Ritz Neural Network (RitzNN) Method for $H(\mathbf{curl})$ Problems

1st Zhiqiang Cai
Department of Mathematics
Purdue University
West Lafayette, USA
caiz@purdue.edu

2nd Min Liu
School of Mechanical Engineering
Purdue University
West Lafayette, USA
liu66@purdue.edu

3rd Dan Jiao
School of Electrical and Computer Engineering
Purdue University
West Lafayette, USA
djiao@purdue.edu

***Abstract*—In this paper, we propose to use deep neural networks for numerically solving $H(\mathbf{curl})$ problems through the Ritz loss functional formulation. A two-dimensional test problem shows that the resulting RitzNN method is capable of approximating $H(\mathbf{curl})$ problem with corner singularity accurately.**

***Index Terms*—Neural network, Ritz method, $H(\mathbf{curl})$ problem**

I. 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., [1]–[5]). 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. More importantly, a neural network function can automatically adapt to the target solution of a PDE.

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 [2], [3] and (2) the deep LS method [1], [2], [4], [5]. In this paper, we propose to use neural networks for numerically solving the $H(\mathbf{curl})$ problem through the Ritz method. To this end, let Ω be a bounded domain in $\mathbb{R}^d$ ($d = 2$ or 3) with boundary $\partial\Omega = \bar{\Gamma}_D \cup \bar{\Gamma}_N$ and $\Gamma_D \cap \Gamma_N = \emptyset$, and let n be the outward unit vector normal to the boundary. Denote by u the electric field, we consider the following $H(\mathbf{curl})$ problem,

$$\begin{cases} \nabla\times(\mu^{-1}\nabla\times u) + \beta\, u = f, & \text{in } \Omega, \\ u\times n\big|_{\Gamma_D} = g_D, & \text{and} \\ (\mu^{-1}\nabla\times u)\times n\big|_{\Gamma_N} = g_N, \end{cases} \tag{1}$$

where $\nabla\times$ is the curl operator; the f, g_D, and g_N are given vector fields defined on Ω, Γ_D, and Γ_N, respectively. This model problem originates from a stable marching scheme of the second-order hyperbolic partial differential equation on the electric field intensity u that is resulted from the Maxwell equations; the μ is the magnetic permeability; and the β depends on the electrical conductivity and the dielectric constant. Assume that μ^{-1} and β are bounded below by positive constants.

Denote by $L^2(\Omega)^d$ the space of the square integrable vector fields in $\mathbb{R}^d$ equipped with the standard L^2 norm: $\|v\| = \sqrt{(v,\, v)}$, where $(u,\, v) = (u,\, v)_\Omega = \int_\Omega u\cdot v\, dx$ denotes the standard L^2 inner product over domain Ω. Let

$$H(\mathbf{curl};\Omega) := \{v \in L^2(\Omega)^d : \nabla\times v \in L^2(\Omega)^{2d-3}\},$$

which is a Hilbert space equipped with the norm $\|v\|_{H(\mathbf{curl})} = \left(\|v\|^2 + \|\nabla\times v\|^2\right)^{1/2}$. The minimization formulation associated to problem (7) is to find $u \in H(\mathbf{curl};\Omega)$ such that

$$J(u) = \min_{v\in H(\mathbf{curl};\Omega)} J(v), \tag{2}$$

This work was supported in part by the National Science Foundation under grant DMS-2110571.

where the energy functional $J(v)$ is given by

$$\begin{aligned} J(v) =& \frac{1}{2}\left\{\left\|\mu^{-1/2}\nabla\times v\right\|^2 + \|v\|^2\right\} \\ &+ \frac{\gamma_D}{2}\left\|v\times\mathbf{n} - g_D\right\|^2_{1/2,\Gamma_D} \\ &- (f,\, v) - \int_{\Gamma_N} g_N\cdot v\, ds, \end{aligned} \tag{3}$$

where γ_D is a positive constant.

II. Ritz Neural Network Method

A neural network defines a function of the form

$$\begin{aligned} y &= N(\mathbf{x}) \\ &= \boldsymbol{\omega}^L\left(N^{(L-1)}\circ\cdots\circ N^{(2)}\circ N^{(1)}(\mathbf{x})\right) \\ &\quad - \mathbf{b}^L : x\in\mathbb{R}^d \longrightarrow y = N(\mathbf{x})\in\mathbb{R}^o, \end{aligned} \tag{4}$$

where d and $o = d$ are dimensions of input $\mathbf{x}$ and output y, respectively, $\boldsymbol{\omega}^{(L)}\in\mathbb{R}^{n_{L-1}\times o}$, $\mathbf{b}^{(L)}\in\mathbb{R}^o$, the symbol $\circ$ denotes the composition of functions, and L is the depth of the network. For $l = 1,\cdots,L-1$, the $N^{(l)}:\mathbb{R}^{n_{l-1}}\to\mathbb{R}^{n_l}$ is called the l^{th} hidden layer of the network defined by

$$\begin{aligned} N^{(l)}(\mathbf{x}^{(l-1)}) &= \sigma(\boldsymbol{\omega}^{(l)}\mathbf{x}^{(l-1)} - \mathbf{b}^{(l)}) \\ &\quad \text{for } \mathbf{x}^{(l-1)}\in\mathbb{R}^{n_{l-1}}, \end{aligned} \tag{5}$$

where $\boldsymbol{\omega}^{(l)}\in\mathbb{R}^{n_l\times n_{l-1}}$, $\mathbf{b}^{(l)}\in\mathbb{R}^{n_l}$, $\mathbf{x}^{(0)} = \mathbf{x}$, and $\sigma(t) = \max\{0,t\}^k$ with positive integer k is the activation function and its application to a vector is defined component-wise. This activation function is referred to as a spline activation ReLU^k. When $k=1$, $\sigma(t)$ is the popular rectified linear unit (ReLU). A typical L-layer (or, (L-1)-hidden layer) fully connected neural network structure is depicted in Fig. 1.

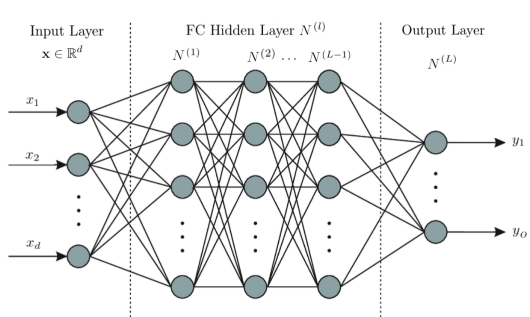


Fig. 1. An L-layer Neural Network.

Denote by $\mathcal{M}_n^o(\sigma)$ the collection of all functions of the form expressed in (4), where the $n = \sum_{l=1}^L n_l\times(n_{l-1}+1)$ is the total number of parameters [0]. Let $J_{\mathcal{T}}(v)$ be the discrete counterpart of the energy functional defined in (3), where integrations and differentiation in (3) are approximated by numerical integration and differentiation, respectively. Then the Ritz neural network (RitzNN) method for solving (7) is to minimize the discrete energy functional over the set $\mathcal{M}_n^1(\sigma)$, i.e., finding $u^n\in\mathcal{M}_n^o(\sigma)\subset H(\mathbf{curl};\Omega)$ such that

$$J_{\mathcal{T}}(u^n) = \min_{v\in\mathcal{M}_n^o(\sigma)} J_{\mathcal{T}}(v). \tag{6}$$

III. Numerical Experiments

This section presents numerical results for a two-dimensional $H(\mathbf{curl})$ problem with a corner singularity. The test problem is given by

$$\begin{cases} \nabla^\perp(\nabla\times u) + u = f, & \text{in } \Omega, \\ u\times n\big|_{\Gamma_D} = g_D, & \text{and } \nabla\times u\big|_{\Gamma_N} = g_N, \end{cases} \tag{7}$$

where $\nabla^\perp = (\partial_y, -\partial_x)^T$ is the formal adjoint of the scalar curl operator $\nabla\times = (-\partial_y, \partial_x)$, and $\Omega = \{(r,\theta)|\, r\in(0,1),\ \theta\in(0,3\pi/2)\}$. Let

$$\Gamma_N = \left\{(1,\theta)\big|\,\theta\in(0,3\pi/2)\right\}$$

and,

$$\begin{aligned} &\Gamma_D = \Gamma_2\cup\Gamma_3 \\ \equiv &\left\{(r,0)\big|\, r\in(0,1)\right\}\cup\left\{(r,3\pi/2)\big|\, r\in(0,1)\right\}. \end{aligned}$$

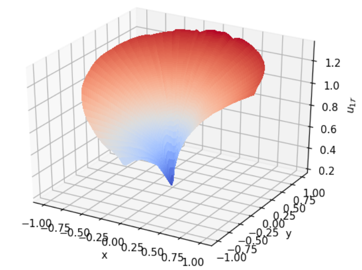
For $f(r,\theta) = \frac{4}{3}r^{\frac{1}{3}}\left(\sin(\frac{\pi+\theta}{3}), \cos(\frac{\pi+\theta}{3})\right)^T$, $g_N = 0$, and $g_D|_{\Gamma_2} = \frac{2\sqrt{3}}{3}r^{\frac{1}{3}}$ and $g_D|_{\Gamma_3} = -\frac{2\sqrt{3}}{3}r^{\frac{1}{3}}$, the exact solution is $u = f$.

Two network structures were utilized to approximate the result of this problem. The first is a NN with one hidden layer of 300 neurons, and the second is a two-hidden-layer NN with 32 neurons at each hidden layer (represented as 2-300-2 NN and 2-32-32-2 NN respectively). These two NNs have the similar number of parameters to learn. For each network structure, we tested the spline activation function with two different orders: a standard ReLU and a ReLU^2. The minimization problems in (1.3) formulated by the RitzNN are solved by the Adam optimizer [7], where Γ_D is picked as 100

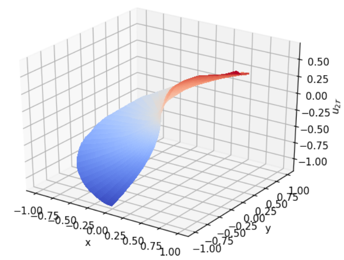
[0]In this paper, the first layer weights are normalized onto a unit hyper sphere, which leads to a smaller total number of parameters, $n = \sum_{l=1}^L n_l\times(n_{l-1}+1) - n_1$, see [6] for detail.

TABLE I
NUMERICAL RESULTS OF THE TEST PROBLEM *(3.1)*

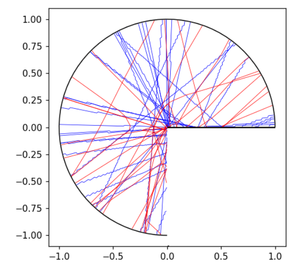
NN Structure	#Parameters	Activation	$\|u-u^n\|_{H(\mathbf{curl})}$	$\frac{\|u-u^n\|_{H(\mathbf{curl})}}{\|u\|_{H(\mathbf{curl})}}$
2-300-2	1202	ReLU	0.036322	0.020493
		ReLU^2	0.069180	0.039031
2-32-32-2	1186	ReLU	0.052693	0.029729
		ReLU^2	0.032767	0.018487



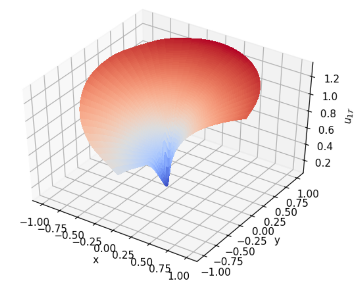
(a) u_x^n, ReLU activation



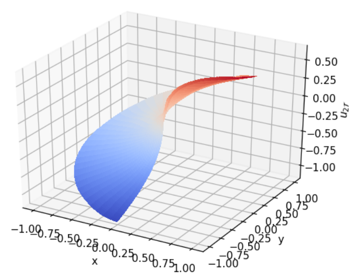
(b) u_y^n, ReLU activation



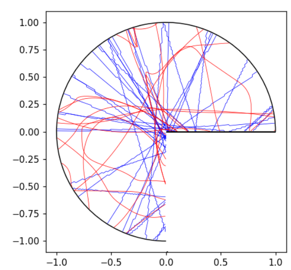
(c) Break lines generated by the RitzNN with ReLU activation



(d) u_x^n, ReLU^2 activation



(e) u_y^n, ReLU^2 activation



(f) Break curves generated by the RitzNN with ReLU^2 activation

Fig. 2. Numerical results of RitzNN method using a *3*-layer NN with two different activation functions, both using a *2-32-32-2* network structure.

empirically. The integrals of the energy functionals are computed numerically using composite mid-point quadrature rules with 100×270 quadrature points, which are uniformly distributed along radial and circumferential directions in a polar coordinate framework. During the network training, for each experiment, we conducted two run training with a learning rate of 0.01 first, and then 0.001 in the second run. The training stops at each run when the value of the energy functional decreases within 0.01% in the last 2000 iterations.

Our numerical results of the energy norm $\|u-u^n\|_{H(\mathbf{curl})}$ and the relative error measured by $\frac{\|u-u^n\|_{H(\mathbf{curl})}}{\|u\|_{H(\mathbf{curl})}}$ are reported in Table.I. It can be seen that both two-layer and three-layer NN structures can approximate the result reasonably accurate. With the similar number of parameters, both the 2-300-2 NN and the 2-32-32-2 NN produce similar numerical results, with the latter performs slightly better. In Fig. 2, the graphical approximation results of are plotted for the three-layer 2-32-32-2 NN structure. Comparing the two spline activation functions, ReLU NN generates a piecewise linear approximation while ReLU^2 generating a piecewise fourth-order polynomial functions. The ReLU NN produces good approximation with spurious zig-zag on the boundaries (see Fig.2(a) and Fig.2(b)) while the ReLU^2 3-layer NN approximates the true solution better with smooth boundary in this example (see Fig.2(d) and Fig.2(e)); and it reports the best accuracy in terms of the absolute or relative energy norm in the four setup, see the last row in

Table. I.

The corresponding physical partition generated by the hidden layer neurons are depicted in Fig.2(c) and 2(c), where one can find that the break curves are denser near the singular point at the origin, which explains the superior expressiveness of NN models: the self adaptivity as a free-knot spline which move their knots according to the corresponding target functions.

IV. Conclusion

A learning based RitzNN method for numerically solving $H(\mathbf{curl})$ problem is proposed and tested in this paper. RitzNN method uses a fully connected neural network with a spline activation function as the discretization model; solving a general $H(\mathbf{curl})$ problem is then formulated as a minimization of the corresponding energy functional $J(v)$ in (3.1). A two-dimensional numerical experiment shows that the RitzNN method is capable of approximating $H(\mathbf{curl})$ problem with corner singularity accurately.

References

[1] J. Berg and K. Nystrom, "A unified deep artificial neural network approach to partial differential equations in complex geometries," *Neurocomputing*, vol. 317, pp. 28–41, 2018.

[2] Z. Cai, J. Chen, M. Liu, and X. Liu, "Deep least-squares methods: An unsupervised learning-based numerical method for solving elliptic pdes," *Journal of Computational Physics*, vol. 420, p. 109707, 2020.

[3] W. E and B. Yu, "The deep ritz method: A deep learning-based numerical algorithm for solving variational problems," *Communications in Mathematics and Statistics*, vol. 6, no. 1, pp. 1–12, 3 2018.

[4] M. Raissia, P. Perdikarisb, and G. Karniadakisa, "Physics-informed neural networks: A deep learning framework for solving forward and inve," *Journal of Computational Physics*, vol. 378, p. 686–707, 2019.

[5] J. Sirignano and K. Spiliopoulos, "DGM: A deep learning algorithm for solving partial differential equations," *Journal of Computational Physics*, vol. 375, pp. 1139–1364, 2018.

[6] M. Liu, Z. Cai, and J. Chen, "Adaptive two-layer relu neural network: I. best least-squares approximation," *submitted and unpublished*, 2021.

[7] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.