RITZNET: A DEEP NEURAL NETWORK METHOD FOR LINEAR STRESS PROBLEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning based method for physics related computation has attracted significant attention recently. Effort has been devoted into learning a surrogate model which simulates system behavior from existing data. This paper presents RitzNet, an unsupervised learning method which takes any point in the computation domain as input, and learns a neural network model to output its corresponding function value satisfying the underlying governing PDEs. We focus on the linear elastic boundary value problem and formulate it as the natural minimization of its associated energy functional, whose discrete version is further utilized as the loss function of RitzNet. A standard fully connected deep neural network structure is explored in this study to model the solutions of a system of elliptic PDEs. Numerical studies on problems with analytical solutions or unknown solutions show that the proposed RitzNet is capable of approximating linear elasticity problems accurately. A parametric sensitivity study sheds light on the potential of RitzNet due to its meshless characteristics.

1 INTRODUCTION

Stress analysis is a fundamental problem of computational engineering and physics, as failures of most engineering components is usually due to stress. The subjects under stress investigation may vary from a submarine pressure vessel to the fuselage of a jumbo jet aircraft, or from the legs of an integrated circuit to the structure of a historical dam. The underlying governing equations for stress analysis problem are the constitutive equations which expresses a relation between the stress and strain tensor field, and the equilibrium equation under Newton's law, these yield a system of elliptic partial differential equations (PDEs). When the geometrical description of the structure or the loading status are complicated, analytical (closed-form) solutions can not be obtained and one must generally resort to numerical approaches such as the finite element, the finite difference, or the finite volume method to solve a general linear stress problem.

In the last decade, Deep Neural networks (DNNs) have achieved astonishing performance in computer vision, natural language processing, and many other machine learning (ML) related tasks. This success encourages their wide applications to many other fields, including recent studies of (i) using *supervised ML* algorithms to create a mid-fidelity surrogate model that learns the performances (e.g. stress distribution) from rich traditional simulations or experimental observations and predicts their distributions in real time (Liang et al., 2018; Gao et al., 2020; Wang et al., 2021; Vurtur Badarinath et al., 2021; Iakovlev et al., 2021; Li et al., 2021); (ii) *semi-supervised ML* algorithms to enforce the corresponding PDEs as a constraints or regularization term in the data-driven learning processes for solution acceleration or simulation improvement (Long et al., 2019; Raissi et al., 2019); and (iii) *unsupervised ML* approaches to directly approximate solutions of various types of known PDEs (Sirignano & Spiliopoulos, 2018; Berg & Nystrom, 2018; E & Yu, 2018; Cai et al., 2020; 2021).

The unsupervised ML methods rely on no training data from previous simulations or experiments and use only the underlying governing equations and/or boundary conditions as constraints to numerically approximate the solutions of PDEs; they offers an alternative method to the existing numerical schemes. Preliminary results have shown advantages of using DNNs for solving PDEs in both high dimensions (Sirignano & Spiliopoulos, 2018) and low dimensions (Cai et al., 2021) for computationally challenging problems. Although some initial investigation has been studied for solving theoretical PDEs and have shown its efficacy, no previous work, to our best knowledge, has

been done for the systems of elliptic PDEs that describe linear stress problems; this motivate us to explore the potential of using unsupervised learning method to simulate this ubiquitous engineering problem, and more important, since the learning based method does not require a mesh, it may benefit the design optimization problem in which the traditional mesh based method may encounter difficulties.

Neural network functions are nonlinear functions of the parameters, discretization of a PDE can be set up as an optimization problem through either the natural minimization or manufactured least-squares (LS) principles. Accordingly, existing methods consist of (1) the deep Ritz method (Cai et al., 2020; E & Yu, 2018) using natural minimization and (2) the deep LS method (Berg & Nystrom, 2018; Raissi et al., 2019; Sirignano & Spiliopoulos, 2018; Cai et al., 2020) using manufactured least-squares. In this paper, we propose RitzNet, an unsupervised ML method which solve linear elasticity problems using DNN functions as the approximation model, and natural minimization principle associated energy functional as the loss function. Section 2 reformulates the linear elasticity problem into a minimization problem using Ritz method. Section 3 presents the RitzNet method in details and we show our numerical studies in Section 4 and conclude the paper in Section 5.

2 RITZ FORMULATION OF LINEAR ELASTICITY

Let Ω be a bounded computational domain in $\mathbb{R}^d$ ($d = 2$ or 3) with boundary $\partial\Omega = \Gamma_D \cup \Gamma_N$ and $\Gamma_D \cap \Gamma_N = \emptyset$, and let $\boldsymbol{n}$ be the outward unit vector normal to the boundary. Denote by $\boldsymbol{u}$ and $\boldsymbol{\sigma}$ the displacement field and the stress tensor, respectively. Consider the following linear structure problem

$$\left\{\begin{array}{rcll} -\nabla\cdot\boldsymbol{\sigma} &=& \boldsymbol{f}, & \text{in } \Omega, \\ \boldsymbol{\sigma}(\boldsymbol{u}) &=& 2\mu\boldsymbol{\epsilon}(\boldsymbol{u}) + \lambda\nabla\cdot\boldsymbol{u}\,\delta_{d\times d} & \text{in } \Omega \end{array}\right. \tag{1}$$

with boundary conditions

$$\boldsymbol{u}\big|_{\Gamma_D} = \boldsymbol{g}_D \quad \text{and} \quad \big(\boldsymbol{\sigma}\boldsymbol{n}\big)\big|_{\Gamma_N} = \boldsymbol{g}_N,$$

where $\nabla\cdot$ is the divergence operator; $\boldsymbol{\epsilon}(\boldsymbol{u}) = \frac{1}{2}\big(\nabla\boldsymbol{u} + (\nabla\boldsymbol{u})^T\big)$ is the strain tensor; the $\boldsymbol{f}$, $\boldsymbol{g}_D$, and $\boldsymbol{g}_N$ are given vector-valued functions defined on Ω, Γ_D, and Γ_N, representing body force, boundary displacement and boundary traction force condition respectively; $\delta_{d\times d}$ is the the d-dimensional identity matrix; μ and λ are the material Lam*é* constants.

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

Since it is difficult for neural network functions to satisfy boundary conditions (see E & Yu (2018)), as in Cai et al. (2020), we enforce the Dirichlet (essential) boundary condition weakly through the energy functional. To this end, define the energy functional by

$$J(\boldsymbol{v}) = \frac{1}{2}\left\{\int_\Omega \boldsymbol{\sigma}(\boldsymbol{v}) : \boldsymbol{\varepsilon}(\boldsymbol{v})\,d\boldsymbol{x} + \|\boldsymbol{v} - \boldsymbol{g}_D\|^2_{1/2,\Gamma_D}\right\} - (\boldsymbol{f},\,\boldsymbol{v}) - \int_{\Gamma_N}\boldsymbol{g}_N\cdot\boldsymbol{v}\,ds \tag{2}$$

$$= \frac{1}{2}\left\{\int_\Omega \Big(2\mu\,|\boldsymbol{\varepsilon}(\boldsymbol{v})|^2 + \lambda\,|\nabla\cdot\boldsymbol{v}|^2\Big)\,d\boldsymbol{x} + \|\boldsymbol{v} - \boldsymbol{g}_D\|^2_{1/2,\Gamma_D}\right\} - (\boldsymbol{f},\,\boldsymbol{v}) - (\boldsymbol{g}_N,\boldsymbol{v})_{0,\Gamma_N}.$$

Then the Ritz formulation of problem (1) is to find $\boldsymbol{u} \in \boldsymbol{H}^1(\Omega)^d$ such that

$$J(\boldsymbol{u}) = \min_{\boldsymbol{v}\in\boldsymbol{H}^1(\Omega)^d} J(\boldsymbol{v}). \tag{3}$$

For any $\boldsymbol{u}, \boldsymbol{v} \in \boldsymbol{H}^1(\Omega)^d$, define the following bilinear form by

$$\begin{aligned} a(\boldsymbol{u},\boldsymbol{v}) &= \int_\Omega \boldsymbol{\sigma}(\boldsymbol{u}) : \boldsymbol{\varepsilon}(\boldsymbol{v})\,dx + (\boldsymbol{u},\boldsymbol{v})_{1/2,\Gamma_D} \\ &= 2\mu(\boldsymbol{\epsilon}(\boldsymbol{u}),\boldsymbol{\epsilon}(\boldsymbol{v})) + \lambda(\nabla\cdot\boldsymbol{u},\nabla\cdot\boldsymbol{v}) + (\boldsymbol{u},\boldsymbol{v})_{1/2,\Gamma_D} \end{aligned}$$

and the linear form by

$$f(\boldsymbol{v}) = (\boldsymbol{f}, \boldsymbol{v}) + (\boldsymbol{g}_N, \boldsymbol{v})_{0,\Gamma_N} + (\boldsymbol{g}_D, \boldsymbol{v})_{1/2,\Gamma_D}.$$

Problem (3) is equivalent to finding $\boldsymbol{u} \in \boldsymbol{H}^1(\Omega)^d$ such that

$$a(\boldsymbol{u}, \boldsymbol{v}) = f(\boldsymbol{v}), \quad \forall\ \boldsymbol{v} \in \boldsymbol{H}^1(\Omega)^d. \tag{4}$$

To establish the well-posedness of (3), we have the following modified Korn inequality.

Lemma 1. *For all $\boldsymbol{v} \in \boldsymbol{H}^1(\Omega)^d$, there exists a positive constant C such that*

$$\|\boldsymbol{v}\|_{1,\Omega} \leq C \left(\|\boldsymbol{\varepsilon}(\boldsymbol{v})\|_{0,\Omega} + \|\boldsymbol{v}\|_{1/2,\Gamma_D} \right). \tag{5}$$

See proof in Appendix A.1

Proposition 1. *Problem* (3) *has a unique solution $\boldsymbol{u} \in \boldsymbol{H}^1(\Omega)^d$. Moreover, the solution $\boldsymbol{u}$ satisfies the following a priori estimate:*

$$\|\boldsymbol{u}\|_{1,\Omega} \leq C \left(\|\boldsymbol{f}\|_{-1,\Omega} + \|\boldsymbol{g}_D\|_{1/2,\Gamma_D} + \|\boldsymbol{g}_N\|_{-1/2,\Gamma_N} \right). \tag{6}$$

See proof in Appendix A.2

3 RITZNET METHOD

In this section, we describe the RitzNet which includes a standard fully connected deep neural network as the model of function $\boldsymbol{u}(\boldsymbol{x})$, the discretized energy function for RiztNet loss and numerical integration and differentiation operators. The structure of the RitzNet is illustrated in Figure 1.

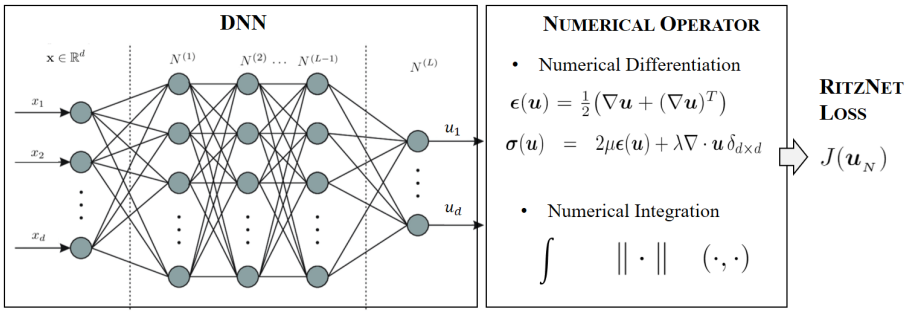


Figure 1: RitzNet architecture. A fully connected L-layer network is used to generate the map from an arbitrary point $\boldsymbol{x}$ in Ω to $\boldsymbol{u}(\boldsymbol{x})$, numerical operators are used to approximate the gradient, divergence and integral in the discrete energy function for the RitzNet loss.

3.1 DEEP NEURAL NETWORK STRUCTURE

For $j = 1, \cdots, l-1$, let $N^{(j)} : \mathbb{R}^{n_{j-1}} \to \mathbb{R}^{n_j}$ be the vector-valued ridge function of the form

$$N^{(j)}(\mathbf{x}^{(j-1)}) = \sigma(\boldsymbol{\omega}^{(j)}\mathbf{x}^{(j-1)} - \mathbf{b}^{(j)}) \quad \text{for } \mathbf{x}^{(j-1)} \in \mathbb{R}^{n_{j-1}}, \tag{7}$$

where $\boldsymbol{\omega}^{(j)} \in \mathbb{R}^{n_j \times n_{j-1}}$ and $\mathbf{b}^{(j)} \in \mathbb{R}^{n_j}$ are the respective weights and bias to be determined; $\mathbf{x}^{(0)} = \mathbf{x}$; and $\sigma(t) = \max\{0, t\}^p$ with positive integer p 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^p. When $p = 1$, $\sigma(t)$ is the popular rectified linear unit (ReLU). There are many other activation functions such as (logistic, Guassian, arctan) sigmoids (see, e.g., Pinkus (1999)).

Let $\boldsymbol{\omega}^{(l)} \in \mathbb{R}^{d \times n_{l-1}}$ and $\boldsymbol{b}^{(l)} \in \mathbb{R}^d$ be the output weights and bias, respectively. Then a l-layer neural network generates the following set of vector fields in $\mathbb{R}^d$

$$\mathcal{M}_N(\boldsymbol{\theta}, l) = \left\{ \boldsymbol{\omega}^l \left(N^{(l-1)} \circ \cdots \circ N^{(1)}(\mathbf{x}) \right) - \boldsymbol{b}^l : \boldsymbol{\omega}^{(j)} \in \mathbb{R}^{n_j \times n_{j-1}}, \mathbf{b}^{(j)} \in \mathbb{R}^{n_j} \text{ for all } j \right\}, \tag{8}$$

where the symbol $\circ$ denotes the composition of functions; $\boldsymbol{\theta}$ denote all parameters to be trained, i.e., the weights $\{\boldsymbol{\omega}^{(j)}\}_{j=1}^{l}$ and the bias $\{\mathbf{b}^{(j)}\}_{j=1}^{l}$. The total number of the parameters is given by

$$N = M_d(L) = \sum_{j=1}^{l} n_j \times (n_{j-1} + 1) \quad \text{with } n_l = d. \tag{9}$$

This class of functions is rich enough to accurately approximate any continuous function defined on a compact set $\Omega \in \mathbb{R}^d$ (see Cybenko (1989); Hornik et al. (1989) for the universal approximation property). However, this is not the main reason why NNs are so effective in practice. One way to understand its approximation power is from the point view of polynomial spline functions with free knots (Schumaker (1981)). The set $\mathcal{M}_N(\boldsymbol{\theta}, 2)$ may be regarded as a beautiful extension of free knot splines from one dimensional scalar-valued function to multi-dimensional vector-valued function. It has been shown that the approximation of functions by splines can generally be dramatically improved if the knots are free.

3.2 RitzNet Method

Note that neural network functions in $\mathcal{M}_N(\boldsymbol{\theta}, l)$ are nonlinear with respect to the parameters $\boldsymbol{\theta}$. This implies that (1) cannot be discretized by the conventional discretization approach based on the corresponding variational formulation (4). Instead, discretization using NNs must be based on a optimization formulation. In this paper, we employ the Ritz formulation (3) that minimizes the energy functional.

To approximate the solution of (1) using neural network functions, the RitzNet method is to minimize the energy functional over the set $\mathcal{M}_N(\boldsymbol{\theta}, l)$, i.e., finding $\boldsymbol{u}_N \in \mathcal{M}_N(\boldsymbol{\theta}, l) \subset \boldsymbol{H}^1(\Omega)^d$ such that

$$J(\boldsymbol{u}_N) = \min_{\boldsymbol{v} \in \mathcal{M}_N(\boldsymbol{\theta}, l)} J(\boldsymbol{v}). \tag{10}$$

Theorem 1. *Let $\boldsymbol{u} \in \boldsymbol{H}^1(\Omega)^d$ be the solution of problem* (4)*, and let $\boldsymbol{u}_N \in \mathcal{M}_N(\boldsymbol{\theta}, l)$ be a solution of* (10)*. Then we have*

$$\|\boldsymbol{u} - \boldsymbol{u}_N\|_a = \inf_{\boldsymbol{v} \in \mathcal{M}_N(\boldsymbol{\theta}, l)} \|\boldsymbol{u} - \boldsymbol{v}\|_a, \tag{11}$$

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

Proof. Let $\boldsymbol{u} \in \boldsymbol{H}^1(\Omega)^d$ be the solution of problem (4). For any $\boldsymbol{w} \in \boldsymbol{H}^1(\Omega)^d$, the definition of the energy functional and (4) imply that

$$\begin{aligned} 2\big(J(\boldsymbol{w}) - J(\boldsymbol{u})\big) &= a(\boldsymbol{w}, \boldsymbol{w}) - 2f(\boldsymbol{w}) - a(\boldsymbol{u}, \boldsymbol{u}) + 2f(\boldsymbol{u}) \\ &= a(\boldsymbol{w}, \boldsymbol{w}) - 2a(\boldsymbol{u}, \boldsymbol{w}) + a(\boldsymbol{u}, \boldsymbol{u}) = \|\boldsymbol{u} - \boldsymbol{w}\|_a^2. \end{aligned}$$

The above equality with $\boldsymbol{w} = \boldsymbol{u}_N \in \mathcal{M}_N(\boldsymbol{\theta}, l) \subset \boldsymbol{H}^1(\Omega)^d$ yields

$$\|\boldsymbol{u} - \boldsymbol{u}_N\|_a^2 = 2\left(J(\boldsymbol{u}_N) - J(\boldsymbol{u})\right) \leq 2\left(J(\boldsymbol{v}) - J(\boldsymbol{u})\right) = \|\boldsymbol{u} - \boldsymbol{v}\|_a^2$$

for any $\boldsymbol{v} \in \mathcal{M}_N(\boldsymbol{\theta}, l)$, and hence the validity of (11). This completes the proof of the theorem. □

Theorem 1 indicates that $\boldsymbol{u}_N$ is the best approximation with respect to the energy norm $\|\cdot\|_a$.

3.3 Effect of Numerical integration

Evaluation of the energy functional requires integration and differentiation. In practice, they are computed by numerical integration and differentiation.

For simplicity of presentation, we use the composite mid-point quadrature rule as in Cai et al. (2020). To this end, let us partition the domain Ω by a collection of subdomains

$$\mathcal{T} = \{K \,:\, K \text{ is an open subdomain of } \Omega\}$$

such that

$$\bar{\Omega} = \cup_{K \in \mathcal{T}} \bar{K} \quad \text{and} \quad K \cap T = \emptyset, \quad \forall\, K, T \in \mathcal{T}.$$

That is, the union of all subdomains of $\mathcal{T}$ equals to the whole domain Ω, and any two distinct subdomains of $\mathcal{T}$ have no intersection. The resulting partitions of the boundary $\Gamma_{_D}$ and $\Gamma_{_N}$ are

$$\mathcal{E}_{_D} = \{E = \partial K \cap \Gamma_{_D} : \ K \in \mathcal{T}\} \quad \text{and} \quad \mathcal{E}_{_N} = \{E = \partial K \cap \Gamma_{_N} : \ K \in \mathcal{T}\},$$

respectively.

Let $\boldsymbol{x}_{_T}$ and $\mathbf{x}_{_E}$ be the centroids of $T \in \mathcal{T}$ and $E \in \mathcal{E}_{_S}$ for $S = D$ and N, respectively. For any integrand $v(\boldsymbol{x})$, the composite "mid-point" quadrature rules over the domain Ω and the boundary $\Gamma_{_S}$ are given by

$$\int_\Omega v(\boldsymbol{x})\, d\mathbf{x} \approx \sum_{T\in\mathcal{T}} v(\boldsymbol{x}_{_T})\,|T| \quad \text{and} \quad \int_S v(\boldsymbol{x})\, ds \approx \sum_{E\in\mathcal{E}_{_S}} v(\boldsymbol{x}_{_E})\,|E|,$$

respectively, where $|T|$ and $|E|$ are the respective volume of element $T \in \mathcal{T}$ and area of boundary element $E \in \mathcal{E}_{_S}$. Similarly, one may use any quadrature rule such as composite trapezoidal, Simpson, Gaussian, etc. The $\boldsymbol{x}_{_T}$ for all $T \in \mathcal{T}$ will be used as quadrature points which are fundamentally different from sampling points used in the setting of standard supervised learning. At each $\boldsymbol{x}_{_T}$, the evaluations of $\boldsymbol{\varepsilon}\big(\boldsymbol{v}(\boldsymbol{x}_{_T})\big)$ and $\nabla\cdot\boldsymbol{v}(\boldsymbol{x}_{_T})$ are done through numerical differentiation with a small mesh size or automatic differentiation denoted by

$$\boldsymbol{\varepsilon}_h\big(\boldsymbol{v}(\boldsymbol{x}_{_T})\big) \quad \text{and} \quad \nabla_h\cdot\boldsymbol{v}(\boldsymbol{x}_{_T}).$$

Define the discrete counterpart of the energy function $J(\cdot)$ by

$$J_{_\mathcal{T}}(\boldsymbol{v}) = \frac{1}{2}\left\{\sum_{T\in\mathcal{T}}\Big(2\mu\,\big|\boldsymbol{\varepsilon}_h\big(\boldsymbol{v}(\boldsymbol{x}_{_T})\big)\big|^2 + \lambda\,|\nabla_h\cdot\boldsymbol{v}(\boldsymbol{x}_{_T})|^2\Big)\,d\boldsymbol{x} + \gamma_{_D}\sum_{E\in\mathcal{E}_{_D}}\big|\boldsymbol{v}-\boldsymbol{g}_{_D}\big|^2(\boldsymbol{x}_{_E})\right\}$$

$$-\sum_{T\in\mathcal{T}}\big(\boldsymbol{f}\cdot\boldsymbol{v}\big)(\boldsymbol{x}_{_T}) - \sum_{E\in\mathcal{E}_{_N}}\big(\boldsymbol{g}_{_N}\cdot\boldsymbol{v}\big)(\boldsymbol{x}_{_E}), \tag{12}$$

where $\gamma_{_D}$ is a relatively large positive constant. The RitzNet approximation to the solution of (1) is to seeking $\boldsymbol{u}_{_\mathcal{T}} \in \mathcal{M}_{_N}(\boldsymbol{\theta}, l)$ such that

$$J_{_\mathcal{T}}(\boldsymbol{u}_{_\mathcal{T}}) = \min_{\boldsymbol{v}\in\mathcal{M}_{_N}(\boldsymbol{\theta},l)} J_{_\mathcal{T}}(\boldsymbol{v}). \tag{13}$$

To understand the effect of numerical integration, we extend the first Strang lemma for the Galerkin approximation over a subspace (see, e.g, Ciarlet (1978)) to the Ritz approximation over a subset. To this end, define the discrete counterpart of the bilinear and linear forms by

$$a_{_\mathcal{T}}(\boldsymbol{u},\boldsymbol{v}) = 2\mu\sum_{T\in\mathcal{T}}\big(\boldsymbol{\varepsilon}_h(\boldsymbol{u}) : \boldsymbol{\varepsilon}_h(\boldsymbol{v})\big)(\boldsymbol{x}_{_T}) + \lambda\sum_{T\in\mathcal{T}}\big(\nabla_h\cdot\boldsymbol{u}\nabla_h\cdot\boldsymbol{v}\big)(\boldsymbol{x}_{_T}) + \gamma_{_D}\sum_{E\in\mathcal{E}_{_D}}\big(\boldsymbol{u}\cdot\boldsymbol{v}\big)(\boldsymbol{x}_{_E})$$

$$\text{and} \quad f_{_\mathcal{T}}(\boldsymbol{v}) = \sum_{T\in\mathcal{T}}\big(\boldsymbol{f}\cdot\boldsymbol{v}\big)(\boldsymbol{x}_{_T}) + \sum_{E\in\mathcal{E}_{_N}}\big(\boldsymbol{g}_{_N}\cdot\boldsymbol{v}\big)(\boldsymbol{x}_{_E}) + \gamma_{_D}\sum_{E\in\mathcal{E}_{_D}}\big(\boldsymbol{g}_{_D}\cdot\boldsymbol{v}\big)(\boldsymbol{x}_{_E}).$$

Theorem 2. *Assume that there exists a positive constant β independent of $\mathcal{M}_{_N}(\boldsymbol{\theta}, l)$ such that*

$$\beta\ \|\boldsymbol{v}\|_a^2 \le a_{_\mathcal{T}}(\boldsymbol{v},\boldsymbol{v}), \quad \forall\ \boldsymbol{v} \in \mathcal{M}_{_N}(\boldsymbol{\theta}, l). \tag{14}$$

Let $\mathbf{u}$ be the solution of (3) *and $\boldsymbol{u}_{_\mathcal{T}}$ a solution of* (13)*. Then there exists a positive constant C such that*

$$\|\boldsymbol{u}-\boldsymbol{u}_{_\mathcal{T}}\|_a \ \le\ C\inf_{\boldsymbol{v}\in\mathcal{M}_{_N}(\boldsymbol{\theta},l)}\left\{\|\boldsymbol{u}-\boldsymbol{v}\|_a + \sup_{\boldsymbol{w}_1,\boldsymbol{w}_2\in\mathcal{M}_{_N}(\boldsymbol{\theta},l)}\frac{|a(\boldsymbol{v},\boldsymbol{w}_1-\boldsymbol{w}_2) - a_{_\mathcal{T}}(\boldsymbol{v},\boldsymbol{w}_1-\boldsymbol{w}_2)|}{\|\boldsymbol{w}_1-\boldsymbol{w}_2\|_a}\right\}$$

$$+ C\sup_{\boldsymbol{w}_1,\,\boldsymbol{w}_2\in\mathcal{M}_{_N}(\boldsymbol{\theta},l)}\frac{|f(\boldsymbol{w}_1-\boldsymbol{w}_2) - f_{_\mathcal{T}}(\boldsymbol{w}_1-\boldsymbol{w}_2)|}{\|\boldsymbol{w}_1-\boldsymbol{w}_2\|_a}. \tag{15}$$

The proof of Theorem 2 is attached in Appendix A.3 and it indicates that the total error in the energy norm is bounded by the approximation error of the set of neural network functions plus the numerical integration error.

4 NUMERICAL STUDIES

In this section, we present our numerical results for two dimensional stress problems. The first test problem has a closed-form solution by which we test the accuracy of the RitzNet method; and the second is a real engineering benchmark problem with unknown analytic solution. We use the approximated solution obtained from a finite element methods with adaptive polynomial order as the baseline for comparison. We further explore the potential of using RitzNet as a design optimization method through the parametric study of a changing parameter.

In the experiments, the network structure is expressed as 2-n_1-$n_2\cdots n_{l-1}$-2 for a l-layer network with n_1, n_2 and n_{l-1} neurons in the respective first, second, and $(l-1)$th layers, and the first and last numbers represent the network input and output dimensions. The minimization of the RitzNet loss function (10) is numerically solved using the Adam version of gradient descent (Kingma & Ba, 2015) with a varying learning rate.

4.1 A TOY PROBLEM WITH CLOSED-FORM SOLUTION

Consider problem (1) defined on $\Omega = (-1,1)\times(-1,1)$ with the body force

$$\boldsymbol{f} = 2\mu\big(3-x^2-2y^2-2xy, 3-2x^2-y^2-2xy\big)^T + 2\lambda\big(1-y^2-2xy, 1-x^2-2xy\big)^T,$$

and the traction

$$\boldsymbol{g}_{_N} = 2(y^2-1)\big(2\mu+\lambda, \mu\big)^T$$

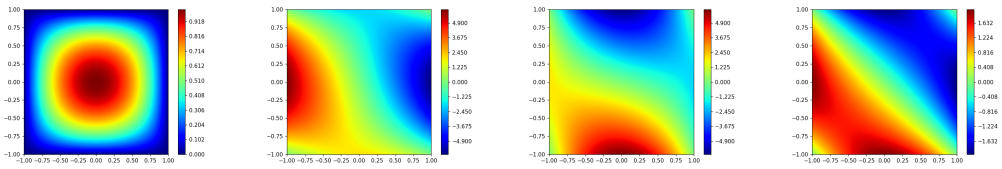
on $\Gamma_{_N} = \{(1,y) : y\in(-1,1)\}$, with the clamped boundary condition on $\Gamma_{_D} = \partial\Omega\setminus\Gamma_{_N}$. The exact solution of the test problem has of the form

$$\boldsymbol{u}(x,y) = (1-x^2)(1-y^2)\big(1,1\big)^T.$$

This function $\boldsymbol{u}(x,y)$ and the corresponding stress tensor $\boldsymbol{\sigma} = [[\sigma_{xx},\tau_{xy}],[\tau_{yx}\,\sigma_{yy}]]^T$ are depicted in Figure. 2.

A three layer RitzNet of structures 2-32-32-2 is tested, and a $ReLU^2$ activation function is selected to obtained a better smoothness of the approximated displacement field $\boldsymbol{u}$. For numerical integration, we use an uniformly distributed quadrature points of size 200×200 to approximate all integrals in the loss function. And all the numerical differentiation is approximated by the forward finite difference quotient with step size 0.001.

Table. 1 list the numerical results of using RitzNet to solve this problem under varying material properties. With a small network of total 1186 parameters, RiztNet can approximate this problem accurately. The graphical results are depicted in Figure. 3 for material property $\mu = 1$, and $\lambda = 1$, which conform to the exact solution listed in Figure.2 accurately.



(a) $u_x = u_y$ (b) σ_{xx} (c) σ_{yy} (d) Stress $\tau_{xy} = \tau_{yx}$

Figure 2: Test Problem 4.1 ($\mu = 1$, and $\lambda = 1$), exact solution of displacement and stress distributions.

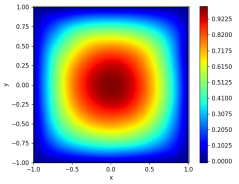
4.2 A TWO-DIMENSIONAL QUADRATIC MEMBRANE UNDER TENSION

The second problem is given by a quadratic membrane of elastic isotropic material with a circular hole in the center. Traction forces act on the upper and lower edges of the strip, body forces are ignored. Because of the symmetry of the problem, it suffices to compute only a fourth of the total geometry. The computational domain is then given by

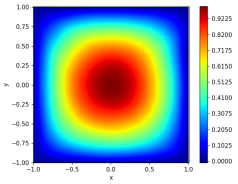
$$\Omega = \{\boldsymbol{x}\in\mathbb{R}^2 : 0<x_1<10, 0<x_2<10, x_1^2+x_2^2>1\}.$$

Table 1: Numerical results of RitzNet method for test problem 4.1 with several choices of material constant λ, and comparison with the exact solutions.

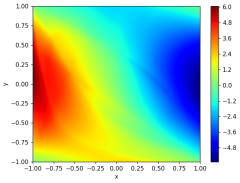
	Exact		**RitzNet (2-32-32-2)**		
	max $\|\boldsymbol{u}\|$	max $\boldsymbol{\sigma}_{11}$	max $\|\boldsymbol{u}_N\|$	max $\boldsymbol{\sigma}_{\mathbf{N}11}$	$\frac{\|\boldsymbol{u}-\boldsymbol{u}_N\|_a}{\|\boldsymbol{u}\|_a}$
$\lambda = 1$	1.4142	6	1.4147	6.19417	0.11280
$\lambda = 10$	1.4142	24	1.4415	24.2099	0.08994
$\lambda = 100$	1.4142	204	1.4264	226.039	0.07342



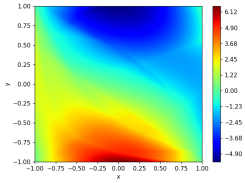
(a) Displacement $\bar{u_x}$



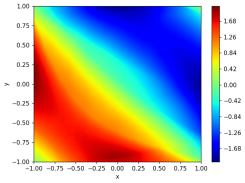
(b) Displacement $\bar{u_y}$



(c) Stress $\bar{\sigma}_{xx}$



(d) Stress $\bar{\sigma}_{xy}$



(e) Stress $\bar{\tau}_{xy}$

Figure 3: Test Problem 4.1 ($\mu = 1$, and $\lambda = 1$), approximated solution using RitzNet (2-32-32-2, activate function: ReLU^2, γ_D =1e+3, total number of iterations: 30000, learning rate: starts with 2e-2, decays 50% every 5000 iterations

The boundary condition on the top edge of the computation domain ($\Gamma_1 : \{x_2 = 10, 0 < x_1 < 10\}$) are set to $\boldsymbol{\sigma n} = (0, 4.5)^T$, the boundary condition on the bottom ($\Gamma_2 : \{x_2 = 0, 1 < x_1 < 10\}$) are set to $(\sigma_{11}, \sigma_{12}) \cdot \boldsymbol{n} = 0, u_2 = 0$ (symmetry condition), and finally, the boundary condition on the left ($\Gamma_3 : \{x_1 = 0, 1 < x_2 < 10\}$) are given by $(\sigma_{21}, \sigma_{22}) \cdot \boldsymbol{n} = 0$, and $u_1 = 0$ (symmetry condition). The material parameters are $E = 206900$ for Young's modulus and $\nu = 0.29$ for Poisson's ratio, and their relation with the Lam$\acute{e}$ constants is given by

$$\mu = \frac{E}{2(1+\nu)} \quad and \quad \lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}.$$

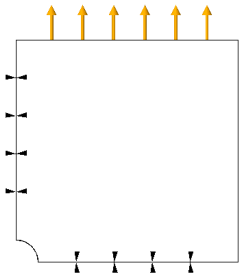
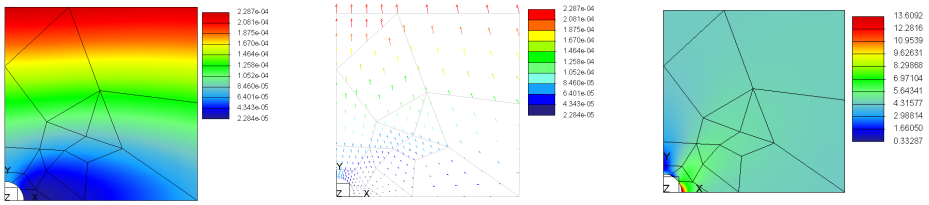


Figure 4: Computational domain and boundary conditions.

This is a benchmark problem taken from (Cai & Starke, 2004) as a real plane stress problem with stress concentration. Since there is no exact solution available, to evaluate the accuracy of the proposed RitzNet method, we use the adaptive finite element analysis (FEA) method to compute a benchmark solution. The discretization of the computation domain is through high-order p-element given in Figure. 5, and the adaptive process starts at cubic elements and stops at the highest edge polynomial order of 5. The resulting reference numerical solution is given in Figure. 5. The stress concentration is located at point (0,1) where the stress has a sharp transition locally due to the presence of the small hole.

For RitzNet, we first tested a three-layer RitzNet equipped with a sigmoid activation function (ReLU^2 activation function provide slightly inferior results). The quadrature points are set at the mid-points of uniformly distributed partition of the domain with size 0.04; and the quadrature points in the bottom left quarter are refined with a smaller size 0.01 to capture the geometric curvature

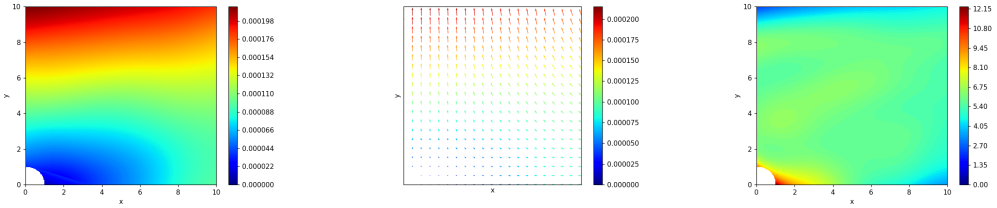


(a) Displacement magnitude $\|\boldsymbol{u}\|$

(b) Displacement field $\boldsymbol{u}$

(c) von Mises stress $\boldsymbol{\sigma}_{vm}$ distribution

Figure 5: Test Problem 4.2: Benchmark solution using FEA adaptive p-element with maximum element order of 5, convergence at 1% w.r.t the local stress and strain energy.



(a) Displacement magnitude $\|\boldsymbol{u}_N\|$

(b) Displacement field $\boldsymbol{u}_N$

(c) von Mises stress $\boldsymbol{\sigma}_{\mathbf{N}vm}$ distribution

Figure 6: Test Problem 4.2: Numerical solution using RitzNet (NN structure: 2-32-32-32-2, activate function: sigmoid, γ_D =1e+7, total number of iterations: 80000, learning rate: starts with 1e-1, decays 80% every 20000 iterations.)

near the hole. We experimented three network sizes (see Table. 2) and all the resulting displacement field solutions are close to the benchmark solution. For the stress concentration point and the corresponding maximum von Mises stress $\boldsymbol{\sigma}_{vm}$, we found larger network size performs better w.r.t the approximation of this near singular point's stress. A four layer network with 32 neurons in each hidden layer can archive closer maximum $\boldsymbol{\sigma}_{vm}$, see the last row in Table 2. One problem we encountered is that beyond this 2-32-32-32-2 network structure, we are not able to further increase the estimated value of the maximum $\boldsymbol{\sigma}_{vm}$ comparing to the benchmark solution. One explanation is that when the network gets larger, it also introduces more and more local minimums for this non-convex nonlinear optimization problem, and thus increases the training difficulties. Another factor that might play a role here is the numerical quadrature. An adaptive quadrature point selection might help for getting a more precise approximation of the energy functional for the numerical integration. Figure. 6 depicts the displacement magnitude, displacement vector field and von Mises stress distribution using RitzNet 2-32-32-32-2. It verifies that the proposed RitzNet is capable of approximating the independent variable $\boldsymbol{u}$ of this challenging problem, although there is still improvement space exist for problems with stress concentration/singularity.

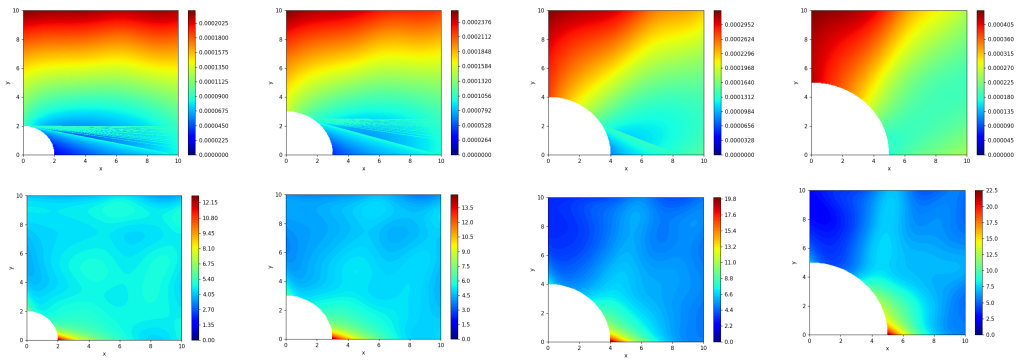
Table 2: A comparison of FEA and RitzNet results for test problem in 4.2

Method	**max** $\|\boldsymbol{u}\|$	**max** $\boldsymbol{u}_1$	**max** $\boldsymbol{u}_2$	**max** $\boldsymbol{\sigma}_{vm}$
FEA (Adaptive p-element)	2.287158e-04	-6.985006e-05	2.287158e-04	13.60922
RitzNet (2-16-16-2)	1.939397e-04	-6.962712e-05	1.939397e-04	8.200386
RitzNet (2-32-32-2)	2.110047e-04	-8.007632e-05	2.110047e-04	9.017071
RitzNet (2-42-42-2)	2.081562e-04	-8.569460e-05	2.081561e-04	9.686732
RitzNet (2-32-32-32-2)	2.274459e-04	-8.232628e-05	2.120236e-04	12.19299

4.3 PARAMETRIC DESIGN STUDY

In the last experiment, we explore the potential of using RitzNet for solving parametric PDEs. A common scenario in engineer design is to make decision in a space of design parameters referencing the simulation results for a series of choices. For example, engineers ask what is the largest hole size allowed to use such that the maximum stress is still within the material's limit.

To this end, we conducted a preliminary sensitivity study by varying the hole size from R1 to R5, taking a step of 0.5. Using the trained model of RitzNet 2-32-32-2 in the previous experiment as a starting point, we step through the various hole size by continuously training a same RitzNet with small number of iterations for each step. In particular, the initial model in the previous experiment took 80000 iterations, while in this sensitivity study, for each step, it continues from the previous step RitzNet model, and takes only 5000 iterations to converge to the results depicted in Figure. 7. Changing hole size results in a varying computational domains. For RitzNet, this only affects the set of quadrature points which can be easily added or removed. While in the traditional mesh-based methods, changing of domain might result in mesh conformity issues, thus introduces extra difficulties. Our results at each step conform to the numerical solutions evaluated in FEA method. Figure. 7 lists the displacement and the associate stress simulation results obtained through this continued RitzNet method. Comparing to the adaptive FEA, for instance, when the hole radius r= 4, our results show maximum von Mises stress of 19.825 while using adaptive p-element FEA solver, we obtain a results of 21.519.



(a) $r = 2$, max $\|\boldsymbol{u}\|$ =2.2160e-4, max $\boldsymbol{\sigma}_{vm}$ =12.6202

(b) $r = 3$, max $\|\boldsymbol{u}\|$ =2.2676e-4, max $\boldsymbol{\sigma}_{vm}$ =14.7910

(c) $r = 4$, max $\|\boldsymbol{u}\|$ =3.2667e-4, max $\boldsymbol{\sigma}_{vm}$ =19.8250

(d) $r = 5$, max $\|\boldsymbol{u}\|$ =4.4880e-4, max $\boldsymbol{\sigma}_{vm}$ =23.5357

Figure 7: Parametric study of the problem in 4.2 with varying hole radii, using RitzNet with continuous model.

5 CONCLUSION

Learning to solve PDEs is still in an early research stage; and this works provides a preliminary investigation of using DNN functions and physical system's natural minimization principle to numerically solve linear elasticity problems. Theoretical studies ensure the RitzNet method is mathematically valid while numerical studies on some two dimensional test problems show initially, the efficacy of the proposed method. However, there are still many open issues awaiting further research. For examples, how to select an appropriate network structure, what is an efficient way to handle singularities or stress concentrations, and how to set up a good initial of the network model and what is a best stopping criteria for the network training.

Perhaps the most exciting future application of the RitzNet is in shape optimization or even topology optimization since RitzNet involves no explicit mesh based discretization. Our initial sensitivity studies on a single parameter show promises which will be further explored in depth in our future work.

REFERENCES

Jens Berg and Kaj Nystrom. A unified deep artificial neural network approach to partial differential equations in complex geometries. *Neurocomputing*, 317:28–41, 2018.

Zhiqiang Cai and Gerhard Starke. Least-squares methods for linear elasticity. *SIAM Journal on Numerical Analysis*, 42(2):826–842, 2004.

Zhiqiang Cai, Jingshuang Chen, Min Liu, and Xinyu Liu. Deep least-squares methods: An unsupervised learning-based numerical method for solving elliptic pdes. *Journal of Computational Physics*, 420:109707, 2020.

Zhiqiang Cai, Jingshuang Chen, and Min Liu. Least-squares ReLU neural network (LSNN) method for linear advection-reaction equation. *Journal of Computational Physics*, 443:110514, 2021.

Philippe G. Ciarlet. *The finite element method for elliptic problems*. Society for Industrial and Applied Mathematics, 1978.

George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, 2:303–314, 1989.

Weinan E and Bing Yu. The deep ritz method: A deep learning-based numerical algorithm for solving variational problems. *Communications in Mathematics and Statistics*, 6(1):1–12, 3 2018.

Wenli Gao, Xinming Lu, Yanjun Peng, and Liang Wu. A deep learning approach replacing the finite difference method for in situ stress prediction. *IEEE Access*, 8:44063–44074, 2020. doi: 10.1109/ACCESS.2020.2977880.

Kur Hornik, Maxwell Stinchcombe, and Halber White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366, 1989.

Valerii Iakovlev, Markus Heinonen, and Harri Lähdesmäki. Learning continuous-time {pde}s from sparse data with graph neural networks. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=aUX5Plaq7Oy.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Representation Learning, San Diego*, 2015.

Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=c8P9NQVtmnO.

Liang Liang, Minliang Liu, Caitlin Martin, and Wei Sun. A deep learning approach to estimate stress distribution: a fast and accurate surrogate of finite-element analysis. *Journal of The Royal Society Interface*, 15(138):20170844, 2018.

Zichao Long, Yiping Lu, and Bin Dong. Pde-net 2.0: Learning pdes from data with a numeric-symbolic hybrid deep network. *J. Comput. Phys.*, 399, 2019.

Allan Pinkus. Approximation theory of the mlp model in nueral networks. *Acta Numerica*, 15: 143–195, 1999.

Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

Larry Schumaker. *Spline Functions: Basic Theory*. John Wiley, New York, 1981.

Justin Sirignano and Konstantinos Spiliopoulos. DGM: A deep learning algorithm for solving partial differential equations. *Journal of Computational Physics*, 375:1139–1364, 2018.

Poojitha Vurtur Badarinath, Maria Chierichetti, and Fatemeh Davoudi Kakhki. A machine learning approach as a surrogate for a finite element analysis: Status of research and application to one dimensional systems. *Sensors*, 21(5), 2021.

Yinan Wang, Diane Oyen, Weihong Guo, Anishi Mehta, Cory Braker Scott, Nishant Panda, M. Giselle Fernández-Godino, Gowri Srinivasan, and Xiaowei Yue. Stressnet - deep learning to predict stress with fracture propagation in brittle materials. *npj Materials Degradation*, 5(1), 2021.

A APPENDIX

A.1 PROOF OF LEMMA 1

Proof. For simplicity, we prove the validity of (5) in $\mathbb{R}^2$. To this end, denote the space of infinitesimal rigid motions in $\mathbb{R}^2$ by

$$RM = \{\boldsymbol{v} = (a,b)^T + c(y,x)^T \,|\, a,b,c \in \mathbb{R}\}.$$

For any $\boldsymbol{v} \in \boldsymbol{H}^1(\Omega)$, there exists a unique pair $(\boldsymbol{z},\boldsymbol{w}) \in \hat{\boldsymbol{H}}^1(\Omega) \times RM$ such that

$$\boldsymbol{v} = \boldsymbol{z} + \boldsymbol{w},$$

where $\hat{\boldsymbol{H}}^1(\Omega) = \left\{\boldsymbol{v} \in \boldsymbol{H}^1(\Omega) \middle| \int_\Omega \boldsymbol{v} = \boldsymbol{0},\ \int_\Omega \nabla \times \boldsymbol{v} = \boldsymbol{0}\right\}$. By the second Korn inequality and the fact that $\boldsymbol{\varepsilon}(\boldsymbol{w}) = \boldsymbol{0}$ for any $\boldsymbol{w} \in RM$, we have

$$\|\boldsymbol{z}\|_{1,\Omega} \leq C\ \|\boldsymbol{\varepsilon}(\boldsymbol{z})\|_{1,\Omega} = C\ \|\boldsymbol{\varepsilon}(\boldsymbol{v})\|_{1,\Omega}\,. \tag{16}$$

By the fact that RM is a finite dimensional space and the triangle and trace inequalities, we have

$$\|\boldsymbol{w}\|_{1,\Omega} \leq C\ \|\boldsymbol{w}\|_{1/2,\partial\Omega} \leq C\left(\|\boldsymbol{v}\|_{1/2,\partial\Omega} + \|\boldsymbol{z}\|_{1/2,\partial\Omega}\right) \leq C\left(\|\boldsymbol{v}\|_{1/2,\partial\Omega} + \|\boldsymbol{z}\|_{1,\Omega}\right).$$

Now, (5) is a direct consequence of the triangle inequality and the above two inequalities. □

A.2 PROOF OF PROPOSITION 1

Proof. By the Korn inequality in (5), it is easy to show that the bilinear form $a(\cdot,\cdot)$ is coercive in $\boldsymbol{H}^1(\Omega)^d \times \boldsymbol{H}^1(\Omega)^d$; i.e., for all $\boldsymbol{v} \in \boldsymbol{H}^1(\Omega)^d$, there exists a positive constant $\alpha > 0$ such that

$$\alpha\,\|\boldsymbol{v}\|_{1,\Omega}^2 \leq a(\boldsymbol{v},\boldsymbol{v}) = 2\mu\|\boldsymbol{\varepsilon}(\boldsymbol{v})\|_{0,\Omega}^2 + \lambda\|\nabla\cdot\boldsymbol{v}\|_{0,\Omega}^2 + \|\boldsymbol{v}\|_{1/2,\Gamma_D}^2. \tag{17}$$

It follows from the Cauchy-Schwarz and the trace inequalities that the bilinear form $a(\cdot,\cdot)$ and the linear form $f(\cdot)$ are continuous in $\boldsymbol{H}^1(\Omega)^d \times \boldsymbol{H}^1(\Omega)^d$ and $\boldsymbol{H}^1(\Omega)^d$, i.e., there exist positive constants M and C such that

$$|a(\boldsymbol{u},\boldsymbol{v})| \leq M\,\|\boldsymbol{u}\|_{1,\Omega}\,\|\boldsymbol{v}\|_{1,\Omega} \tag{18}$$

and that

$$|f(\boldsymbol{v})| \leq C\ \left(\|f\|_{-1,\Omega} + \|g_D\|_{1/2,\Gamma_D} + \|g_N\|_{-1/2,\Gamma_N}\right)\ \|\boldsymbol{v}\|_{1,\Omega}. \tag{19}$$

Now, the Lax-Milgram lemma implies that problem (3) has one and only one solution in $\boldsymbol{H}^1(\Omega)^d$. The *a priori* estimate in (6) is a direct consequence of (4) with $\boldsymbol{v} = \boldsymbol{u}$, (17), and (19). This completes the proof of the proposition. □

A.3 PROOF OF THEOREM 2

Proof. For any $\boldsymbol{v} \in \mathcal{M}_{_N}(\boldsymbol{\theta}, l)$, it is easy to see that $\boldsymbol{u}_{_\mathcal{T}} - \boldsymbol{v} \in \boldsymbol{H}^1(\Omega)^d$. By the assumption in (14), the definition of $J_{_\mathcal{T}}(\cdot)$, and the relations:

$$J_{_\mathcal{T}}(\boldsymbol{u}_{_\mathcal{T}}) \leq J_{_\mathcal{T}}(\boldsymbol{v}) \quad \text{and} \quad a(\boldsymbol{u}, \boldsymbol{u}_{_\mathcal{T}} - \boldsymbol{v}) = f(\boldsymbol{u}_{_\mathcal{T}} - \boldsymbol{v}),$$

we have

$$\begin{aligned}
\frac{\beta}{2}\,\|\boldsymbol{u}_{_\mathcal{T}} - \boldsymbol{v}\|_a^2 \ &\leq\ \frac{1}{2}a_{_\mathcal{T}}(\boldsymbol{u}_{_\mathcal{T}} - \boldsymbol{v}, \boldsymbol{u}_{_\mathcal{T}} - \boldsymbol{v}) = J_{_\mathcal{T}}(\boldsymbol{u}_{_\mathcal{T}}) - J_{_\mathcal{T}}(\boldsymbol{v}) + f_{_\mathcal{T}}(\boldsymbol{u}_{_\mathcal{T}} - \boldsymbol{v}) - a_{_\mathcal{T}}(\boldsymbol{v}, \boldsymbol{u}_{_\mathcal{T}} - \boldsymbol{v}) \\
&\leq\ f_{_\mathcal{T}}(\boldsymbol{u}_{_\mathcal{T}} - \boldsymbol{v}) - a_{_\mathcal{T}}(\boldsymbol{v}, \boldsymbol{u}_{_\mathcal{T}} - \boldsymbol{v}) \\
&=\ \Big(f_{_\mathcal{T}}(\boldsymbol{u}_{_\mathcal{T}} - \boldsymbol{v}) - f(\boldsymbol{u}_{_\mathcal{T}} - \boldsymbol{v})\Big) + \Big(a(\boldsymbol{v}, \boldsymbol{u}_{_\mathcal{T}} - \boldsymbol{v}) - a_{_\mathcal{T}}(\boldsymbol{v}, \boldsymbol{u}_{_\mathcal{T}} - \boldsymbol{v})\Big) \\
&\quad\ + a(\boldsymbol{u} - \boldsymbol{v}, \boldsymbol{u}_{_\mathcal{T}} - \boldsymbol{v})
\end{aligned}$$

which, together with the Cauchy-Schwarz inequality, implies

$$\|\boldsymbol{u}_{\mathcal{T}} - \boldsymbol{v}\|_a^2 \leq C\left(\|\boldsymbol{u} - \boldsymbol{v}\|_a^2 + \sup_{\boldsymbol{w}_1, \boldsymbol{w}_2 \in \mathcal{M}_N(\boldsymbol{\theta}, l)} \frac{|a(\boldsymbol{v}, \boldsymbol{w}_1 - \boldsymbol{w}_2) - a_{\mathcal{T}}(\boldsymbol{v}, \boldsymbol{w}_1 - \boldsymbol{w}_2)|}{\|\boldsymbol{w}_1 - \boldsymbol{w}_2\|_a}\right)$$

$$+ C \sup_{\boldsymbol{w}_1, \boldsymbol{w}_2 \in \mathcal{M}_N(\boldsymbol{\theta}, l)} \frac{|f(\boldsymbol{w}_1 - \boldsymbol{w}_2) - f_{\mathcal{T}}(\boldsymbol{w}_1 - \boldsymbol{w}_2)|}{\|\boldsymbol{w}_1 - \boldsymbol{w}_2\|_a}.$$

Combining the above inequality with the triangle inequality

$$\|\boldsymbol{u} - \boldsymbol{u}_{\mathcal{T}}\|_a \leq \|\boldsymbol{u} - \boldsymbol{v}\|_a + \|\boldsymbol{v} - \boldsymbol{u}_{\mathcal{T}}\|_a$$

and taking the infimum over all $\boldsymbol{v} \in \mathcal{M}_N(\boldsymbol{\theta}, l)$ yield (15). This completes the proof of the theorem. □