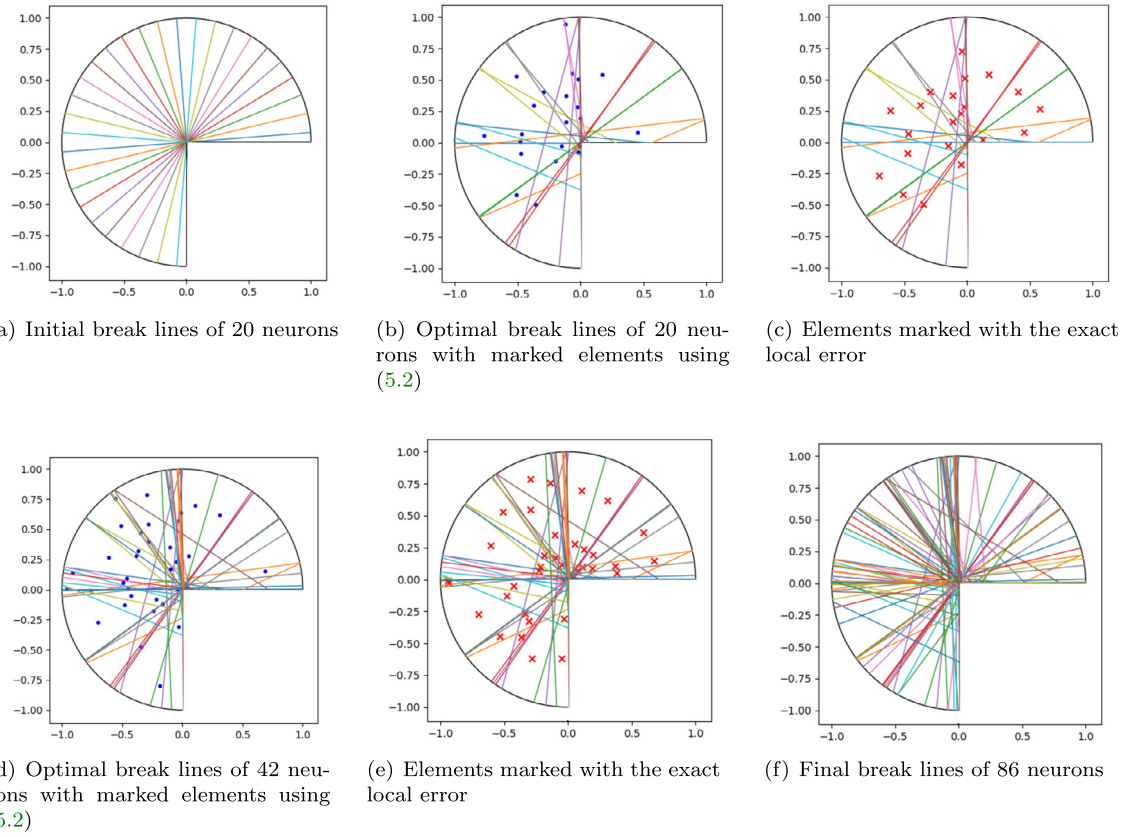
Numerical integration is calculated using a partition $\mathcal{T}$ with 50×270 quadrature points uniformly distributed along radial and circumferential directions in a polar coordinate framework. The γ_D is set as 200 and the learning rate of the optimizer is fixed at 0.001. The target approximation accuracy is set as $\varepsilon = 0.15$ for this problem considering the difficulty posed by the point singularity at the origin.

Our adaptive model starts from 20 neurons which are initialized such that the corresponding break lines are distributed uniformly along circumferential direction as shown in Fig. 3(a). The initial NN model obtained after solving (7.1) is illustrated in Fig. 2(d). This initial model gives a fair approximation of u and the relative error in the L^2 norm is 0.13, while approximation to ∇u (see Fig. 2(e) and 2(f)) still presents relatively large errors (the relative error in the energy norm is 0.49). After optimization in the first run, the break lines of these 20 neurons move and form a non-uniform partition of the domain as shown in Fig. 3(b), which results in an NN model of improved performance, see Table 2, first row for the numerical results. The graphical results of $u_{\mathcal{T}}$ and $\nabla u_{\mathcal{T}}$ approximate by a NN of 20 neurons are depicted in Fig. 2(g) - Fig. 2(i).

During the neuron enhancement step, elements with relative large local error ξ_K are marked using the bulk marking strategy with $\gamma_1 = 0.5$ (see (5.5) in [1]). After two runs of the ANE, adding 22 and 44 neurons respectively, the ANE process stops at 86 neurons with a relative recovery error

Table 2
Poisson equation with re-entrant corner: comparing adaptive ReLU NN with a fixed NN.

NN (neurons)	#Parameters	$\frac{\|u-u_\tau\|_0}{\|u\|_0}$	$\frac{\|\nabla(u-u_\tau)\|_0}{\|\nabla u\|_0}$	$\xi_{\text{rel}}=\frac{\|\boldsymbol{\sigma}_\tau+\nabla u_\tau\|_0}{\|\boldsymbol{\sigma}_\tau\|_0}$
Adaptive 2-layer (20)	61	0.033947	0.230226	0.233191
Adaptive 2-layer (42)	127	0.021939	0.154129	0.158150
Adaptive 2-layer (86)	259	**0.014162**	**0.064652**	0.132546
Fixed 2-layer (86)	259	0.025932	0.217009	0.162507



(a) Initial break lines of 20 neurons

(b) Optimal break lines of 20 neurons with marked elements using (5.2)

(c) Elements marked with the exact local error

(d) Optimal break lines of 42 neurons with marked elements using (5.2)

(e) Elements marked with the exact local error

(f) Final break lines of 86 neurons

Fig. 3. Poisson equation with re-entrant corner: break lines generated in the ANE process.

estimator $\xi_{\text{rel}}=0.13$. Intermediate results are recorded in Table 2 and the final visual results of u_τ and ∇u_τ approximated by a NN of 86 neurons are illustrated in Fig. 2(j) - Fig. 2(l).

Comparing to a two-layer fixed ReLU NN with the same number of neurons in the hidden layer, the proposed adaptive method converges to a better approximation result (see the last two rows in Table 2). This experiment also shows that for a Poisson equation containing a corner singularity, a two-layer ReLU NN is capable of generating a good approximation to the solution. Fig. 3(e) shows the final arrangement generated by the adaptive ReLU NN of 86 neurons in which we observe that the break lines adapt themselves to account for the singular point at the origin. This self-adaptivity of generated physical meshes is a highly desirable feature of using NN model to approximate problems with singularities or discontinuities.

To verify that the proposed error estimator of the recovery type provides a valid indicator for neuron enhancement, we compare the elements marked using the proposed error indicator (5.2) and those marked using the exact error, the element marking results for the two intermediate runs in the adaptive process are illustrated in Fig. 3(b)-3(e). The comparison results show similar sets of elements being marked for further enhancement, which indicates the validity of the proposed recovery error indicator.

8.3. Two dimensional interface problem

The third test problem is the intersecting interface problem defined on the unit disk $\Omega=\{(r,\theta)|\, r\in[0,1),\theta\in[0,2\pi])\}$ and satisfying (2.1) with $c=f=0$, $\Gamma_N=\emptyset$, and $A=\alpha(\theta)I$, where the diffusion coefficient $\alpha(\theta)$ equals to 1 in the second and fourth quadrants and $R=161.4476387975881$ in the first and third quadrants. This is a difficult benchmark test problem for adaptive mesh refinement (see, e.g., [21,22]) and the exact solution is $u(r,\theta)=r^\beta\mu(\theta)$ with

$$\mu(\theta)=\begin{cases}\cos((\pi/2-\sigma)\beta)\cdot\cos((\theta-\pi/2+\rho)\beta), & \text{if}\quad 0\le\theta\le\pi/2,\\ \cos(\rho\beta)\cdot\cos((\theta-\pi+\sigma)\beta), & \text{if}\quad \pi/2\le\theta\le\pi,\\ \cos(\theta\beta)\cdot\cos((\theta-\pi-\rho)\beta), & \text{if}\quad \pi\le\theta\le 3\pi/2,\\ \cos((\pi/2-\rho)\beta)\cdot\cos((\theta-3\pi/2-\sigma)\beta), & \text{if}\quad 3\pi/2\le\theta\le 2\pi.\end{cases}$$

Table 3
Interface Problem: comparing adaptive network with fixed networks using the Ritz formulation.

NN (neurons)	#Para.	$\frac{\|u-\tilde{u}_\tau\|_0}{\|u\|_0}$	$\frac{\|A^{1/2}\nabla(u-u_\tau)\|_0}{\|A^{1/2}\nabla u\|_0}$	$\xi_{\text{rel}}=\frac{\|(\sigma_\tau+A\nabla u_\tau)\|_0}{\|\sigma_\tau\|_0}$
Adaptive (20)	61	0.212283	0.985976	0.817516
Adaptive (31)	94	0.168116	0.760418	0.749148
Adaptive (52)	157	0.129946	0.546362	0.714189
Adaptive (92)	277	0.107181	0.362692	0.634297
Adaptive (150)	451	0.087608	**0.047335**	0.549564
Fixed (150)	451	0.160656	0.826836	0.717022
Fixed (20-20)	481	**0.070198**	0.624581	0.535260

Considering the inherent difficulty introduced by the intersecting interfaces along x-axis and y-axis and the fact that the recovery estimator over-estimate the true error, we set the stopping criterion as $\xi_{\text{rel}} \le \epsilon = 0.6$. Numerical integration is calculated on an uniform partition $\mathcal{T}$ of 50×360 quadrature points. The γ_D is set at 200 and a constant learning rate of 0.001 is used throughout the training. For the error estimator of recovery type in (5.2), we use the identity matrix for D. A bulk marking strategy is adopted in the adaptive process with $\gamma_1 = 0.7$ (see (5.5) of [1]).

Again, we start from a small size NN of 20 neurons, the adaptive process enhances four runs and stops at 150 neurons with the relative error estimator $\xi_{\text{rel}} = 0.55 < \epsilon$. The initial NN model, the optimized NN model of 20 neurons and the final model of 150 neurons are all illustrated in Fig. 4, and the values of the relative error estimator in each run, from 20 neurons to 150 neurons, are recorded in Table 3.

As shown in Table 3, the adaptive model of 150 neurons yields a better approximation than the fixed model of the same size. Comparing to the adaptive finite element method adopted in [22] which uses more than three thousands of grid points (parameters) in an adaptive refined mesh to achieve a similar result in the relative energy norm, the adaptive ReLU NN presents a more efficient model since all break lines are allowed to move and to adapt to the characteristics of the target function. In this problem, the singularity at the origin and the intersecting interface along axes are captured well through the moving break lines, see Fig. 4(g) for the optimized arrangements generated by the break lines of those 150 neurons.

We notice that the final adaptive model of u_τ contains certain degrees of oscillation on the boundary (see Fig. 4(h)). To seek a remedy, we further test a fixed three-layer ReLU NN with 20 neurons in each hidden layer. The oscillation is reduced (see Fig. 4(l)) and the relative error in the L^2 norm is also improved. The reduction in oscillation is presumably due to the facts that the neurons in deeper layers provide fine scale approximation while those in the first hidden layer provide only coarse scale approximation. Nevertheless, it is also noticed that the adaptive two-layer NN achieves better accuracy in the energy norm comparing to the fixed three-layer NN of similar size (see the last three rows in Table 3). This result is perhaps caused by training of difficult nonlinear optimization problem, because a good initialization is systematically provided through our ANE method for adaptive two-layer NN, but is not available for the fixed three-layer NN. To extend our ANE method to multi-layer NN is our current research project and requires a deeper understanding on the role of depth in a neural network.

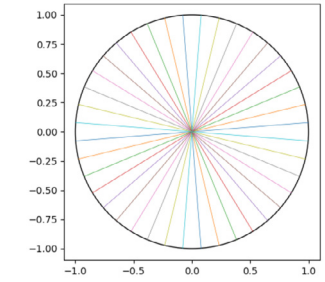
9. Discussion and conclusion

To adaptively construct a two-layer spline NN with a nearly minimum number of neurons and parameters such that its approximation accuracy is within the prescribed tolerance, we develop and test the adaptive neuron enhancement (ANE) method for the Ritz approximation to elliptic PDEs in this paper. A key component of the ANE method for its application in PDEs is the development of computable local indicators since the solution of a PDE is unknown. The recovery and the least-squares estimators are introduced. Numerical results for the ReLU NN approximation to problems with corner or intersecting interface singularities show that the recovery estimator is effective. When using other activation functions, the recovery estimator need to be modified by adding weighted L^2 norm of the residual of the original equation (see the hybrid estimator in [23,10]) since the recovery estimator may not be reliable. The least-squares estimator provides a constant free, guaranteed upper bound of the true error in the energy norm, and hence it is useful to serve as a stopping criterion.

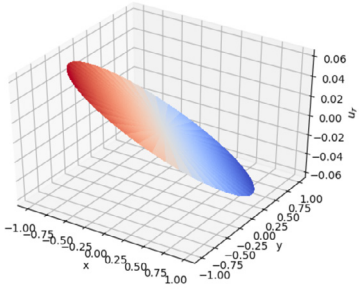
When a PDE has an underlying minimization principle, our experience suggests that the Ritz formulation is better than various manufactured least-squares formulations as stated in the introduction due to the number of independent variables and the smoothness of the solution. Moreover, a loss functional with fewer independent variables is easier to train than one having more variables. Unlike existing NN methods, we approximate the integral of the loss functional by numerical integration. Effect of numerical integration is analyzed for both approximations to a given function and PDE (see Theorem 4.1 of [1] and Theorem 4.3 of this paper).

Universal approximation theorem shows that a two-layer ReLU NN is able to accurately approximate any continuous function defined on a compact set in $\mathbb{R}^d$ provided that there are enough neurons in the NN. Indeed, our numerical results demonstrate that problems with corner or intersecting interface singularities may be approximated accurately by the ANE method with fewer degrees of freedom than that of adaptive finite element method. On the other hand, numerical results for problems with sharp circular transition layer [1] and discontinuous solution [24] show that a three-layer NN is needed in order to approximate the target function well without undesired oscillation. Extension of our ANE method to multi-layer ReLU NN will be presented in [25].

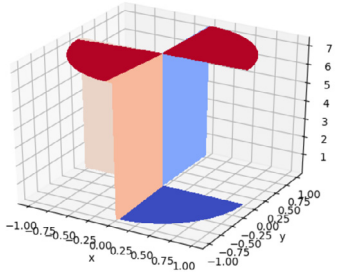
The procedure of training the value of the parameters is a problem in non-convex optimization which usually has many solutions and are complicated and computationally demanding. As discussed in [1], the method of continuation [26] is a common way to obtain a good initial and the ANE is a natural continuation process by itself with respect to the number of neurons. Numerical results show that this method is effective for reducing the number of the parameters of the network. Moreover, a good initial is very helpful in training as well. Nevertheless, training is still a challenging problem since the learning rate of the methods of the gradient type is difficult to tune. A reasonably good learning rate can only be discovered through the method of trial and error. Using NNs to solve PDEs is relatively new, developing fast solvers is an open and challenging problem and requires lots of efforts from numerical analysts. Because of its great potential and many difficulties at the same time, scientific machine learning is a hot research topic in scientific computing.



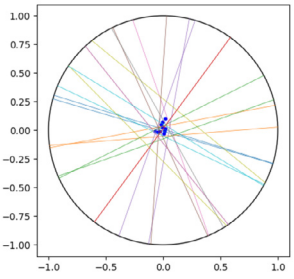
(a) Initial break lines (20 neurons)



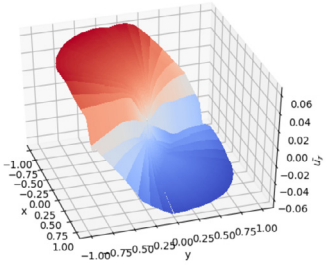
(b) Initial $u_{\mathcal{T}}$ (20 neurons)



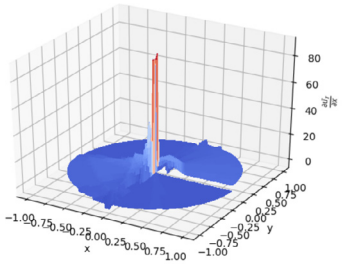
(c) Initial $-\alpha\partial_x u_{\mathcal{T}}$ (20 neurons)



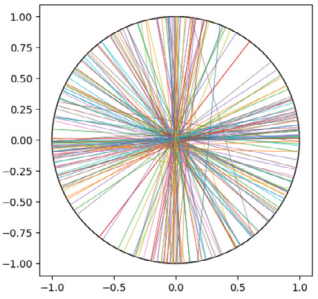
(d) Optimal break lines (20 neurons) with marked elements



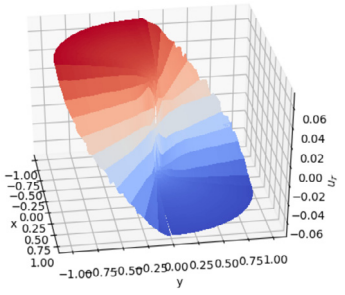
(e) Optimal $u_{\mathcal{T}}$ (20 neurons)



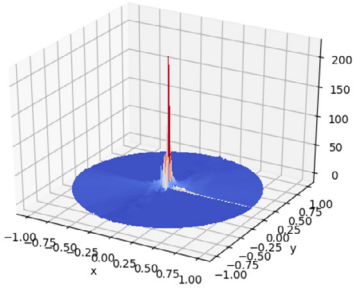
(f) Optimal $-\alpha\partial_x u_{\mathcal{T}}$ (20 neurons)



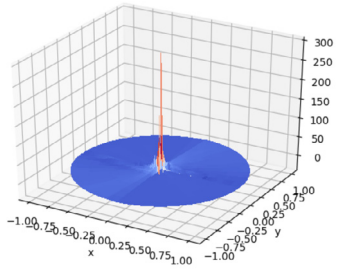
(g) Final adaptive NN of 150 neurons: break lines



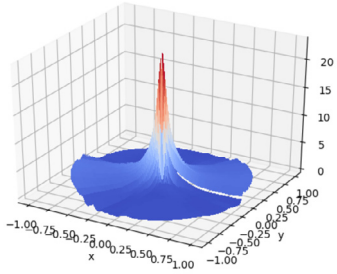
(h) Adaptive model of $u_{\mathcal{T}}$ (150 neurons)



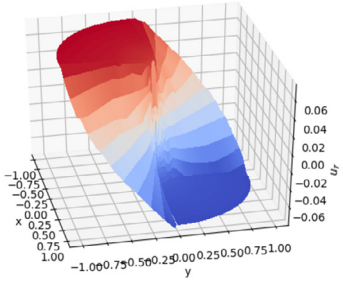
(i) Final $-\alpha\partial_x u_{\mathcal{T}}$ (150 neurons)



(j) Final $-\alpha\partial_y u_{\mathcal{T}}$ (150 neurons)



(k) Recovered flux $\boldsymbol{\sigma}_{x\mathcal{T}}$ (150 neurons)



(l) Fixed NN model of $\bar{u}_{\mathcal{T}}$ with two hidden layers(20 neurons in each)

Fig. 4. Kellogg problem: Results of an adaptive 2-layer ReLU NN and a fixed 3-layer ReLU NN.

References

[1] M. Liu, Z. Cai, J. Chen, Adaptive two-layer ReLU neural network: I. Best least-squares approximation, Comput. Math. Appl. (2022), https://doi.org/10.1016/j.camwa.2022.03.005, in press.

[2] J. Berg, K. Nystrom, A unified deep artificial neural network approach to partial differential equations in complex geometries, Neurocomputing 317 (2018) 28–41.

[3] Z. Cai, J. Chen, M. Liu, X. Liu, Deep least-squares methods: an unsupervised learning-based numerical method for solving elliptic pdes, J. Comput. Phys. 420 (2020) 109707.

[4] W. E, B. Yu, The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems, Commun. Math. Stat. 6 (1) (2018) 1–12.

[5] M. Raissia, P. Perdikarisb, G. Karniadakisa, Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707.

[6] J. Sirignano, K. Spiliopoulos, DGM: a deep learning algorithm for solving partial differential equations, J. Comput. Phys. 375 (2018) 1139–1364.
[7] J. Xu, The finite neuron method and convergence analysis, Commun. Comput. Phys. 28 (2020) 1707–1745.
[8] P. Ciarlet, The Finite Element Method for Elliptic Problems, Society for Industrial and Applied Mathematics, 1978.
[9] R.E. Bank, A. Weiser, Some a posteriori error estimators for elliptic partial differential equations, Math. Comput. 44 (170) (1985) 283–301.
[10] Z. Cai, S. Zhang, Flux recovery and a posteriori error estimators: conforming elements for scalar elliptic equations, SIAM J. Numer. Anal. 48 (2010) 578–602.
[11] G. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Syst. 2 (1989) 303–314.
[12] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Netw. 2 (1989) 359–366.
[13] P.P. Petrushev, Approximation by ridge functions and neural networks, SIAM J. Math. Anal. 30 (1998) 155–189.
[14] J.W. Siegel, J. Xu, High-order approximation rates for neural networks with ReLU^k activation functions, arXiv preprint arXiv:2012.07205, 2020.
[15] W. E, S. Wojtowytsch, Some observations on partial differential equations in Barron and multi-layer spaces, arXiv:2012.01484, 2020.
[16] R. Verfurth, A Posteriori Error Estimation Techniques for Finite Element Methods, Numerical Mathematics and Scientific Computation, Oxford University Press, 2013.
[17] L. Bottou, F.E. Curtis, J. Nocedal, Optimization methods for large-scale machine learning, SIAM Rev. 60 (2018) 223–311.
[18] K. Pearson, On lines and planes of closest fit to systems of points in space, Philos. Mag. 2 (11) (1901) 559–572.
[19] D.P. Kingma, J. Ba, Adam: a method for stochastic optimization, arXiv preprint arXiv:1412.6980, 2014.
[20] J. He, L. Li, J. Xu, C. Zheng, Relu deep neural networks and linear finite elements, J. Comput. Math. 38 (3) (2020) 502–527.
[21] P. Morin, R.H. Nochetto, K.G. Siebert, Convergence of adaptive finite element methods, SIAM Rev. 44 (4) (2002) 631–658.
[22] Z. Cai, S. Zhang, Recovery-based error estimator for interface problems: conforming linear elements, SIAM J. Numer. Anal. 47 (2009) 2132–2156.
[23] D. Cai, Z. Cai, A hybrid a posteriori error estimator for conforming finite element approximations, Comput. Methods Appl. Mech. Eng. 339 (2018) 320–340.
[24] Z. Cai, J. Chen, M. Liu, Least-squares ReLU neural network (LSNN) method for linear advection-reaction equation, J. Comput. Phys. 443 (2021) 110514.
[25] Z. Cai, J. Chen, M. Liu, Self-adaptive deep neural network: numerical approximation to functions and PDEs, arXiv:2109.02839 [math.NA], J. Comput. Phys. 455 (2022) 111021.
[26] E. Allgower, K. Georg, Numerical Continuation Methods, Springer-Verlag, Berlin and Heidelberg, 1990.