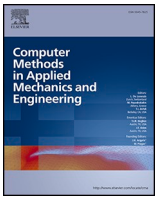
Computer Methods in Applied Mechanics and Engineering 437 (2025) 117755

Contents lists available at ScienceDirect

Computer Methods in Applied Mechanics and Engineering

journal homepage: www.elsevier.com/locate/cma



Differentiable finite element method with Galerkin discretization for fast and accurate inverse analysis of multidimensional heterogeneous engineering structures

Xi Wang [a], Zhen-Yu Yin [a,*], Wei Wu [b,c,d], He-Hua Zhu [b,c,d]

[a] Department of Civil and Environmental Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong, PR China
[b] College of Civil Engineering, Tongji University, Shanghai 200092, PR China
[c] State Key Laboratory of Disaster Reduction in Civil Engineering, Tongji University, Shanghai 200092, PR China
[d] Key laboratory of Geotechnical and Underground Engineering of Ministry of Education, Tongji University, Shanghai 200092, PR China

HIGHLIGHTS

- Galerkin discretization is utilized to build a novel differentiable finite element method (DFEM) that encodes physics and significantly reduces training cost.
- DFEM embeds the weak-form physics, boundary/initial conditions, and data constraints into the network architecture.
- Both the accuracy and efficiency of inverse analysis are improved by several orders of magnitude.
- Inverse analysis of three-dimensional heterogeneous engineering structures can be accomplished in seconds.
- DFEM can be readily extended as Physics-Encoded Numerical Network (PENN) to revitalize classical numerical methods for AI4Science.

ARTICLE INFO

Keywords:
Physics-Informed Neural Network (PINN)
Differentiable Finite Element Method (DFEM)
Physics-Encoded Numerical Network (PENN)
Inverse analysis
Heterogeneous engineering structures

ABSTRACT

Physics-informed neural networks (PINNs) are well-regarded for their capabilities in inverse analysis. However, efficient convergence is hard to achieve due to the necessity of simultaneously handling physics constraints, data constraints, blackbox weights, and blackbox biases. Consequently, PINNs are highly challenged in the inverse analysis of unknown boundary loadings and heterogeneous material parameters, particularly for three-dimensional engineering structures. To address these limitations, this study develops a novel differentiable finite element method (DFEM) based on Galerkin discretization for diverse inverse analysis. The proposed DFEM directly embeds the weak form of the partial differential equation into a discretized and differentiable computational graph, yielding a loss function from fully interpretable trainable parameters. Moreover, the labeled data, including boundary conditions, are strictly encoded into the computational graph without additional training. Finally, two benchmarks validate the DFEM's superior efficiency and accuracy: (1) With only 0.3 % training iterations, the DFEM can achieve an accuracy three orders of magnitude higher for the inverse analysis of unknown loadings. (2) With a training time five orders of magnitude faster, the DFEM is validated to be five orders of magnitude more accurate in determining unknown material parameters. Furthermore, two cases validate DFEM as effective for three-dimensional engineering structures: (1) A damaged cantilever beam characterized by twenty heterogeneous materials with forty unknown parameters is

* Corresponding author.
E-mail addresses: xiwang.wang@polyu.edu.hk (X. Wang), zhenyu.yin@polyu.edu.hk (Z.-Y. Yin), weiwu@tongji.edu.cn (W. Wu), zhuhehua@tongji.edu.cn (H.-H. Zhu).

https://doi.org/10.1016/j.cma.2025.117755
Received 7 January 2025; Received in revised form 13 January 2025; Accepted 13 January 2025
Available online 22 January 2025

efficiently solved. (2) A tunnel lining ring with sparse noisy data under unknown heterogeneous boundary loadings is successfully analyzed. These problems are solved in seconds, corroborating DFEM's potential for engineering applications. Additionally, the DFEM framework can be generalized to a Physics-Encoded Numerical Network (PENN) for further development and exploration.

1. Introduction

Solving governing partial differential equations (PDEs) is the essential way to predict and optimize system behaviors. Since analytical solutions are rarely available for most PDEs, numerical methods have garnered significant interest and achieved substantial success. For example, finite element method (FEM) [1,2], finite difference method (FDM), peridynamics (PD) [3,4], material point method (MPM) [5,6], discontinuous deformation analysis (DDA) [7–9], and discrete element method (DEM) [10,11] have flourished. Advanced schemes like the fluid-solid coupling methods have been developed for disaster mitigation and complex engineering problems [12,13]. These numerical methods can readily solve forward problems. However, in scenarios where material properties or boundary conditions are not well-defined, classical numerical methods often cannot be directly applied and may require costly iterative invocations. Thus, inverse analysis and data assimilation pose challenges to classical numerical methods, hindering further application and development in digital twins.

The development of neural networks has provided a universal approximator to PDE solutions [14]. However, the data-driven way to train neural networks can incur unaffordable costs in scientific and engineering areas. Thus, the recently developed physics-informed neural network (PINN) [15] has emerged as a new paradigm to solve PDEs considering both physics and data. The key of classical PINNs is to incorporate both residuals from strong-form PDEs and errors from labeled data into the loss function. Thus, conventional forward and inverse analysis of various PDEs can be solved in a unified way by minimizing this loss function [16]. Extensive progress has been made in porous flow [17,18], constitutive modeling [19,20], earthquakes [21], fracture modeling [22], geoengineering [23,24], monitoring [25], etc.

Focusing on solid mechanics, fruitful work has been made to perform forward and inverse analysis with homogeneous materials. Haghighat et al. [26] proposed a multi-network model for inverse analysis using labeled displacements and stresses with or without their derivatives. Abueidda et al. [27] applied PINN on forward analysis of a three-dimensional cantilever beam without labeled data. Vahab [28] solved forward biharmonic equations of elasticity. Roy et al. [29] developed a PINN formulation for linear elastic plate theory and practiced forward analysis. On the other hand, a variant of PINN, the deep energy method (DEM) [30] or deep Ritz method [31] employs the functional as the loss function. Li et al. [32] compared DEM and conventional PINN in forward analysis of elastic plates with finite deformation. Nguyen-Thanh et al. [33] used a parametric/reference space to accelerate DEM forward analysis in elasticity and strain gradient elasticity. Wang et al. [34] implemented exact Dirichlet boundary conditions in forward analysis of elastic solid mechanics.

PINN methods can also be combined with classical numerical methods to circumvent sampling points and reduce automatic differentiations, for example, the FEM and PD. Wang et al. [35] reviewed PINN methods leveraging FEM in three genres. Our work can be defined as the way with a fully differentiable computational graph. In this way, Gao et al. [36] predicted nodal variables of a finite element mesh by a graph convolutional network (GCN). Motiwale et al. [37] used fully-connected neural networks (FCNN) with FEM. Wang et al. [35] implemented FCNN, modified FCNN, and GCN with FEM, and systematically revealed the role of finite element mesh in three-dimensional cases. Millions of degrees of freedom were considered in Wang et al. [35]. PD formulations were also absorbed in PINNs to solve solids with cracks [38], cracks of correspondence materials [39], thermomechanical loadings [40], and crack initiation/propagation in rock-like materials [41]. Alternatively, an advanced graph transformer was proposed as the surrogate model to predict long-term evolution of physics systems [42]. In these studies, blackbox weights and biases are entailed in problems with homogeneous materials.

In solid mechanics with homogeneous materials, concentration features often take place near fixed boundaries, local tractions, and cracks. Conventional PINNs and DEM usually end up with low accuracy in such scenarios. To solve these problems, peridynamic differential operator, mixed-form residuals, Fourier feature mapping, and some other methods were developed. Haghighat et al. [43] devised a nonlocal PINN using the peridynamic differential operator to remove spatial derivatives and solve the inverse analysis of an indentation problem with sharp gradients. Fuhg and Bouklas [44] proposed the mixed DEM method to solve forward hyperelastic problems with concentration features. Abueidda et al. [45] mixed strong-form residuals and the energy functional in the loss function to solve forward hyperelastic problem. Chadha et al. [46] applied random Fourier feature mapping to DEM method. Jeong et al. [47] adopted a two-stage learning strategy for forward analysis of the specimen under uniaxial tension with a crack. With extra output from neural networks or additional operations, these methods can well reproduce concentration features in homogeneous materials.

Another challenge arises when dealing with heterogeneous materials, which are crucial in damaged structures, composite materials, and geotechnical engineering. Heterogeneity with discontinuous material interfaces also introduces concentration features and is more difficult for inverse analysis. Wang et al. [48] developed domain decomposition for DEM and solved forward heterogeneous problems with two different materials. Diao et al. [49] also embedded domain decomposition in PINN with multi-task learning for forward analysis of two-material heterogeneity. Zhang et al. [50] considered a two-material inverse analysis and also analyzed geometric parameters. Rezaei et al. [51] included both energy and strong-form residual in the loss to handle forward heterogeneous problems with two materials. Harandi et al. [52] employed a similar formulation for forward analysis and parametric learning of a

thermo-mechanically coupled problem with two materials. Ren and Lyu [53] utilized mixed-form PINN based on dimensionless strong form PDEs to solve forward problems with two-phase random materials. These applications are mostly investigated with simple geometry and experimental setup.

Recently, advancements utilizing PINN have been presented for forward analysis in engineering scenarios. For example, with normalization factors into the loss function, Ouyang [54] simplified the pile as a one-dimensional (1D) Euler–Bernoulli beam with soil resistance described by p-y curves. However, in this study, we focus on the inverse analysis in solid mechanics, especially the application of PINNs to engineering structures. For instance, considering two-dimensional (2D) linear elastic heterogeneous strata, Vahab et al. [55] employed domain decomposition for inverse analysis of the modulus of a 2D bottom layer of soil. Zhang et al. [56] examined a tunnel in 2D linear elastic homogeneous strata and performed inverse discovery of Lame parameters. He et al. [57,58] simplified the immersed tunnel into a 1D linear elastic multi-beam model to estimate the foundation modulus. Ouyang [59] performed back analysis for a parameter related to soil stiffness using a simplified 1D linear elastic beam. In addition to material parameters, inverse analysis is also crucial for unknown boundary loadings. For example, using multi-task learning, Xu et al. [60] predicted loadings on a 2D elastic tunnel ring. By simplifying the tunnel ring as a 1D linear elastic curved beam, Wang et al. [61] estimated loadings considering linear elastic soil. Despite the significant achievements, current PINN inverse analyses for engineering structures remain constrained to homogeneous parameters and 1D/2D cases.

PINNs are required to simultaneously handle physics constraints, data constraints, blackbox weights, and blackbox biases. Thus, PINNs are highly challenged in complex problems involving heterogeneous engineering structures. Existing enhancements to PINNs for heterogeneity, such as those employing mixed form and domain decomposition cannot guarantee universal improvements in efficiency and accuracy. Moreover, extra hyperparameters and design of neural networks in improved PINNs are not friendly for engineers to start with.

To perform efficient inverse analysis for multidimensional heterogeneous engineering structures, we develop a new Differentiable Finite Element Method (DFEM) based on Galerkin discretization. Using this new DFEM method, several nontrivial gains can be leveraged: (1) Rather than employing classical neural networks with blackbox weights and biases, the DFEM in this study directly encodes physics into a discretized differentiable network. The computational graph is fully interpretable, obviating the need for hyperparameters in neural network design. (2) The Dirichlet and Neumann boundary conditions, as well as labeled data, can be directly encoded into the DFEM, eliminating the need for multiple loss terms that may cause gradient pathology. (3) By only keeping interpretable trainable parameters and encoding constraints from data and physics, the DFEM facilitates more accurate inverse analyses of 2D and 3D engineering structures with significantly faster convergence and higher accuracy.

As follows, further introduction of PINN and its challenges are presented. Subsequently, the methodology of the new DFEM, discretized using the Galerkin method, is detailed. Section 4 validates the superior accuracy and efficiency of the DFEM by reproducing some typical published case studies. The inverse analysis of 3D engineering structures is performed in Section 5. Discussions and conclusions offer relevant insights, along with the advantages and possible limitations of the approach.

2. PINN and its challenges

The classical PINN in Raissi et al. [15] incorporates residuals of strong-form PDEs and mean square errors of labeled data. Collocation points are sampled inside the domain and on the boundaries. Haghighat et al. [26] investigated PINN in solid mechanics with the strong-form PDE:

$$\begin{aligned} &\nabla \cdot \boldsymbol{\sigma} + \boldsymbol{f} = 0, \ \text{on } \Omega \\ &\boldsymbol{u} = \boldsymbol{g}, \ \text{on } \partial\Omega_{\mathrm{D}} \\ &\boldsymbol{\sigma} \cdot \boldsymbol{n} = \boldsymbol{t}, \ \text{on } \partial\Omega_{\mathrm{N}} \end{aligned} \tag{1}$$

where $\boldsymbol{\sigma} \in \mathrm{R}^{n_{\mathrm{dim}} \times n_{\mathrm{dim}}}$ is the Cauchy stress tensor. $\boldsymbol{f} \in R^{n_{\mathrm{dim}}}$ is the body force. $\boldsymbol{u} \in \mathrm{R}^{n_u}$ is the displacement. $\boldsymbol{g} \in \mathrm{R}^{n_{\mathrm{dim}}}$ is the prescribed Dirichlet boundary value on $\partial\Omega_{\mathrm{D}}$. $\boldsymbol{t} \in \mathrm{R}^{n_{\mathrm{dim}}}$ is the force on Neumann boundaries $\partial\Omega_{\mathrm{N}}$ with outer normal $\boldsymbol{n} \in \mathrm{R}^{n_{\mathrm{dim}}}$. n_{dim} is the spatial dimension. n_u is the dimension of target variables. The linear constitutive and small strain are

$$\begin{aligned} &\boldsymbol{\sigma} = \lambda \mathrm{tr}(\boldsymbol{\varepsilon})\mathbf{I} + 2\mu\boldsymbol{\varepsilon} \\ &\boldsymbol{\varepsilon} = \frac{1}{2}\left(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^{\mathrm{T}}\right) = \nabla_s \boldsymbol{u} \end{aligned} \tag{2}$$

where λ and μ are Lamé constants, which can be computed with the Young's modulus E and Poisson's ratio ν:

$$\begin{aligned} &\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)} \\ &\mu = \frac{E}{2(1+\nu)} \end{aligned} \tag{3}$$

Neural networks are used in PINN to predict displacements $\boldsymbol{u}$:

$$\begin{aligned}
&\boldsymbol{u}(\boldsymbol{x}) = \left(A_{n_l} \circ \sigma_{n_l-1} \circ A_{n_l-1} \circ \cdots \circ \sigma_1 \circ A_1\right)(\boldsymbol{x}) \\
&A_i(\boldsymbol{x}) = \boldsymbol{W}_i \boldsymbol{x} + \boldsymbol{b}_i \\
&\sigma_i(\boldsymbol{x}) = \mathrm{RELU}(\boldsymbol{x}) = \max(0, \boldsymbol{x})
\end{aligned} \tag{4}$$

A typical neural network is composed of linear transformations A_i and nonlinear activation functions σ_i. The type of activation functions, as well as depth and width of neural networks are hyperparameters that are commonly determined by trial-and-error. Trainable parameters of neural networks typically include black box weights $\boldsymbol{W}_i$ and biases $\boldsymbol{b}_i$. The influence of weights and biases on the model's performance is implicit and can hardly be quantified or controlled. PINN leverages the universal approximation ability of neural networks to predict the field variable over the entire domain. With the predicted displacement $\boldsymbol{u}$, strain $\boldsymbol{\varepsilon}$ and stress $\boldsymbol{\sigma}$ can be derived using Eq. (2). In mixed-form PINN [44], strain $\boldsymbol{\varepsilon}$ and stress $\boldsymbol{\sigma}$ can be predicted directly from a neural network. After getting these variables, the loss function of PINN can be computed with

$$\begin{aligned}
&L^{\mathrm{PINN}} = \lambda_\Omega L_\Omega^{\mathrm{S}} + \lambda_{\partial\Omega_{\mathrm{D}}} L_{\partial\Omega_{\mathrm{D}}} + \lambda_{\partial\Omega_{\mathrm{N}}} L_{\partial\Omega_{\mathrm{N}}} + \lambda_{\mathrm{data}} L_{\mathrm{data}} \\
&L_\Omega^{\mathrm{S}} = \frac{1}{N_\Omega} \sum_{i=1}^{N_\Omega} \left| \nabla \cdot \boldsymbol{\sigma}\left(\boldsymbol{x}_{\Omega_i}\right) + \boldsymbol{f}\left(\boldsymbol{x}_{\Omega_i}\right) \right| \\
&L_{\partial\Omega_{\mathrm{D}}} = \frac{1}{N_{\partial\Omega_{\mathrm{D}}}} \sum_{i=1}^{N_{\partial\Omega_{\mathrm{D}}}} \left| \boldsymbol{u}\left(\boldsymbol{x}_{\partial\Omega_{\mathrm{D}i}}\right) - \boldsymbol{g}\left(\boldsymbol{x}_{\partial\Omega_{\mathrm{D}i}}\right) \right| \\
&L_{\partial\Omega_{\mathrm{N}}} = \frac{1}{N_{\partial\Omega_{\mathrm{N}}}} \sum_{i=1}^{N_{\partial\Omega_{\mathrm{N}}}} \left| \boldsymbol{\sigma}\left(\boldsymbol{x}_{\partial\Omega_{\mathrm{N}i}}\right) \cdot \boldsymbol{n} - \boldsymbol{t}\left(\boldsymbol{x}_{\partial\Omega_{\mathrm{N}i}}\right) \right| \\
&L_{\mathrm{data}} = L_{\mathrm{data}_u} + L_{\mathrm{data}_\varepsilon} + L_{\mathrm{data}_\sigma} \\
&L_{\mathrm{data}_u} = \frac{1}{N_{\mathrm{data}_u}} \sum_{i=1}^{N_{\mathrm{data}_u}} \left| \boldsymbol{u}\left(\boldsymbol{x}_{\mathrm{data}_{ui}}\right) - \boldsymbol{u}^*\left(\boldsymbol{x}_{\mathrm{data}_{ui}}\right) \right| \\
&L_{\mathrm{data}_\varepsilon} = \frac{1}{N_{\mathrm{data}_\varepsilon}} \sum_{i=1}^{N_{\mathrm{data}_\varepsilon}} \left| \boldsymbol{\varepsilon}\left(\boldsymbol{x}_{\mathrm{data}_{\varepsilon i}}\right) - \boldsymbol{\varepsilon}^*\left(\boldsymbol{x}_{\mathrm{data}_{\varepsilon i}}\right) \right| \\
&L_{\mathrm{data}_\sigma} = \frac{1}{N_{\mathrm{data}_\sigma}} \sum_{i=1}^{N_{\mathrm{data}_\sigma}} \left| \boldsymbol{\sigma}\left(\boldsymbol{x}_{\mathrm{data}_{\sigma i}}\right) - \boldsymbol{\sigma}^*\left(\boldsymbol{x}_{\mathrm{data}_{\sigma i}}\right) \right|
\end{aligned} \tag{5}$$

where $|\circ|$ denotes the square error. The total loss L^{PINN} is composed of the loss from the strong form governing equation L_Ω^{S}, loss from Dirichlet boundary conditions $L_{\partial\Omega_{\mathrm{D}}}$, loss from Neumann boundary conditions $L_{\partial\Omega_{\mathrm{N}}}$, and loss from labeled data L_{data}. L_{data_u}, $L_{\mathrm{data}_\varepsilon}$, and L_{data_σ} represent losses from labeled displacements, strains, and stresses, respectively. λ_Ω, $\lambda_{\partial\Omega_{\mathrm{D}}}$, $\lambda_{\partial\Omega_{\mathrm{N}}}$, and λ_{data} denote weights of each loss term, they can be adjusted to mitigate the gradient pathology problem caused by multiple unbalanced losses. An adaptive strategy applies trainable weights to each sampling point and accelerate convergence [62], or we can sample the collocation points adaptively according to a distribution proportional to the loss function [63]. Alternatively, multiple losses can be balanced using multitask learning techniques [64]. The quantities with asterisks denote the given labeled data. Haghighat et al. [43] used labeled data of displacements, strains, and stresses. Xu et al. [60] only took displacements as labeled data. Assuming $\boldsymbol{\theta}$ denotes the unknown parameters of material properties and boundary conditions. PINNs solve the inverse problem by minimizing a loss function with respect to $\boldsymbol{\theta}$, $\boldsymbol{W}_i$, and $\boldsymbol{b}_i$. Physics softly informs PINNs during this minimization process.

However, PINNs are facing challenges during applications in solid mechanics. Both classical PINN and DEM cannot readily reproduce concentration features, like those near fixed boundaries, local tractions, and material interfaces [26,44,49]. Moreover, the inverse analysis for boundary loadings using a vanilla PINN is hard to converge [60]. The neural network itself is a black box, while interpretability is crucial and often desired in modeling physical system [65]. The necessity of simultaneously handling physics constraints, data constraints, blackbox weights, and blackbox biases hinders the convergence and efficient application of PINNs. For example, Wang et al. [66] identified a common failure mode of PINN, in which gradients of the boundary loss term in each layer are sharply concentrated at the origin and overall attain significantly smaller values than the gradients of the PDE residual loss. Aiming to solve these problems, this study develops a new DFEM method that is fully interpretable and can resolve fine features with fast convergence in inverse analysis.

Classical numerical methods and neural networks are closely related. For instance, Rao et al. [67] encoded the physics by reformulating the explicit forward Euler time integration as a recurrent neural network (RNN). Furthermore, neural network architectures

like RNNs and ResNet (residual neural network) have been demonstrated to be closely connected to Runge-Kutta methods, which are well-established classical numerical integration techniques [68,69]. Rao et al. [67] and Karnakov et al. [70] used mesh grids to discretize field variables. They adopted finite difference methods to discretize variables on structured Cartesian grids, which cannot easily deal with complex geometries. For example, extra penalization parameters and body fraction functions have to be constructed to consider irregular boundaries in Karnakov et al. [70]. In comparison, we use weak-form PDEs to reduce the smoothness requirement of field variables and finite element approximation to readily handle complex engineering geometries. Thus, the finite element formulation used in this study can have nontrivial gains compared with the finite difference or finite volume formulation in Rao et al. [67] and Karnakov et al. [70] for multidimensional engineering structures. Wang and Yin [71] utilized the finite element mesh to discretize field variables with the Ritz energy functional as the loss function. Significant speedup and accuracy were achieved in cases with concentration features and heterogeneous hyperelasticity. Its training from zero can be faster than FEM. The inverse analysis for an unknown Dirichlet boundary condition was performed. However, the Galerkin loss in this study is more proper for diverse inverse analysis.

3. Methodology of DFEM

Instead of using the blackbox deep neural networks and softly informing the physics in the loss function with PDE residuals, in this section, we design an interpretable differentiable network directly based on physics. Physics will be introduced in its weak form, which is then discretized using the Galerkin method.

3.1. Physics in terms of weak form PDE

Here we take a test function $\boldsymbol{w}(\boldsymbol{x}) \in \mathrm{R}^{n_u}$ and $\boldsymbol{u}(\boldsymbol{x}) \in R^{n_u}$ as the trial function. $\boldsymbol{u}$ is assumed to satisfy Dirichlet boundary conditions, and $\boldsymbol{w} = 0$ on Dirichlet boundaries. Multiplying the test function $\boldsymbol{w}$ to the strong-form equations in Eq. (1) and integrating over corresponding domains lead to

$$\int_{\Omega} \boldsymbol{w} \cdot (-\nabla \cdot \boldsymbol{\sigma} - \boldsymbol{f}) \mathrm{d}\Omega + \int_{\partial\Omega_{\mathrm{N}}} \boldsymbol{w} \cdot (\boldsymbol{\sigma} \cdot \boldsymbol{n} - \boldsymbol{t}) \mathrm{d}\Gamma = 0 \tag{6}$$

Applying integration by parts and the divergence theorem, the following weak form should hold for arbitrary test function $\boldsymbol{w}$:

$$\int_{\Omega} \nabla_s \boldsymbol{w} : \boldsymbol{\sigma} \mathrm{d}\Omega - \int_{\Omega} \boldsymbol{w} \cdot \boldsymbol{f} \mathrm{d}\Omega - \int_{\partial\Omega_{\mathrm{N}}} \boldsymbol{w} \cdot \boldsymbol{t} \mathrm{d}\Gamma = 0 \tag{7}$$

where $\boldsymbol{\sigma}$ is computed from the first-order derivative of $\boldsymbol{u}$. In the first term of Eq. (7), the first-order derivative of $\boldsymbol{u}$ and $\boldsymbol{w}$ should be square integrable, namely $\boldsymbol{u} \in H^1(\Omega)$ and $\boldsymbol{w} \in H^1(\Omega)$, where $H^1(\Omega)$ is the interested Sobolev space. This is a weaker requirement compared with that in strong-form Eq. (1). To facilitate the efficient implementation into a deep learning framework, we present the vector form corresponding to the tensor representation.

The constitutive in vector form is:

$$[\boldsymbol{\sigma}] = [\mathbf{D}][\boldsymbol{\varepsilon}] \tag{8}$$

where $[\mathbf{D}]$ is the elasticity matrix. $[\boldsymbol{\sigma}]$ and $[\boldsymbol{\varepsilon}]$ are vectors of stress and strain, respectively. In 2D plane strain scenarios, $\varepsilon_{33} = \varepsilon_{23} = \varepsilon_{31} = 0$, $\sigma_{33} = \lambda(\varepsilon_{11} + \varepsilon_{22})$. The strain vector, stress vector, and elasticity matrix are:

$$\begin{aligned} [\boldsymbol{\varepsilon}] &= [\,\varepsilon_{11} \quad \varepsilon_{22} \quad 2\varepsilon_{12}\,]^{\mathrm{T}} \\ [\boldsymbol{\sigma}] &= [\,\sigma_{11} \quad \sigma_{22} \quad \sigma_{12}\,]^{\mathrm{T}} \\ [\mathbf{D}] &= \begin{bmatrix} \lambda + 2\mu & \lambda & 0 \\ \lambda & \lambda + 2\mu & 0 \\ 0 & 0 & \mu \end{bmatrix} \end{aligned} \tag{9}$$

In 2D plane stress scenarios, $\sigma_{33} = \sigma_{23} = \sigma_{31} = 0$:

$$
\begin{aligned}
&[\boldsymbol{\varepsilon}] = [\,\varepsilon_{11} \quad \varepsilon_{22} \quad 2\varepsilon_{12}\,]^{\mathrm{T}} \\
&[\boldsymbol{\sigma}] = [\,\sigma_{11} \quad \sigma_{22} \quad \sigma_{12}\,]^{\mathrm{T}} \\
&[\mathbf{D}] = \frac{E}{(1-\nu^2)}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & (1-\nu)/2 \end{bmatrix}
\end{aligned} \tag{10}
$$

In 3D cases:

$$
\begin{aligned}
&[\varepsilon] = [\,\varepsilon_{11} \quad \varepsilon_{22} \quad \varepsilon_{33} \quad 2\varepsilon_{12} \quad 2\varepsilon_{23} \quad 2\varepsilon_{31}\,]^{\mathrm{T}} \\
&[\sigma] = [\,\sigma_{11} \quad \sigma_{22} \quad \sigma_{33} \quad \sigma_{12} \quad \sigma_{23} \quad \sigma_{31}\,]^{\mathrm{T}} \\
&[\mathbf{D}] = \begin{bmatrix} \lambda+2\mu & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda+2\mu & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & \lambda+2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu \end{bmatrix}
\end{aligned} \tag{11}
$$

Denoting $[\boldsymbol{\varepsilon}(\boldsymbol{u})] = [\nabla_s \boldsymbol{u}]$, and $[\boldsymbol{\varepsilon}(\boldsymbol{w})] = [\nabla_s \boldsymbol{w}]$, the vector representation of the weak form in Eq. (7) is:

$$
\int_{\Omega} [\boldsymbol{\varepsilon}(\boldsymbol{w})]^{\mathrm{T}} [\mathbf{D}] [\boldsymbol{\varepsilon}(\boldsymbol{u})] \mathrm{d}\Omega - \int_{\Omega} [\boldsymbol{w}]^{\mathrm{T}} [\boldsymbol{f}] \mathrm{d}\Omega - \int_{\partial\Omega_{\mathrm{N}}} [\boldsymbol{w}]^{\mathrm{T}} [\boldsymbol{t}] \mathrm{d}\Gamma = 0 \tag{12}
$$

3.2. Differentiable computational graph with Galerkin discretization

In this section, we use the Galerkin method to discretize the weak-form physics and encode it into a differentiable network. Assuming there is a finite dimensional subspace $U_h \subset H^1(\Omega)$, the discretized Galerkin formulation is to find $\boldsymbol{u}_h \in U_h$ such that the following equation holds for arbitrary $\boldsymbol{w}_h \in U_h$:

$$
\int_{\Omega} [\boldsymbol{\varepsilon}(\boldsymbol{w}_h)]^{\mathrm{T}} [\mathbf{D}] [\boldsymbol{\varepsilon}(\boldsymbol{u}_h)] \mathrm{d}\Omega - \int_{\Omega} [\boldsymbol{w}_h]^{\mathrm{T}} [\boldsymbol{f}] \mathrm{d}\Omega - \int_{\partial\Omega_{\mathrm{N}}} [\boldsymbol{w}_h]^{\mathrm{T}} [\boldsymbol{t}] \mathrm{d}\Gamma = 0 \tag{13}
$$

Global finite element basis functions $\{N_i\}_{i=1}^{N_{\mathrm{nodes}}}$ are used to build the space U_h. N_{nodes} is the number of all the nodes in a finite element mesh. For any point in the domain, it will be inside an element e, and we denote this elemental domain as Ω_e. The number of nodes in an element is n_{en}, each with n_u freedoms. n_{dim} equals n_u for solid mechanics. In element e, $\boldsymbol{u}^e_{i_{\mathrm{en}}}$ is the n_u degrees of freedom of node i_{en}. $N^e_{i_{\mathrm{en}}}$ is the nodal basis function of node i_{en}. In this study, linear Lagrange basis functions are utilized. $\boldsymbol{u}_h(\boldsymbol{x}) \in \mathrm{R}^{n_u}$ and its derivatives $\frac{\partial \boldsymbol{u}_h(\boldsymbol{x})}{\partial x_{i_{\mathrm{dim}}}} \in \mathrm{R}^{n_u}$ can be computed with:

$$
\begin{aligned}
&\boldsymbol{u}_h(\boldsymbol{x}) = \sum_{i_{\mathrm{en}}}^{n_{\mathrm{en}}} N^e_{i_{\mathrm{en}}}(\boldsymbol{x}) \boldsymbol{u}^e_{i_{\mathrm{en}}} \\
&\frac{\partial \boldsymbol{u}_h(\boldsymbol{x})}{\partial x_{i_{\mathrm{dim}}}} = \sum_{i_{\mathrm{en}}}^{n_{\mathrm{en}}} \frac{\partial N^e_{i_{\mathrm{en}}}(\boldsymbol{x})}{\partial x_{i_{\mathrm{dim}}}} \boldsymbol{u}^e_{i_{\mathrm{en}}}
\end{aligned} \tag{14}
$$

We write $\boldsymbol{u}_h$ in its vector form:

$$
\begin{aligned}
&\boldsymbol{u}_h = [\boldsymbol{N}^e][\boldsymbol{u}^e] \\
&[\boldsymbol{N}^e] = [\,N^e_1 \boldsymbol{I}_{n_u} \quad N^e_2 \boldsymbol{I}_{n_u} \quad \cdots \quad N^e_{n_{\mathrm{en}}} \boldsymbol{I}_{n_u}\,] \\
&[\boldsymbol{u}^e] = \begin{bmatrix} \boldsymbol{u}^e_1 \\ \boldsymbol{u}^e_2 \\ \vdots \\ \boldsymbol{u}^e_{n_{\mathrm{en}}} \end{bmatrix}
\end{aligned} \tag{15}
$$

where $\boldsymbol{I}_{n_u}$ is the $n_u \times n_u$ identity matrix, $[\boldsymbol{N}^e] \in \mathrm{R}^{n_u \times (n_u n_{\mathrm{en}})}$, $\boldsymbol{u}^e_{i_{\mathrm{en}}} \in \mathrm{R}^{n_u}$, and $[\boldsymbol{u}^e] \in \mathrm{R}^{n_u n_{\mathrm{en}}}$. The strain vector $[\boldsymbol{\varepsilon}(\boldsymbol{u}_h)]$ is

$$
\begin{aligned}
&[\boldsymbol{\varepsilon}(\boldsymbol{u}_h)] = [\boldsymbol{B}^e][\boldsymbol{u}^e] \\
&[\boldsymbol{B}^e] = \left[\begin{array}{cccc} \boldsymbol{B}_1^e & \boldsymbol{B}_2^e & \cdots & \boldsymbol{B}_{n_{\text{en}}}^e \end{array}\right] \\
&\left[\boldsymbol{B}_{i_{\text{en}}}^e\right] = \left[\begin{array}{ccc} \frac{\partial N_{i_{\text{en}}}^e}{\partial x_1} & 0 & 0 \\ 0 & \frac{\partial N_{i_{\text{en}}}^e}{\partial x_2} & 0 \\ 0 & 0 & \frac{\partial N_{i_{\text{en}}}^e}{\partial x_3} \\ \frac{\partial N_{i_{\text{en}}}^e}{\partial x_2} & \frac{\partial N_{i_{\text{en}}}^e}{\partial x_1} & 0 \\ 0 & \frac{\partial N_{i_{\text{en}}}^e}{\partial x_3} & \frac{\partial N_{i_{\text{en}}}^e}{\partial x_2} \\ \frac{\partial N_{i_{\text{en}}}^e}{\partial x_3} & 0 & \frac{\partial N_{i_{\text{en}}}^e}{\partial x_1} \end{array}\right] \text{ in 3D} \\
&\left[\boldsymbol{B}_{i_{\text{en}}}^e\right] = \left[\begin{array}{cc} \frac{\partial N_{i_{\text{en}}}^e}{\partial x_1} & 0 \\ 0 & \frac{\partial N_{i_{\text{en}}}^e}{\partial x_2} \\ \frac{\partial N_{i_{\text{en}}}^e}{\partial x_1} & \frac{\partial N_{i_{\text{en}}}^e}{\partial x_2} \end{array}\right] \text{ in 2D}
\end{aligned}
\tag{16}
$$

In 3D scenarios, a node is assigned three unknown displacement variables. In this case, $[\boldsymbol{\varepsilon}(\boldsymbol{u}_h)] \in \mathrm{R}^6$, $\left[\boldsymbol{B}_{i_{\text{en}}}^e\right] \in \mathrm{R}^{6\times 3}$, $[\boldsymbol{B}^e] \in \mathrm{R}^{6\times 3n_{\text{en}}}$. For a fixed point, the computation to get $\boldsymbol{u}_h$ and $[\boldsymbol{\varepsilon}(\boldsymbol{u}_h)]$ from nodal displacements $[\boldsymbol{u}^e]$ of an element e is a linear transformation, which is like the linear transformation by A_i in the neural network of Eq. (4). In contrast to stacking linear transformations and nonlinear activation functions in depth to build neural networks, the DFEM uses piecewise finite element approximation and combines them in spatial width. Neglecting the interfaces between elements, the weak form of an element e is:

$$
\int_{\Omega_e} [\boldsymbol{\varepsilon}(\boldsymbol{w}_h)]^{\mathrm{T}}[\mathbf{D}][\boldsymbol{\varepsilon}(\boldsymbol{u}_h)]\mathrm{d}\Omega - \int_{\Omega_e} [\boldsymbol{w}_h]^{\mathrm{T}}[\boldsymbol{f}]\mathrm{d}\Omega - \int_{\partial\Omega_{e\mathrm{N}}} [\boldsymbol{w}_h]^{\mathrm{T}}[\boldsymbol{t}]\mathrm{d}\Gamma = 0 \tag{17}
$$

We have got $[\boldsymbol{\varepsilon}(\boldsymbol{w}_h)] = [\boldsymbol{B}^e][\boldsymbol{w}^e]$, $[\boldsymbol{\varepsilon}(\boldsymbol{u}_h)] = [\boldsymbol{B}^e][\boldsymbol{u}^e]$, $[\boldsymbol{w}_h] = [\boldsymbol{N}^e][\boldsymbol{w}^e]$, and Eq. (17) becomes

$$
\begin{aligned}
&[\boldsymbol{w}^e]^{\mathrm{T}}[\boldsymbol{R}^e] = 0 \\
&[\boldsymbol{R}^e] = \left[\int_{\Omega_e} [\boldsymbol{B}^e]^{\mathrm{T}}[\mathbf{D}][\boldsymbol{\varepsilon}(\boldsymbol{u}_h)]\mathrm{d}\Omega\right] - \int_{\Omega_e} [\boldsymbol{N}^e]^{\mathrm{T}}[\boldsymbol{f}]\mathrm{d}\Omega - \int_{\partial\Omega_{e\mathrm{N}}} [\boldsymbol{N}^e]^{\mathrm{T}}[\boldsymbol{t}]\mathrm{d}\Gamma \Leftrightarrow \\
&[\boldsymbol{R}^e] = \left[\int_{\Omega_e} [\boldsymbol{B}^e]^{\mathrm{T}}[\mathbf{D}][\boldsymbol{B}^e]\mathrm{d}\Omega\right][\boldsymbol{u}^e] - \int_{\Omega_e} [\boldsymbol{N}^e]^{\mathrm{T}}[\boldsymbol{f}]\mathrm{d}\Omega - \int_{\partial\Omega_{e\mathrm{N}}} [\boldsymbol{N}^e]^{\mathrm{T}}[\boldsymbol{t}]\mathrm{d}\Gamma
\end{aligned}
\tag{18}
$$

Given the arbitrariness of $[\boldsymbol{w}^e]$, a set of linear equations can be obtained as $[\boldsymbol{R}^e] = 0$. In FEM, $\int_{\Omega_e} [\boldsymbol{B}^e]^{\mathrm{T}}[\mathbf{D}][\boldsymbol{B}^e]\mathrm{d}\Omega$ will be assembled into the global stiffness matrix, and $\int_{\Omega_e} [\boldsymbol{N}^e]^{\mathrm{T}}[\boldsymbol{f}]\mathrm{d}\Omega + \int_{\partial\Omega_{e\mathrm{N}}} [\boldsymbol{N}^e]^{\mathrm{T}}[\boldsymbol{t}]\mathrm{d}\Gamma$ will be added into the global force vector. In the proposed DFEM with Galerkin discretization, the loss function only needs to consider $[\boldsymbol{R}^e] \in \mathrm{R}^{n_u n_{\text{en}}}$ as a 1D vector, instead of a 2D stiffness matrix. $[\boldsymbol{R}^e]$ will be collected into a global residual $[\boldsymbol{R}] \in \mathrm{R}^{n_{\text{dof}}}$, where n_{dof} is the number of total degrees of freedom. Gaussian quadrature is employed to break the integral into a weighted sum:

$$[\boldsymbol{R}^e] = \sum_{i_{\text{vquad}}}^{n_{\text{vquad}}} \beta_{i_{\text{vquad}}} \left[\boldsymbol{R}_v^e\left(\boldsymbol{x}_{i_{\text{vquad}}}\right)\right] + \sum_{i_{\text{squad}}}^{n_{\text{squad}}} \beta_{i_{\text{squad}}} \left[\boldsymbol{R}_s^e\left(\boldsymbol{x}_{i_{\text{squad}}}\right)\right] =$$
$$\left[\sum_{i_{\text{vquad}}}^{n_{\text{vquad}}} \beta_{i_{\text{vquad}}} \left[\boldsymbol{B}^e\left(\boldsymbol{x}_{i_{\text{vquad}}}\right)\right]^{\mathrm{T}} [\mathbf{D}] \left[\boldsymbol{B}^e\left(\boldsymbol{x}_{i_{\text{vquad}}}\right)\right]\right] [\boldsymbol{u}^e] - \tag{19}$$
$$\sum_{i_{\text{vquad}}}^{n_{\text{vquad}}} \beta_{i_{\text{vquad}}} \left[\boldsymbol{N}^e\left(\boldsymbol{x}_{i_{\text{vquad}}}\right)\right]^{\mathrm{T}} \left[\boldsymbol{f}\left(\boldsymbol{x}_{i_{\text{vquad}}}\right)\right] - \sum_{i_{\text{squad}}}^{n_{\text{squad}}} \beta_{i_{\text{squad}}} \left[\boldsymbol{N}^e\left(\boldsymbol{x}_{i_{\text{squad}}}\right)\right]^{\mathrm{T}} \left[\boldsymbol{t}\left(\boldsymbol{x}_{i_{\text{squad}}}\right)\right]$$

where $\left(\beta_{i_{\text{vquad}}}, \boldsymbol{x}_{i_{\text{vquad}}}\right)$ and $\left(\beta_{i_{\text{squad}}}, \boldsymbol{x}_{i_{\text{squad}}}\right)$ are Gaussian weights and points inside the domain and on boundaries, respectively. $\left[\boldsymbol{R}_v^e\left(\boldsymbol{x}_{i_{\text{vquad}}}\right)\right]$ is the residual computed on the volume Gaussian point $\boldsymbol{x}_{i_{\text{vquad}}}$. $\left[\boldsymbol{R}_s^e\left(\boldsymbol{x}_{i_{\text{squad}}}\right)\right]$ is the residual computed on the boundary Gaussian point $\boldsymbol{x}_{i_{\text{squad}}}$. Based on the above derivation, the DFEM with Galerkin discretization is built as shown in Fig. 1.

Fig. 1a presents the differentiable computational graph of the DFEM. Unknown nodal displacements $[\boldsymbol{u}^e]$, material parameters $[\mathbf{D}]$, volume forces $[\boldsymbol{f}]$, and boundary forces $[\boldsymbol{t}]$ are trainable parameters, which are marked green in Fig. 1a. If we already know some of them, they can be substituted as constants. Similarly, the Dirichlet boundary conditions and monitored displacements are encoded into $[\boldsymbol{u}^e]$ by directly substituting unknown variables with labeled data, while PINNs usually add data terms in the loss function like that in Eq. (5). The displacements and strains at a point can be interpolated using Eq. (15) and Eq. (16). This interpolation procedure is an interpretable linear transformation, while the linear transformation using A_i in Eq. (4) is a blackbox operation. The DFEM's nonlinearity comes from nonlinear piecewise basis functions. As shown in Fig. 1b, the 1D element e has two nodes with displacements $[u_i,$

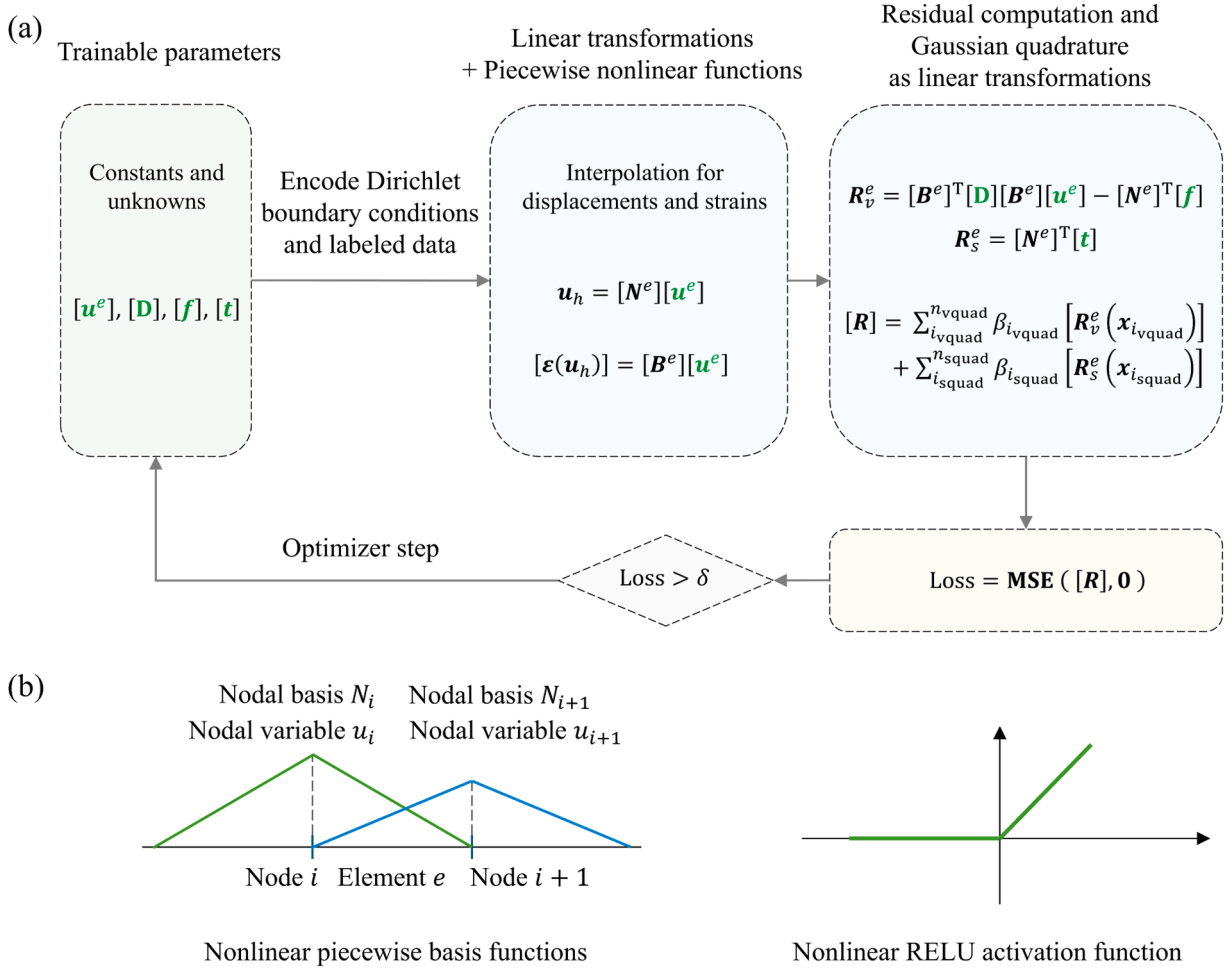


Fig. 1. The structure of the DFEM with Galerkin discretization: (a) The differentiable computational graph encodes weak form physics, Dirichlet boundary conditions, and labeled data (possible trainable parameters are marked green); (b) Comparison of nonlinear basis functions of the DFEM and the nonlinear RELU activation function in neural networks.

$u_{i+1}]$ and corresponding basis functions $[N_i, N_{i+1}]$. The displacement at element e is approximated with $[u_i, u_{i+1}]\begin{bmatrix} N_i \\ N_{i+1} \end{bmatrix}$. The basis function is nonlinear since it is effective only in its neighborhood, just like the RELU activation function which is nonzero only when the input is positive. The difference is that basis functions are distributed with partial overlaps across the spatial domain and are fully interpretable, while activation functions are stacked between linear transformations and are hard to interpret. With displacements and strains at Gaussian points, the residual is integrated over the domain and boundary surfaces. The weighted sum is also an interpretable linear transformation. The loss function is built by computing the mean square error of the overall residual. Leveraging the automatic differentiation of the computational graph, gradients of the loss function with respect to the trainable parameters are collected via backpropagation, circumventing the implementation of an adjoint framework. When the loss approaches zero, trainable parameters will gradually converge to the target value. Compared with PINNs, in DFEM, there are no blackbox weights and biases, and the target trainable parameters are fully interpretable. Thus, the DFEM can be regarded as a very sparse and fully interpretable neural network specialized in AI for Science. Moreover, this framework is also compatible with classical equation solvers, such as those utilizing Newton's method. Thus, it is perfectly valid and valuable to perform forward analysis given a specific weak form.

3.3. Physics-Encoded Numerical Network (PENN) as a novel paradigm

In this section, we put classical numerical methods, PINN, and the proposed DFEM in juxtaposition, demonstrating that the proposed DFEM is a kind of Physics-Encoded Numerical Network (PENN) as a novel paradigm to encode physics and data in a differentiable framework. In Fig. 2, these methods are mainly composed of three parts: approximation/discretization of field variables, PDE embedding, and solvers. Classical numerical methods usually use meshes (FEM, FDM, FVM) or particle methods (MPM, PD, SPH) to discretize field variables. The physics embodied in PDEs is thus transformed into different forms and organized into a simultaneous equation system. Finally, direct methods or iterative methods are used to solve the equation system. In comparison, the recently flourished PINN [15] directly uses a neural network to approximate field variables. It incorporates the PDE information by absorbing PDE residuals. Thus, it can perform forward and inverse analysis in a unified way. PINN also faces limitations of blackbox parameters and spectral bias [72], which limits its training efficiency and application to engineering cases. Thus, this study proposed the DFEM. The approximation of continuous field variables using finite element procedures has already been fully analyzed [73], and its powerful approximation ability in PDE solving is not inferior to neural networks. Since the same formulation can be readily applied to most classical numerical methods, Physics-Encoded Numerical Network (PENN) is utilized to name the novel paradigm. PENN is fully interpretable with classical approximation methods. Only necessary trainable parameters are used. PENN also circumvents spectral bias of classical neural networks and thus can easily handle problems with heterogeneous materials. Subsequent numerical experiments corroborate the superiority of PENN over the conventional PINN and some improved PINNs.

4. Benchmark experiments

This section reproduces two typical numerical experiments from published works about improved PINNs [26,60] to validate the DFEM's superior accuracy and efficiency. Firstly, a thick cylinder with unknown internal pressure is simulated to perform inverse analysis of the boundary loading. Then the test of indentation by a rigid punch is performed for the inverse analysis of material parameters. Double-precision floating point numbers are used in all cases. Taking $\boldsymbol{u}_x$ as the vector of the x-axis displacement component at all sampling points as an example, if the reference result is $\boldsymbol{u}_x^*$, error vector is computed with $\left(\boldsymbol{u}_x^* - \boldsymbol{u}_x\right) = (err_1, err_2, \cdots, err_n)$. To reduce the influence of magnitudes and get a reasonable estimation of the overall error, the relative L^2 error norm is defined as $\| \boldsymbol{u}_x^* - \boldsymbol{u}_{x2}\| / \| \boldsymbol{u}_x^*\|_2$. Since the DFEM uses the same approximation method as FEM, correct nodal displacements naturally lead to reasonable strain and stress.

4.1. Thick cylinder with internal pressure

In this section, we conduct the numerical experiment of a thick cylinder with unknown internal pressure, which is formulated as a plane stress problem. The configuration is set as the same with Xu et al. [60]. Fig. 3a is the geometry and boundary conditions of this experiment. Fig. 3b is the mesh used in COMSOL Multiphysics 6.2 and the DFEM. A quarter of the cylinder is taken for simplicity. The inner radius is 1m, and the outer radius is 5m. The center is used as the origin. The Neumann boundary is the inner pressure $P = 20$Pa. Dirichlet boundary conditions are $u_x(x = 0) = 0$, and $u_y(y = 0) = 0$. Young's modulus is $E = 1 \times 10^5$Pa, and Poisson's ratio is $\nu = 0.3$. This test can help us to understand components like pipes, pressure vessels, and other cylindrical structures. The DFEM is expected to perform better than PINN for the inverse analysis of internal pressure P.

Fig. 4 is reprinted from Xu et al. [60], in which the reference displacement components u_x and u_y are used as labeled data. The plain PINN failed in this case. With non-dimensionalization, a scaled PINN can get a P with relative error 2.2 % after 50,000 iterations. Improved PINNs like SA-PINN (scaled self-adaptive PINN) and scaled PINN with uncertainty weighting can further improve the accuracy of the unknown loading P. Detailed algorithms can be referred to in Xu et al. [60].

Commonly, an Adam optimizer using first-order derivatives will be used before an LBFGS optimizer to avoid being trapped at saddle points [74]. The LBFGS optimizer uses an approximation of the Hessian matrix to minimize the loss function. Thus, the convergence using LBFGS is faster than Adam by leveraging second-order information. Since the DFEM only keeps minimal number of trainable parameters, we can directly use the LBFGS optimizer in PyTorch for cases with simple loadings. In this case, we also utilize an

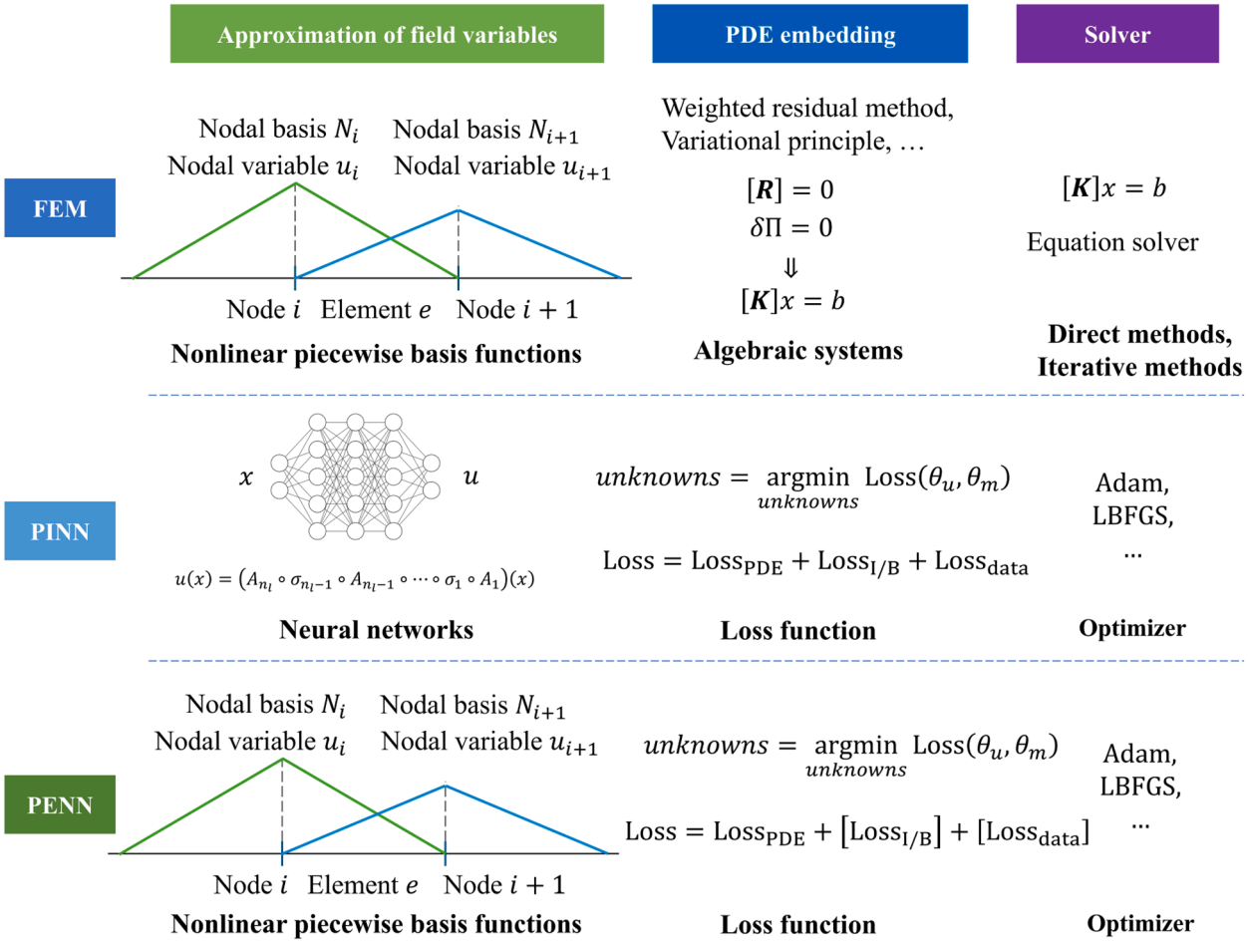


Fig. 2. A novel paradigm: Physics-Encoded Numerical Network (PENN) versus the classical Finite Element Method (FEM) and Physics-Informed Neural Network (PINN).

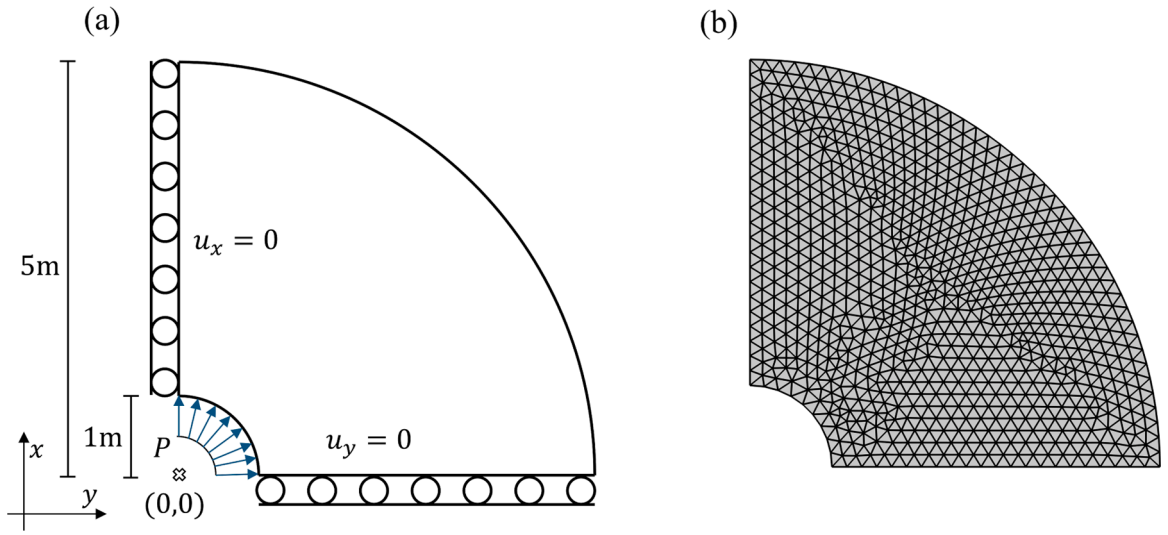


Fig. 3. Thick cylinder under internal pressure: (a) Geometry and boundary conditions; (b) Mesh for FEM and the PENN (DFEM).

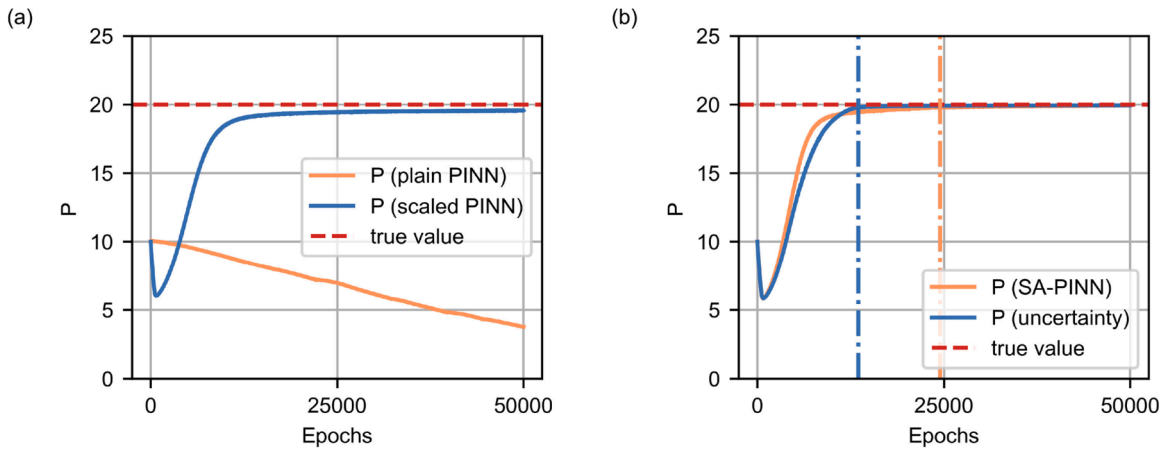


Fig. 4. Reprinted from Fig. 6 in Xu et al. [60]: Results of loads for the thick cylinder problem: (a) plain PINN versus scaled PINN and (b) SA-PINN versus PINN with uncertainty weighting. (The first epoch to converge to <1 % prediction error is marked with dash-dotted line).

Adam optimizer to compare the convergence iterations with PINN and improved PINNs. Referring to PyTorch documentation, the step size of the LBFGS optimizer is set as 1.0 with "strong_wolfe" line search function. Adam optimizer uses step size 1.0. Displacement components u_x and u_y at mesh nodes are used as labeled data. The DFEM converges in 150 Adam iterations like that in Fig. 5a and b. The loss curve slightly fluctuates in the beginning and stabilizes with high accuracy. The DFEM encodes labeled displacements u_x and u_y from FEM. Thus, the training loss only contains the mean square error of the residual as shown in Fig. 1. When using an LBFGS optimizer, only six steps are consumed to reach convergence. The loss and relative error share the same trend. It is obvious that there is no underfitting or overfitting. Fig. 6 gives displacement components u_x and u_y. Errors with respect to the reference displacements u_x^* and u_y^* are negligible.

Table 1 compares the accuracy and efficiency of the conventional PINN, improved PINNs in Xu et al. [60] and the DFEM in this study. The conventional plain PINN fails in this case. Three improved PINNs achieved satisfactory accuracy in 50,000 training iterations with an Adam optimizer. The DFEM, in comparison, achieves a relative error three orders of magnitude lower than improved PINNs with only 0.3 % Adam training iterations of improved PINNs. The DFEM trains well in only 0.31 s with an Adam optimizer. The LBFGS optimizer is much faster with 0.025 s. Such superior accuracy and speed can be attributed to two essential merits of the DFEM. The DFEM eliminates all implicit parameters and only keeps those interpretable and necessary. The DFEM encodes boundary conditions and labeled data into its interpretable architecture, circumventing gradient pathology and additional training to fit data constraints.

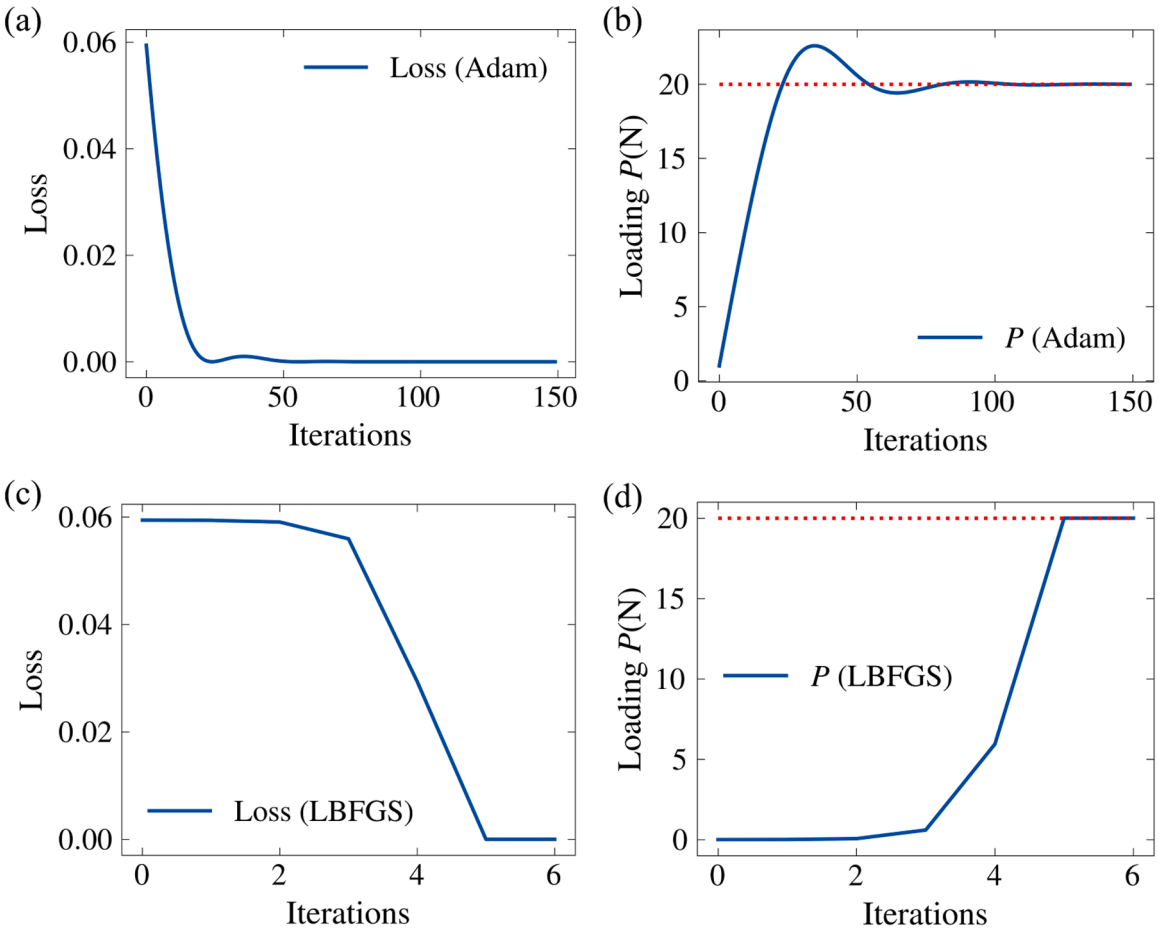


Fig. 5. Convergence with the PENN (DFEM) of the thick cylinder problem: (a) Loss and (b) Loading with respect to training iterations using the Adam optimizer; (c) Loss and (d) Loading with respect to training iterations using the LBFGS optimizer (red dotted line denotes the real value).

4.2. Indentation by a rigid punch

This section studies the problem of indentation by a rigid punch, which is formulated as a plane strain problem. The configuration is set after Haghighat et al. [43]. Fig. 7a presents the geometry and boundary conditions of the problem. Fig. 7b is the mesh used in COMSOL Multiphysics 6.2 and the DFEM. Dimensions of the domain are 1m $\times$ 1m. The left lower corner is set as the origin. Dirichlet boundary conditions are $u_x(x=0)=0$, $u_x(x=1)=0$, $u_x(y=0)=0$, and $u_y(y=0)=0$. The frictionless rigid punch with a width 0.2m exerts pressure by a downward displacement $u_y=\Delta=1\times10^{-3}$m from the upper center. It is a common mixed-type boundary, since the rest of the upper boundary is Neumann boundary with zero traction. The sharp transition of boundary condition leads to stress concentration, and the accuracy of PINN degrades in such scenarios [43]. Young's modulus is $E=7\times10^{10}$Pa, and Poisson's ratio is $\nu=0.3$. This case can help us to understand material behaviors in hardness testing, material analysis, and it also resembles the footing problem in geotechnical engineering. Haghighat et al. [43] proposed a nonlocal PINN to perform inverse analysis of Lamé elastic constants $\lambda=40.358\times10^{9}$Pa, $\mu=26.923\times10^{9}$Pa with labeled displacements, strains, and stresses from FEM. The DFEM uses reference displacements and modulus as labeled data.

Fig. 8 presents the convergence of loss and Lamé elastic constants. An Adam optimizer is used to compare the convergence iterations with PINN and improved PINNs. A second-order LBFGS will converge in much less iterations. Referring to PyTorch documentation, the step size of the LBFGS optimizer is set as 1.0 with "strong_wolfe" line search function. Adam optimizer uses step size 0.0001. In Fig. 8a and b with an Adam optimizer, a satisfactory accuracy of material parameters is achieved within 69 iterations. The LBFGS optimizer, in comparison, converges much faster. With only 12 iterations, the loss reaches a very small value, and the unknown material parameters converge to the real value. In both cases with the Adam optimizer and LBFGS optimizer, respectively, the material parameters slightly fluctuate in the beginning and stabilize with high accuracy. The DFEM encodes labeled displacements u_x and u_y from FEM. Thus, the training loss only contains the mean square error of the residual. Fig. 9 gives displacement components u_x and u_y. Errors with respect to the reference displacements u_x^* and u_y^* are negligible.

Table 2 compares the accuracy and efficiency of the conventional PINN, improved PINNs in Haghighat et al. [43] and the DFEM in

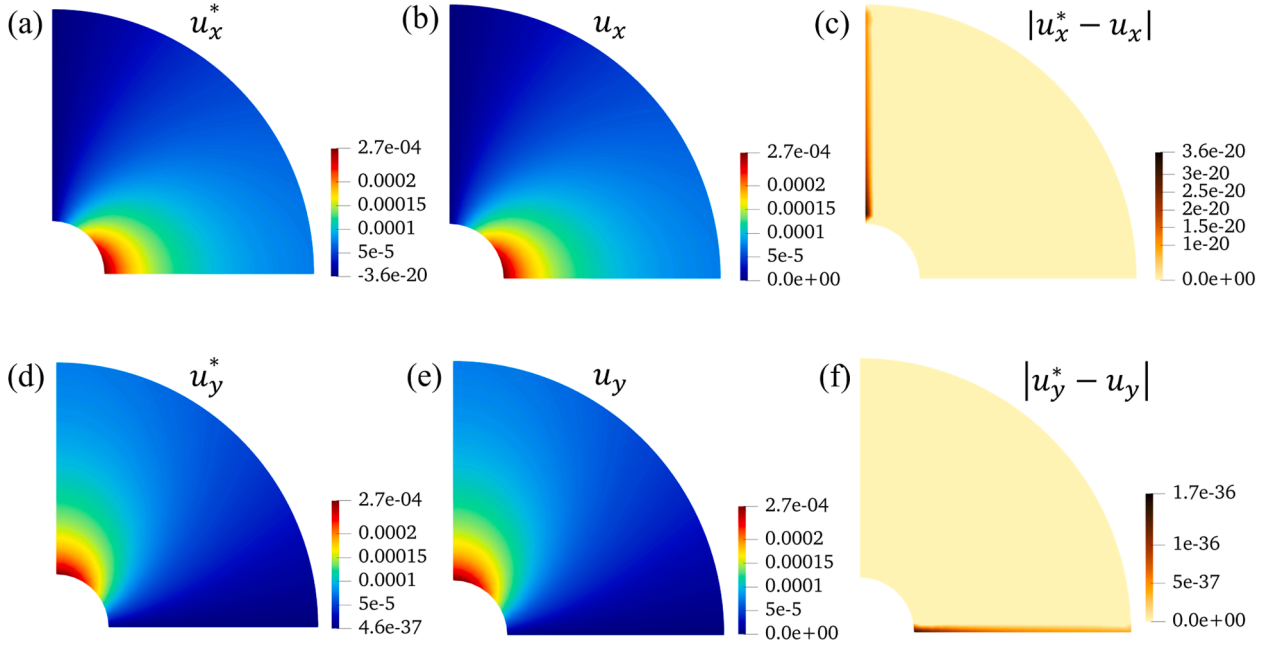


Fig. 6. Displacement components and errors of the thick cylinder: (a) Reference displacement along x axis of FEM; (b) Displacement along x axis of the PENN (DFEM); (c) The PENN (DFEM) error of displacement along x axis; (d) Reference displacement along y axis of FEM; (e) Displacement along y axis of the PENN (DFEM); (f) The PENN (DFEM) error of displacement along y axis.

Table 1
Comparison of the conventional PINN, improved PINN and PENN (DFEM) in the thick cylinder experiment.

Method	L^2 relative error			Computational cost	
	u_x	u_y	P	Iterations (Optimizer)	Time (Hardware)
Conventional PINN	–	–	–	–	–
Scaled PINN	1.2e-2	1.2e-2	2.2e-2	50,000 (Adam)	–
SA-PINN	1.8e-3	2.1e-3	3.5e-3	50,000 (Adam)	–
PINN with uncertainty weighting	3.5e-3	2.8e-3	3.6e-3	50,000 (Adam)	–
PENN (DFEM)	3.9e-17	1.7e-33	3.3e-6	150 (Adam) or 6 (LBFGS)	0.31s with Adam or 0.025 s with LBFGS (Intel i9–14900KF)

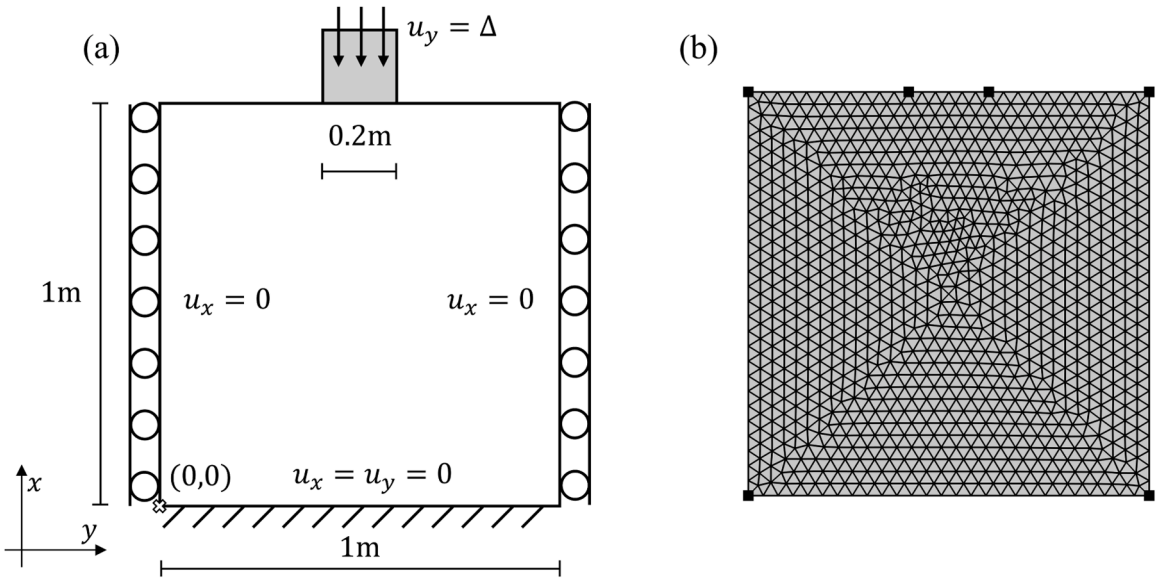


Fig. 7. Indentation by a rigid punch: (a) Geometry and boundary conditions; (b) Mesh for FEM and the PENN (DFEM).

this study. Two improved PINNs (AD-PDDO-PINN and PDDO-PINN) achieved higher accuracy than conventional PINN in 20,000 training iterations. In comparison, the DFEM achieves a relative error that is five orders of magnitude lower than the conventional PINN, while requiring only 0.34 % of the Adam training iterations needed for the conventional PINN. The DFEM trains well in only 0.27 s with an Adam optimizer. The LBFGS optimizer is much faster with 0.051 s. Good performance during the inverse analysis of material parameters validates the merits of the DFEM. The DFEM only contains minimal interpretable parameters. The DFEM encodes labeled data into its interpretable architecture without additional training cost.

Combining Galerkin loss with the DFEM enables extraordinary efficiency and accuracy in solving inverse analysis as shown in Fig. 8. When the previous Ritz/energy loss is utilized in this inverse analysis, the corresponding losses and unknown parameters are plotted in Fig. 10. Both the losses and parameters diverge towards the negative infinity when using the Adam or LBFGS optimizers. The failure of Ritz/energy loss for the inverse analysis of material parameters is as expected. The energy loss is

$$\text{energy} = \int_{\Omega} [\boldsymbol{\varepsilon}(\boldsymbol{u}_h)]^{\mathrm{T}} [\mathbf{D}] [\boldsymbol{\varepsilon}(\boldsymbol{u}_h)] \mathrm{d}\Omega - \int_{\Omega} [\boldsymbol{u}_h]^{\mathrm{T}} [\boldsymbol{f}] \mathrm{d}\Omega - \int_{\partial\Omega_{\mathrm{N}}} [\boldsymbol{u}_h]^{\mathrm{T}} [\boldsymbol{t}] \mathrm{d}\Gamma \tag{20}$$

Assuming all $\boldsymbol{u}_h$ are known, the energy boils down to the linear function of material parameters $[\mathbf{D}]$ and forces $[\boldsymbol{f}]$ and $[\boldsymbol{t}]$. Thus, minimizing the Ritz/energy loss with respect to unknown $[\mathbf{D}]$, $[\boldsymbol{f}]$, and $[\boldsymbol{t}]$ will drive the loss and unknown parameters to negative infinity. On the contrary, the Galerkin loss is proved capable of dealing with inverse analysis of material parameters and forces in Gao et al. [36] and this study. Moreover, the differentiable and interpretable network architecture in this study enables much faster and more accurate inverse analysis.

5. Inverse analysis of three-dimensional engineering structures

Previous 2D benchmark tests showcase the significantly improved accuracy and efficiency of the DFEM over the conventional PINN and some published improved PINNs. This section presents the inverse analysis of 3D engineering structures. A damaged cantilever beam is tested for the inverse analysis of heterogeneous material parameters. With partial observed data, unknown heterogeneous

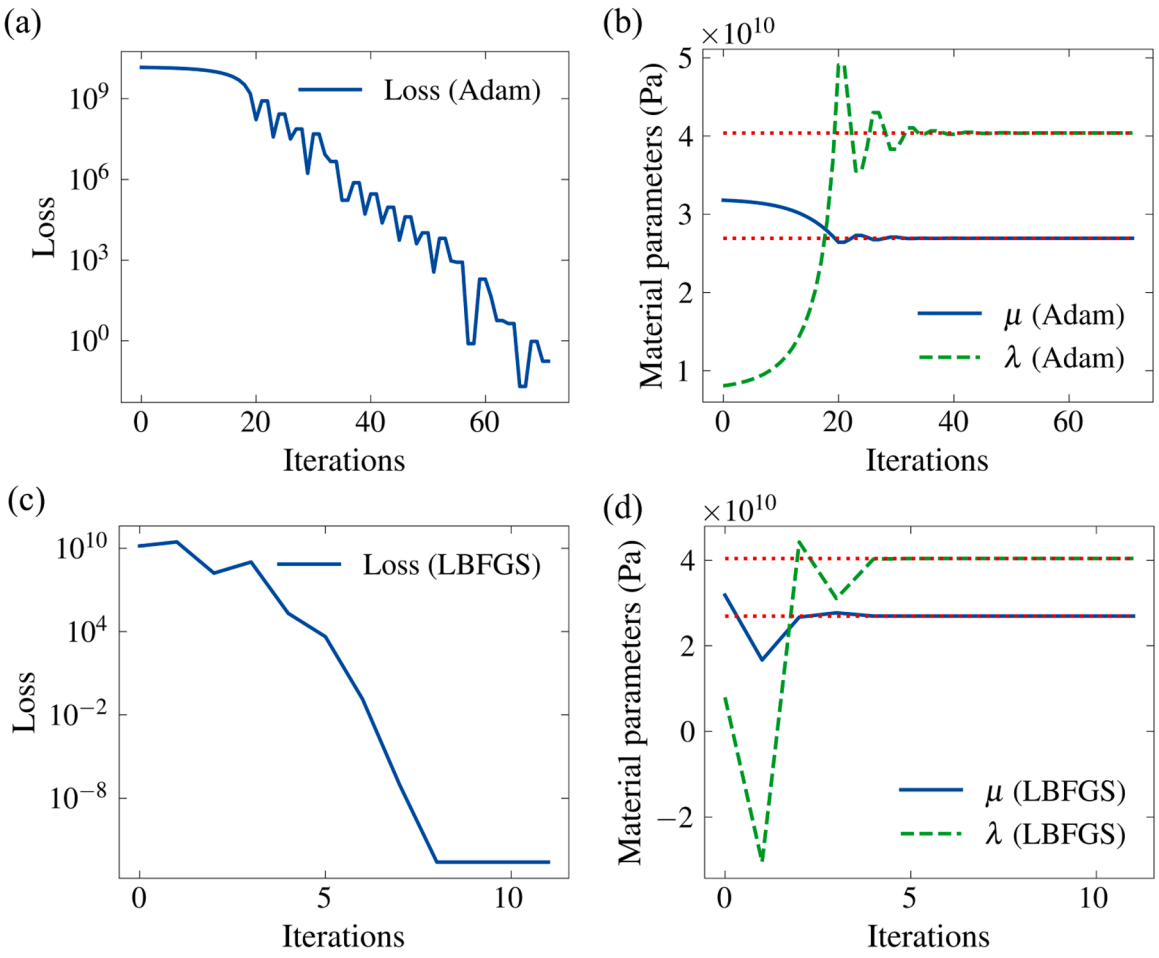


Fig. 8. Convergence of the PENN (DFEM) with Galerkin loss for the indentation problem: (a) Loss and (b) Material parameters with respect to training iterations using the Adam optimizer; (c) Loss and (d) Material parameters with respect to training iterations using the LBFGS optimizer (red dotted line denotes the real value).

loadings acting on the tunnel lining can be back calculated. The influence of noisy data is also discussed. The 3D geometry and mesh building are completed in COMSOL Multiphysics 6.2. The mesh is exported in the format of nastran file. The computational program is written in Python. The postprocess is realized with Python and Paraview. It seems DFEM cannot be easily embedded into commercial software like COMSOL or ABAQUS because the classical solvers are not used, and a fully differentiable computational process is not available in commercial software.

5.1. A damaged cantilever beam with ten unknown materials

A 3D cantilever beam is simulated for the inverse analysis of material parameters. The geometry and boundary conditions are presented in Fig. 11a. Fig. 11b is the mesh used in COMSOL Multiphysics 6.2 and the DFEM. Dimensions are 2m $\times$ 2m $\times$ 10m. It is fixed on the left corner. A distributed loading $P = 1 \times 10^6$Pa is uniformly applied on the right end. The initial Young's modulus and Poisson's ratio are $E = 2 \times 10^{10}$Pa and $\nu = 0.15$, which are homogeneous over the domain. We assume this cantilever beam is damaged after being used, and it becomes heterogeneous characterized with ten different degraded materials as shown in Fig. 11a. Parameters of the deteriorated materials are $[\mathrm{E}_1 \sim \mathrm{E}_{10}] = [1.0,\ 1.1,\ 1.2,\ 1.3,\ 1.4,\ 1.5,\ 1.6,\ 1.7,\ 1.8,\ 1.9] \times 10^{10}$Pa, and $[\nu_1 \sim \nu_{10}] = [0.2,\ 0.3,\ 0.2,\ 0.3,\ 0.2,\ 0.3,\ 0.2,\ 0.3,\ 0.2,\ 0.3] \times 10^{10}$Pa. For inverse analysis, the reference displacements from FEM are encoded into the DFEM architecture. There are ten materials with twenty unknown parameters, namely $[\mathrm{E}_1 \sim \mathrm{E}_{10}]$ and $[\nu_1 \sim \nu_{10}]$.

It is already validated in previous tests that the LBFGS optimizer will converge in much less iterations and computational time than the Adam optimizer. Thus, an LBFGS optimizer is utilized with step size 0.5 and "strong_wolfe" line search function. Fig. 12 presents the convergence of twenty unknown material parameters. These twenty material parameters are initialized from $E = 2 \times 10^{10}$Pa and $\nu = 0.15$. Young's modulus parameters $\mathrm{E}_1 \sim \mathrm{E}_{10}$ share the same trend. They initially decrease to a lower value than the real modulus, and then they converge to the target values. Poisson's ratios $\nu_1 \sim \nu_{10}$ also converge at the same time after some perturbations in the

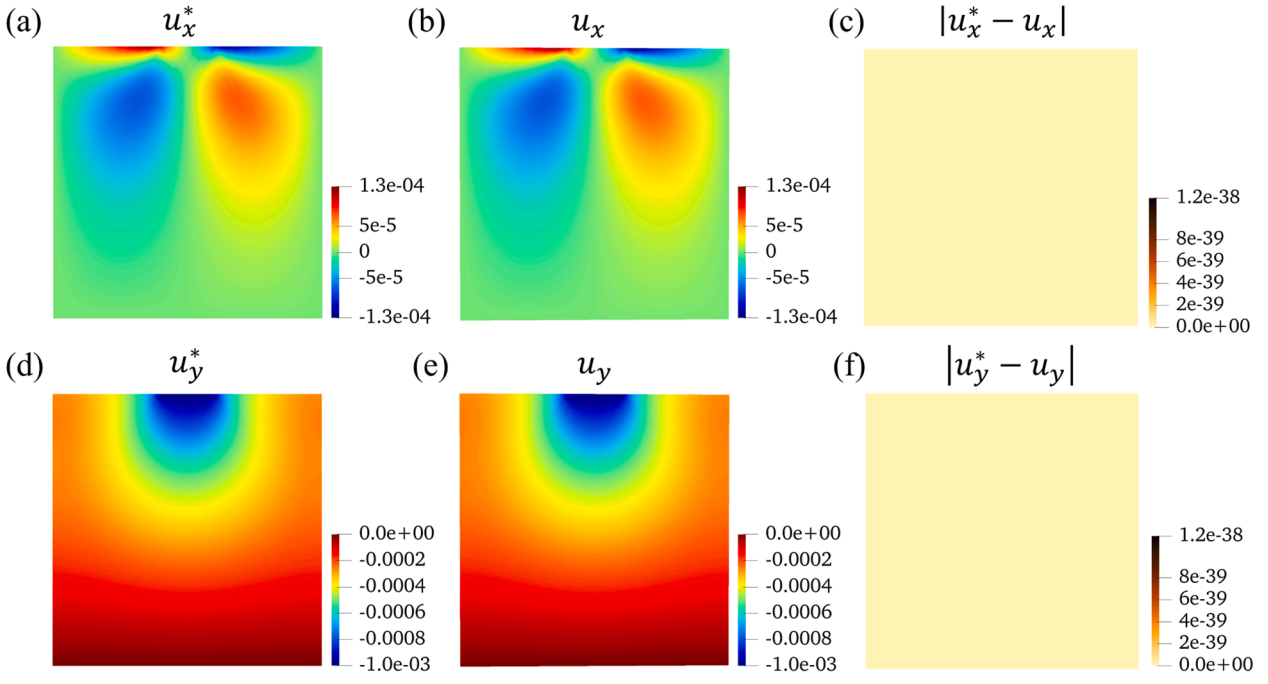


Fig. 9. Displacement components and errors of the indentation problem: (a) Reference displacement along x axis of FEM; (b) Displacement along x axis of the PENN (DFEM); (c) The PENN (DFEM) error of displacement along x axis; (d) Reference displacement along y axis of FEM; (e) Displacement along y axis of the PENN (DFEM); (f) The PENN (DFEM) error of displacement along y axis.

Table 2
Comparison between the conventional PINN, improved PINN and PENN (DFEM) in indentation experiment.

Method	L^2 relative error				Computational cost	
	u_x	u_y	λ	μ	Iterations (Optimizer)	Time (Hardware)
Conventional PINN	1.1e-2	3.2e-3	4.2e-3	2.5e-2	20,000 (Adam)	≈24,000 s (-)
AD-PDDO-PINN	1.1e-3	1.6e-3	2.4e-5	1.8e-5	20,000 (Adam)	≈34,000 s (-)
PDDO-PINN	2.1e-3	2.4e-3	9.1e-5	8.0e-4	20,000 (Adam)	≈21,000 s (-)
PENN (DFEM)	0	0	3.1e-6	3.1e-7	69 (Adam) or 12 (LBFGS)	0.27s with Adam or 0.051 s with LBFGS (Intel i9–14900KF)

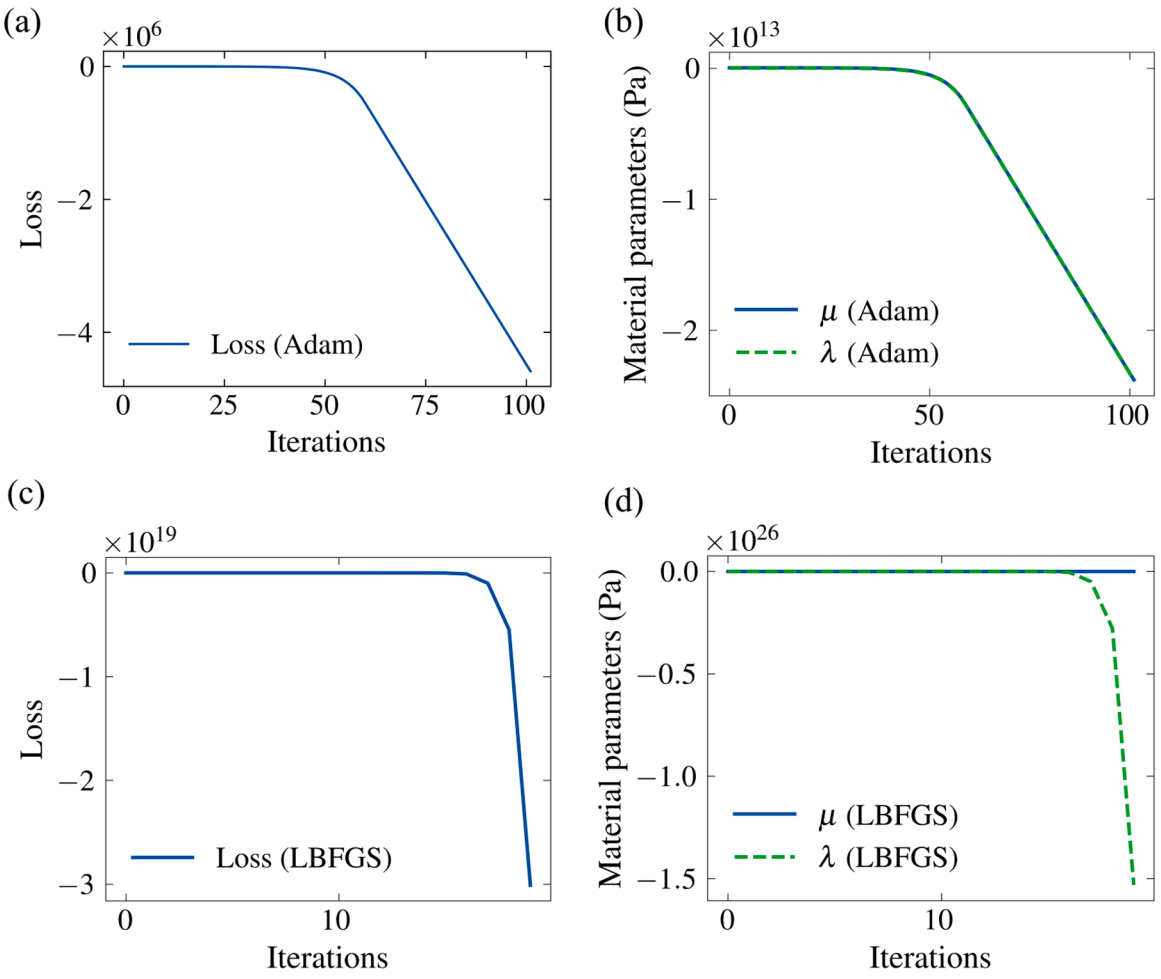


Fig. 10. Convergence of the PENN (DFEM) with Ritz/energy loss for the indentation problem: (a) Loss and (b) Material parameters with respect to training iterations using Adam optimizer; (c) Loss and (d) Material parameters with respect to training iterations using LBFGS optimizer.

beginning. These twenty unknown material parameters converge in just 117 iterations. This training process takes 1.5 s on a CPU (Intel i9–14900KF). Fig. 13 presents the displacements along three orthogonal axes. Since the DFEM strictly encodes labeled displacements from FEM, without the cost of training, the DFEM can maintain a very high accuracy. Relative errors of $E_1 \sim E_{10}$ are $[7.8 \times 10^{-12},\ 7.7 \times 10^{-12},\ 7.6 \times 10^{-12}, 7.5 \times 10^{-12},\ 7.4 \times 10^{-12},\ 7.1 \times 110^{-12},\ 7.7 \times 10^{-12},\ 6.6 \times 10^{-12},\ 7.4 \times 10^{-12},\ 4.4 \times 10^{-12}]$. Relative errors of $\nu_1 \sim \nu_{10}$ are $[3.7 \times 10^{-14},\ 4.3 \times 10^{-14},\ 3.2 \times 10^{-13},\ 2.0 \times 10^{-13},\ 5.2 \times 10^{-13},\ 4.0 \times 10^{-13},\ 5.0 \times 10^{-13}, 1\ 2.2 \times 10^{-12},\ 2.0 \times 10^{-12},\ 8.4 \times 10^{-12}]$. The accuracy is quite high with very short training time, corroborating the potential of the DFEM being used in real-time analysis and data assimilation for digital twin.

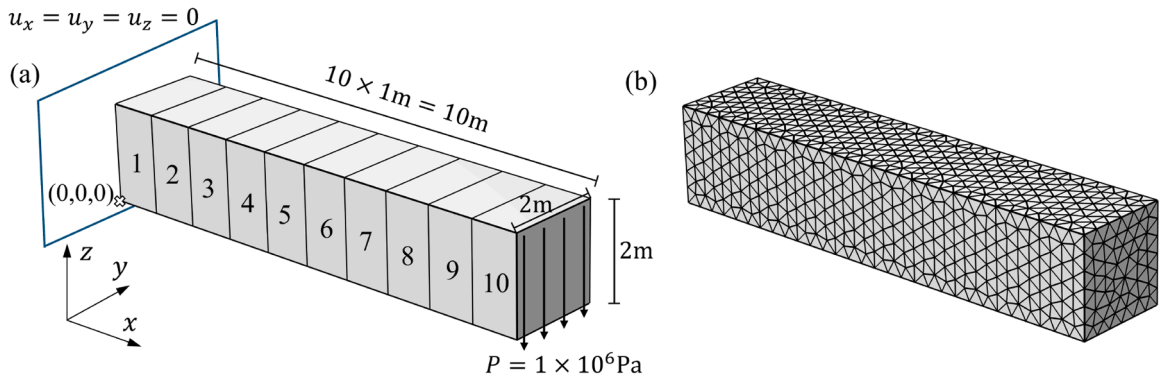


Fig. 11. A damaged heterogeneous cantilever beam with ten unknown materials: (a) Geometry and boundary conditions; (b) Mesh for FEM and the PENN (DFEM).

5.2. A damaged cantilever beam with twenty unknown materials

As shown in Fig. 14, the same initial setting as the former section is utilized, but with more refined representation of damage using twenty unknown materials and forty unknown parameters. Actual parameters of the deteriorated materials are $[\mathrm{E}_1 \sim \mathrm{E}_{20}] = [1.00,\ 1.05,\ 1.10,\ 1.15,\ 1.20,\ 1.25,\ 1.30,\ 1.35,\ 1.40,\ 1.45,\ 1.50,\ 1.55,\ 1.60,\ 1.65,\ 1.70,\ 1.75,\ 1.80,\ 1.85,\ 1.90,\ 1.95] \times 10^{10}$Pa, and $[\nu_1 \sim \nu_{20}] = [0.200,\ 0.205,\ 0.210,\ 0.215,\ 0.220,\ 0.225,\ 0.230,\ 0.235,\ 0.240,\ 0.245,\ 0.250,\ 0.255, 0.260,\ 0.265,\ 0.270,\ 0.275,\ 0.280,\ 0.285,\ 0.290,\ 0.295] \times 10^{10}$Pa. An LBFGS optimizer is utilized with step size 0.5 and "strong_wolfe" line search function. These forty material parameters are initialized from $E = 2 \times 10^{10}$Pa and $\nu = 0.15$. Fig. 15 shows that Young's modulus parameters $\mathrm{E}_1 \sim \mathrm{E}_{20}$ share the same trend. They initially decrease to a lower value than the real modulus, and then they converge to target values. Fig. 16 demonstrates that Poisson's ratios $\nu_1 \sim \nu_{20}$ also converge at the same time after some perturbations in the beginning. These forty unknown material parameters converge in just 251 iterations (the former case with twenty unknown material parameters converge in 117 iterations). This training process takes 2.8 s on a CPU (Intel i9–14900KF). Fig. 17 presents the displacements along three orthogonal axes. Since the DFEM strictly encodes labeled displacements from FEM, without the cost of training, the DFEM can maintain a very high accuracy. Relative errors of $\mathrm{E}_1 \sim \mathrm{E}_{20}$ are $[1.5 \times 10^{-11}, 1.5 \times 10^{-11}, 1.5 \times 10^{-11}, 1.5 \times 10^{-11}, 1.5 \times 10^{-11}, 1.5 \times 10^{-11}, 1.5 \times 10^{-11}, 1.4 \times 10^{-11}, 1.4 \times 10^{-11}, 1.4 \times 10^{-11}, 1.3 \times 10^{-11}, 1.2 \times 10^{-11}, 1.2 \times 10^{-11}, 1.1 \times 10^{-11}, 9 \times 10^{-12}, 8 \times 10^{-12}, 6 \times 10^{-12}, 4 \times 10^{-12}, 1 \times 10^{-12}, 1.5 \times 10^{-121}]$. Relative errors of $\nu_1 \sim \nu_{20}$ are $[3.6 \times 10^{-15}, 3.3 \times 10^{-14}, 1.2 \times 10^{-13}, 2.2 \times 10^{-13}, 1.5 \times 10^{-13}, 2.6 \times 10^{-13}, 3.7 \times 10^{-13}, 4.8 \times 10^{-13}, 3.7 \times 10^{-13}, 6.4 \times 10^{-13}, 3.3 \times 10^{-13}, 5.9 \times 10^{-13}, 1.1 \times 10^{-12}, 1.9 \times 10^{-13}, 2.4 \times 10^{-12}, 1.4 \times 10^{-12}, 1.3 \times 10^{-12}, 3.2 \times 10^{-12}, 2.3 \times 10^{-12}, 6.4 \times 10^{-12}]$. The same accuracy and efficiency as the former case corroborate DFEM's good scalable performance for inverse analysis of engineering cases.

5.3. The tunnel lining under unknown heterogeneous loadings

This section investigates the application of the DFEM for tunnel lining under unknown loadings with partially observed data. In Fig. 18a, half of the tunnel lining is studied in this section under the assumption of symmetry. Fig. 18b is the mesh used in COMSOL Multiphysics 6.2 and the DFEM. The inner radius is 4.0m, and the thickness is 0.5m. The initial Young's modulus and Poisson's ratio are $\mathrm{E} = 3 \times 10^{10}$Pa and $\nu = 0.2$. The longitudinal width is 4.0m. The boundary conditions are as follows:

$$\begin{aligned}
&u_x(x = 0.0,\ y, z) = 0 \quad && y \in (0.0,\ 4.0),\ z \in (0.0,\ 0.5) \cup (3.5,\ 4.0\ ) \\
&u_y(x = 0.0,\ y,\ z) = 0 && y \in (0.0,\ 4.0),\ z \in (0.0,\ 0.5) \\
&u_y(x,\ y = 0.0,\ z) = 0 && y, z \text{ on the lining} \\
&u_y(x,\ y = 4.0, z) = 0 && y, z \text{ on the lining} \\
&u_z(x = 0.0,\ y,\ z) = 0 && y \in (0.0,\ 4.0),\ z \in (0.0,\ 0.5)
\end{aligned} \tag{21}$$

We assume the tunnel lining structure is under heterogeneous loadings $P_1 = 1 \times 10^5$N and $P_2 = 8 \times 10^4$N like that in Fig. 18c. In underground space, only partial monitoring data can be obtained. Taking the 3D tunnel lining in Fig. 18 as an example, displacements on the inner boundary can be readily monitored. There are 1,111 nodes in total, we only have 342 nodes being monitored. Displacements of one-third of the nodes are encoded into the DFEM architecture. The rest of the displacement variables are initialized as zeros.

Since the loading acting on the outside boundary of the lining is hard to measure, P_1 and P_2 are set as unknowns. Loadings are obtained by inverse analysis with monitored displacements on the inner boundary. The LBFGS optimizer is utilized with step size 0.5

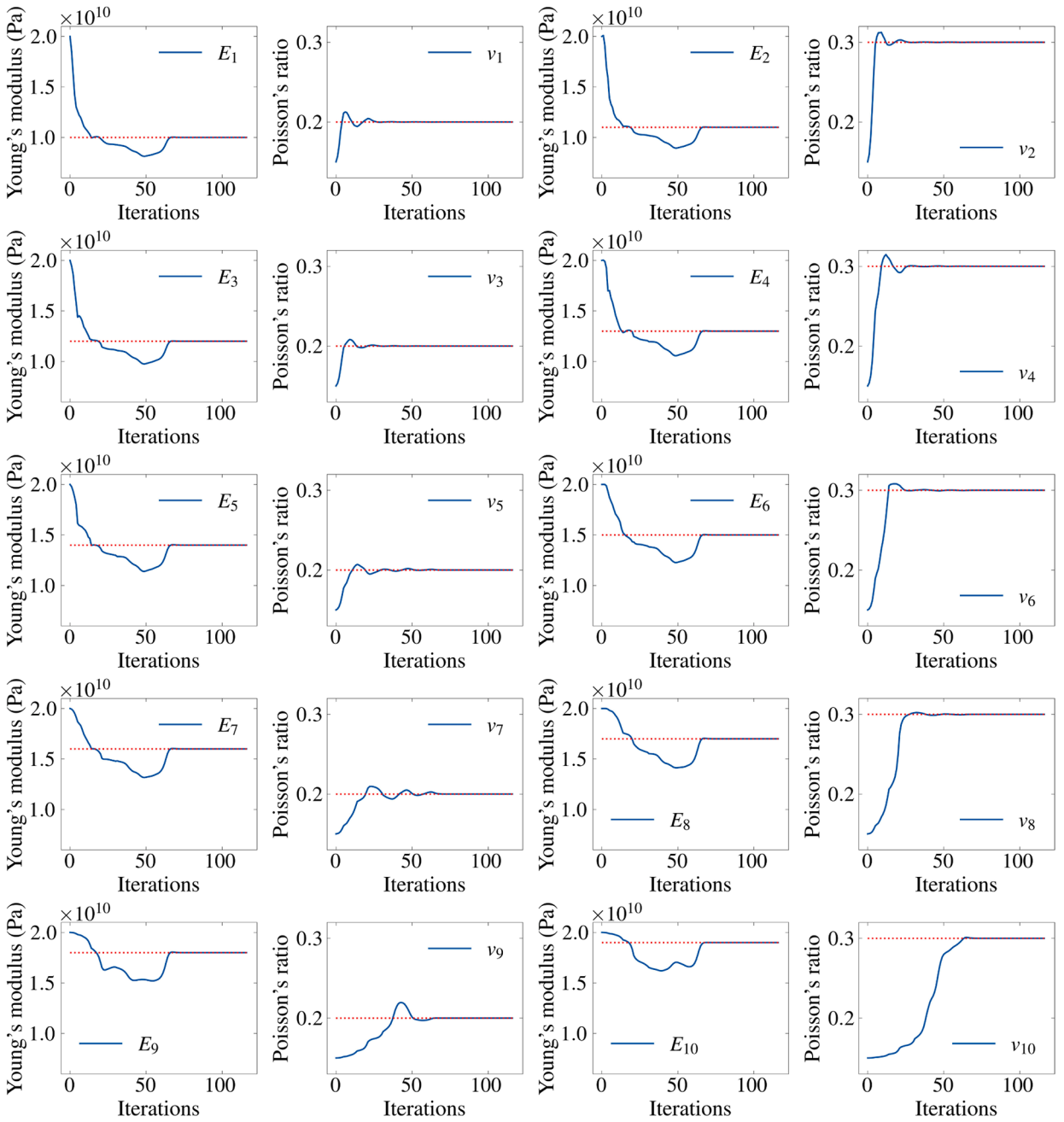


Fig. 12. Inverse analysis of ten unknown materials with twenty unknown parameters (Young's modulus E_i, $i = 1 \sim 10$; Poisson's ratio ν_i, $i = 1 \sim 10$) in the damaged heterogeneous cantilever beam (red dotted line denotes the real value).

and "strong_wolfe" line search function. The loss curve and the convergence curve of P_1 and P_2 are presented in Fig. 19. The training can be divided into three stages. In the first stage (about 0∼350 steps), the loss decreases rapidly and reaches a plateau. Unknown loadings are initialized as zeros. P_1 and P_2 do not change much in the first stage. The reason for loss reduction in this stage is probably the adjustment of displacement variables conforming to the encoded labeled displacements. The second stage is about 350∼700 steps. Estimates of P_1 and P_2 start to change, corresponding to the decrease of loss. A stabilized stage also follows, during which the unknown displacements should also undergo a period of adjustment. The final stage (700 steps∼end) reaches the accurate estimate of loadings, as well as a quite small loss value. In the end, as shown in Fig. 20, the L^2 relative error of u_x is 2.1×10^{-9}, and L^2 relative error of u_y is 7.4×10^{-7}. The relative error of P_1 is 2.8×10^{-8}, and the relative error of P_2 is 2.8×10^{-8}. The DFEM converges in 1,079 steps with 13 s on a CPU (Intel i9–14900KF), proving its potential for real time inverse analysis and data assimilations for digital twin in engineering.

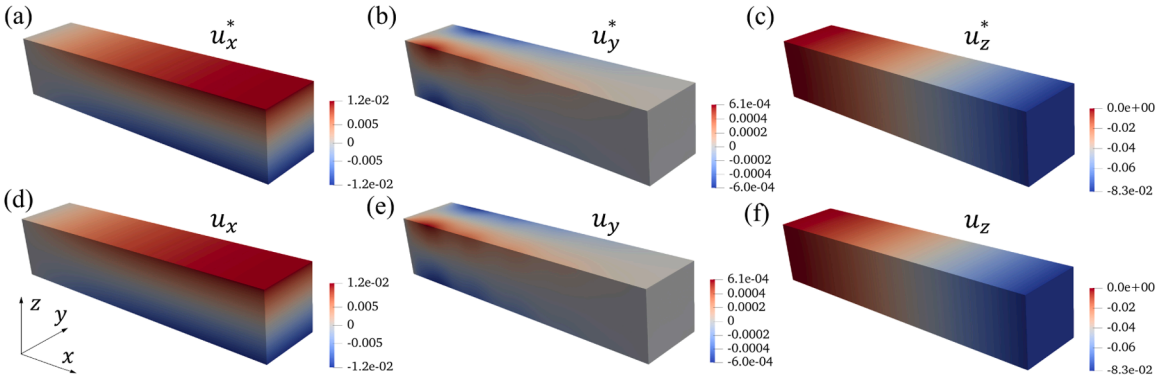


Fig. 13. Displacement components of the damaged heterogeneous cantilever with ten unknown materials: FEM reference displacements (a) along x axis, (b) along y axis, (c) along z axis; the PENN (DFEM) displacements (d) along x axis, (e) along y axis, (f) along z axis.

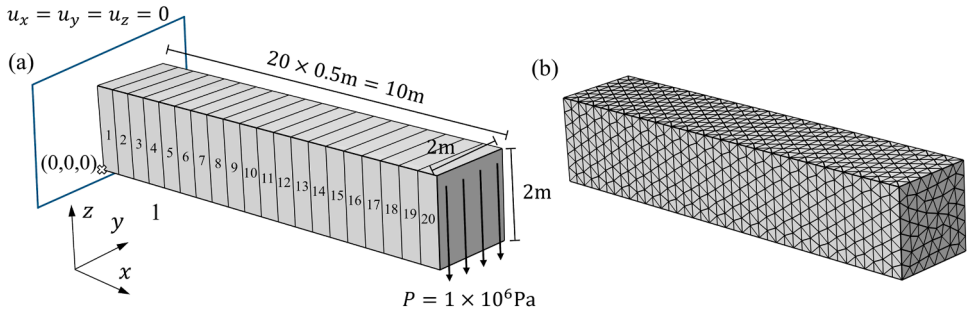


Fig. 14. A damaged heterogeneous cantilever beam with twenty unknown materials: (a) Geometry and boundary conditions; (b) Mesh for FEM and the PENN (DFEM).

5.4. Discussions about the influence of noisy data

The previous section studying tunnel lining under unknown loadings with sparse labeled displacements on the inner boundary is employed here to evaluate the performance of the DFEM during inverse analysis utilizing noisy data. A random noise sampled from the standard Gaussian distribution with zero mean and unit deviation is amplified for the expected noise level. The corrupted labeled data is also directly encoded to the DFEM's interpretable structure. L_2 relative errors of the labeled displacements match the relative noise magnitude. The error bar is plotted in Fig. 21. Each noise level is sampled ten times for a fair quantification. The data noise can lead to a large standard deviation in the inverse analysis of P_1 and P_2. For example, relative magnitude of 0.05 can lead to 0.4 relative error of P_1. It can be explained by the precise absorption of labeled data in the DFEM. The DFEM is sensitive to high frequency patterns, and thus it can efficiently and accurately deal with heterogeneous materials and complex cases with fine or concentration features. However, the noise is also a kind of local fluctuation with high-frequency patterns. The DFEM is like a high-resolution device that is sensitive to noises. Effective denoising methods can be leveraged for preprocessing. For example, a very simple way is to averaging monitoring data from several observations. The mean value of P_1 under 0.05 relative magnitude of noise only deviates from real value by a relative error of 0.01.

6. Conclusions

This study proposes the novel DFEM, a kind of Physics-Encoded Numerical Network (PENN) based on Galerkin discretization for diverse inverse analyses of multidimensional heterogeneous engineering structures. PINNs use neural networks with blackbox weights and biases to learn physics and data in a soft way by incorporating PDE residuals, the loss of boundary conditions, the loss of initial conditions, and the loss of data into the loss function, while multiple loss terms can impede convergence. With the DFEM, the weak-form physics discretized with the Galerkin method is encoded into the interpretable differentiable network. The DFEM eliminates all blackbox trainable parameters that are required in classical neural networks, retaining only those with clear and interpretable meanings (nodal displacements, material parameters, and boundary conditions). The interpretability enables the DFEM to accurately encode labeled data without additional training. These merits contribute to the DFEM's superior performance.

Two benchmark experiments are conducted to validate the DFEM's superior efficiency and accuracy. In the thick cylinder case to back calculate the internal pressure with inverse analysis, the conventional PINN failed. Compared with improved PINNs, the DFEM requires only 0.3 % of the training iterations to achieve an accuracy three orders of magnitude higher. In the test of indentation by a

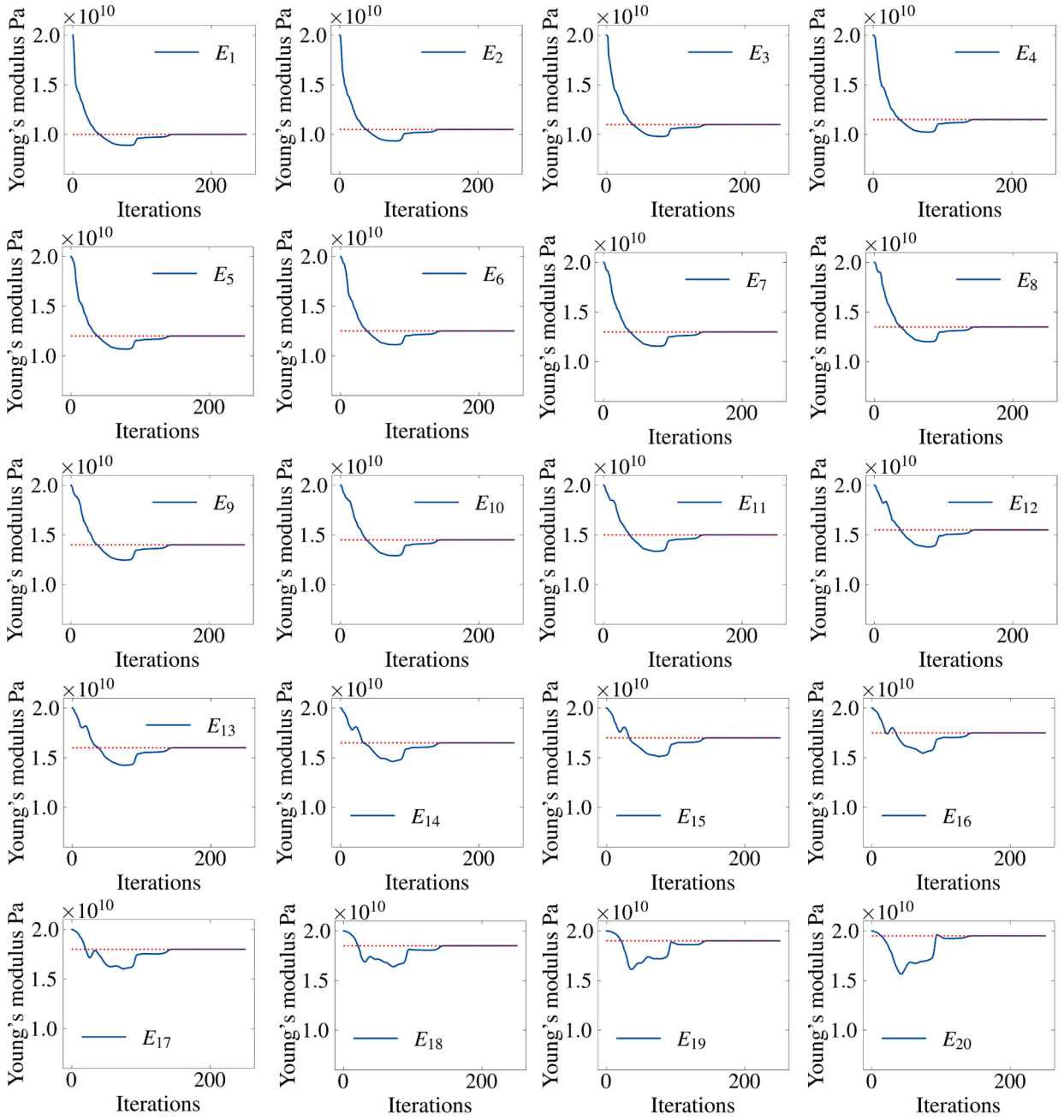


Fig. 15. Inverse analysis of twenty unknown materials (Young's modulus E_i, $i = 1 \sim 20$) in the damaged heterogeneous cantilever beam (red dotted line denotes the real value).

rigid punch, the accuracy of material parameters is not as expected using the conventional PINN because of stress concentration features. With a training time five orders of magnitude faster than the conventional PINN, the DFEM is also validated to be five orders of magnitude more accurate in determining unknown material parameters.

Moreover, the DFEM proves effective for 3D engineering structures. A damaged cantilever beam, characterized by ten heterogeneous materials with twenty unknown parameters, is efficiently solved in just 1.5 s. A tunnel lining ring with sparse labeled data under unknown heterogeneous boundary loadings is also readily cracked with very low errors and high efficiency in 13 s. The performance of the DFEM under noisy data is also discussed. The average results are robust and accurate under high noise levels. The vanilla Galerkin loss of Gao et al. [36] was realized with black box graph neural network. The DFEM eliminates all black box parameters, retaining only those with clear interpretable meanings. To the best of the authors' knowledge, it has never been done using a vanilla Galerkin loss. Comparisons with improved PINNs fully demonstrate the improved efficiency and accuracy of DFEM, which are several orders of

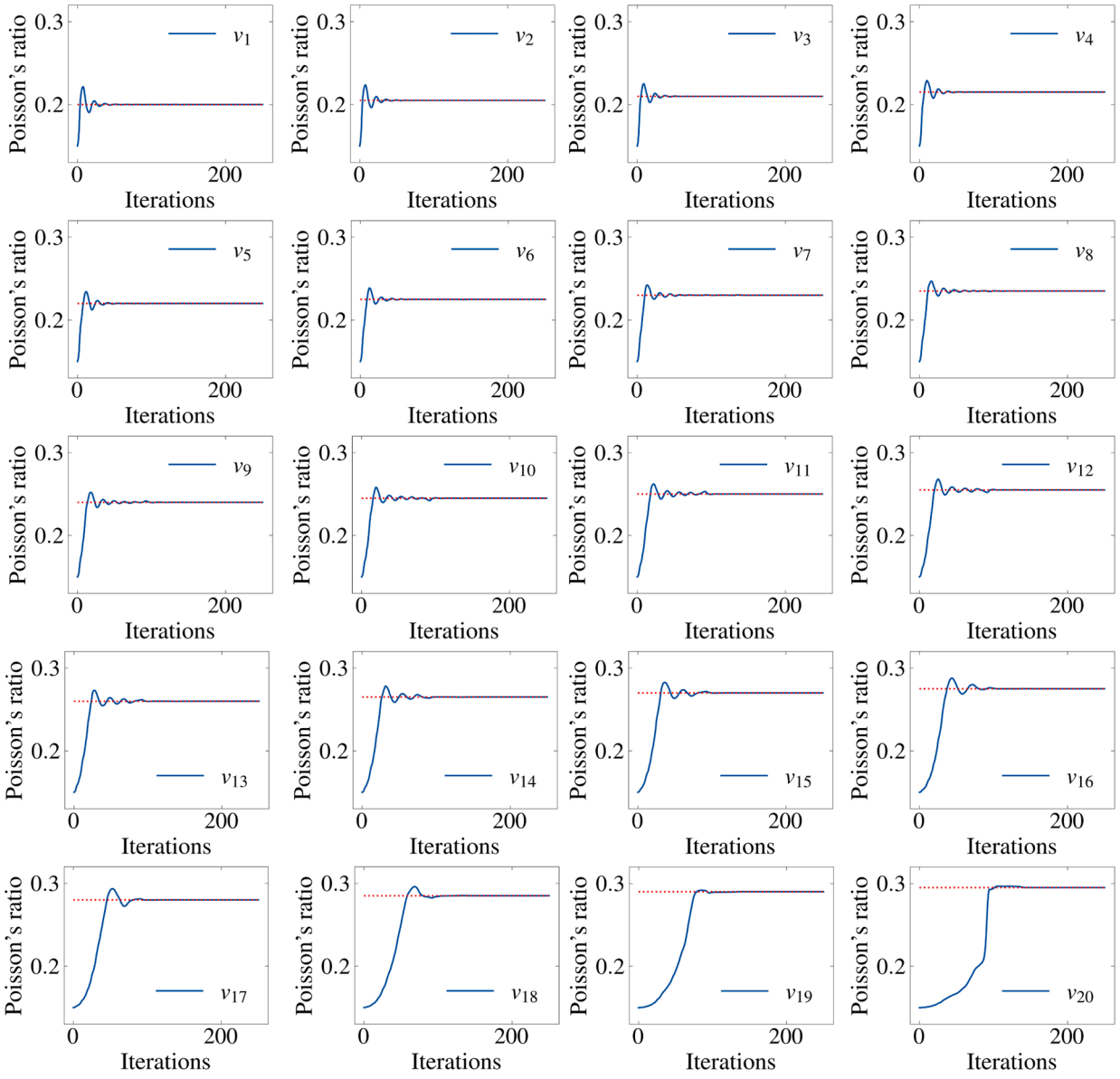


Fig. 16. Inverse analysis of twenty unknown materials (Poisson's ratio v_i, $i = 1 \sim 20$) in the damaged heterogeneous cantilever beam (red dotted line denotes the real value).

magnitude faster and more accurate. Inverse analysis of multidimensional heterogeneous structures is accurately solved in seconds. To the best of the authors' knowledge, such performance has never been achieved using classical PINNs or vanilla Galerkin loss. DFEM provides an innovative and practical way to bridge PINN and classical numerical methods. Similar to PINN, DFEM leverages physics and automatic differentiation, serving as a general and flexible framework to seamlessly combine physics and data. In addition, DFEM is fully compatible with classical numerical methods.

This study significantly expands the application scope of PINN-like methods. From mostly 1D/2D simplified homogeneous inverse analysis in engineering scenarios with conventional PINNs to 3D heterogeneous inverse analysis for engineering structures using the DFEM, we demonstrate its substantial speed improvements and high-fidelity results, corroborating the DFEM's potential for digital twin in civil and mechanical engineering.

For future research, more complex constitutive models like path-dependent plastic ones should be implemented by sampling enough steps along the loading path. A multistep update can be leveraged to capture the dependency of stress evolution path. Other physical phenomena like fluid flow and heat conduction can be addressed by following the similar derivation in our methodology. Multi-physics problems can be more difficult, and a straightforward solution is to use multistep sequential update scheme or coupled update scheme. Moreover, our Differentiable Finite Element Method (DFEM) can be easily generalized into Physics-Encoded

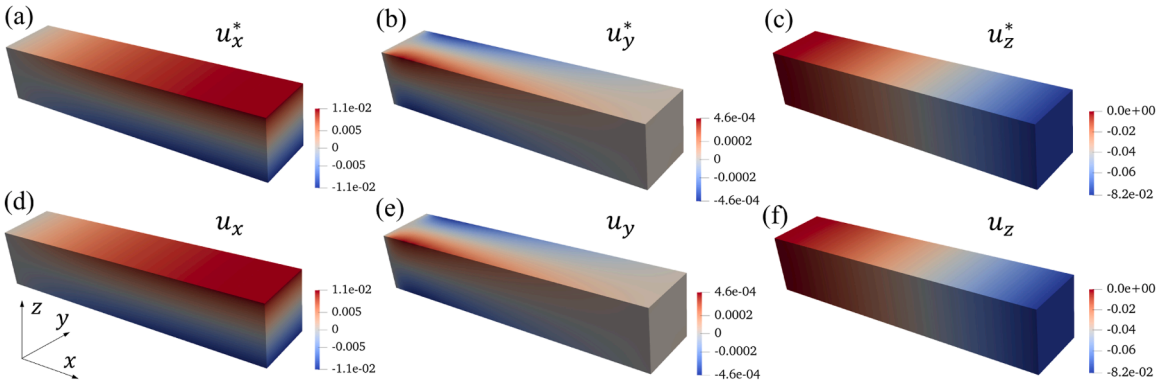


Fig. 17. Displacement components of the damaged heterogeneous cantilever with twenty unknown materials: FEM reference displacements (a) along x axis, (b) along y axis, (c) along z axis; the PENN (DFEM) displacements (d) along x axis, (e) along y axis, (f) along z axis.

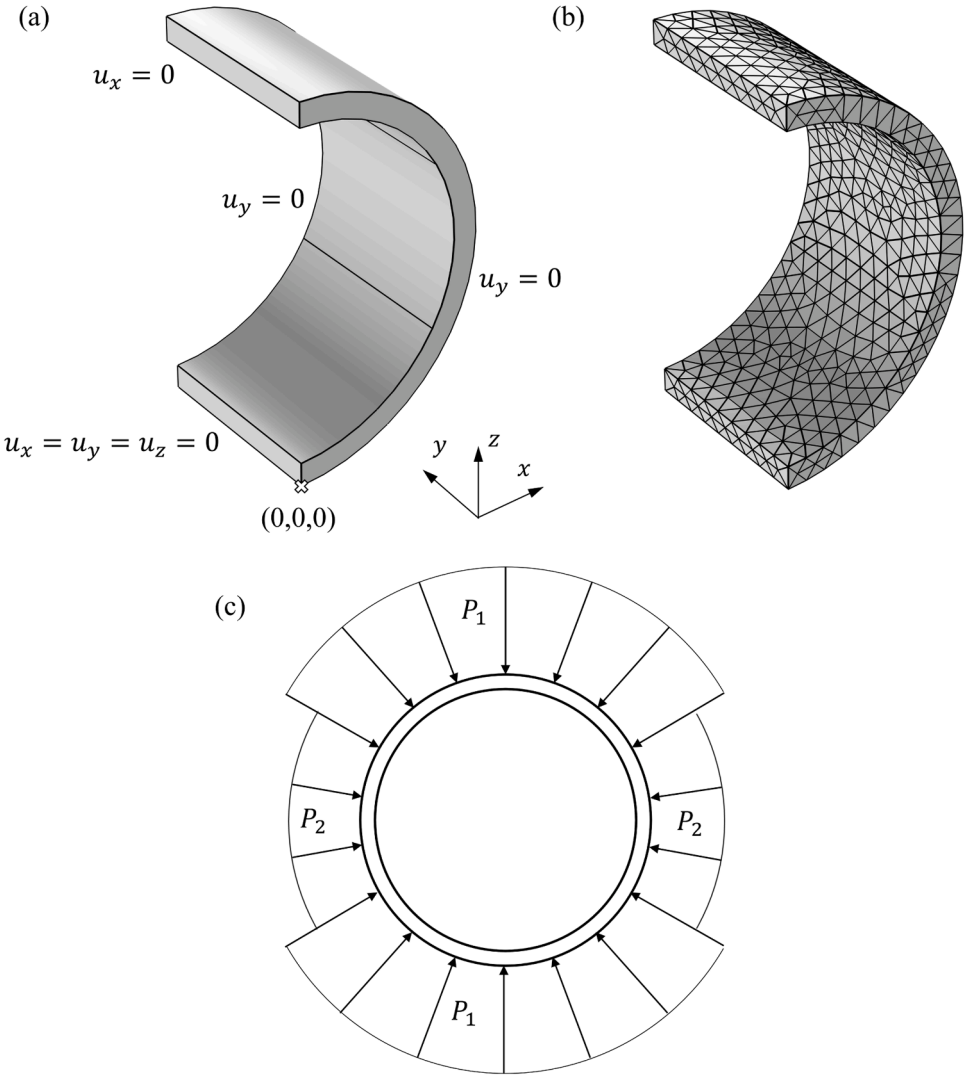


Fig. 18. Tunnel lining under unknown heterogeneous loadings: (a) Geometry and boundary conditions; (b) Mesh for FEM and the PENN (DFEM); (c) Loading patterns on the lining.

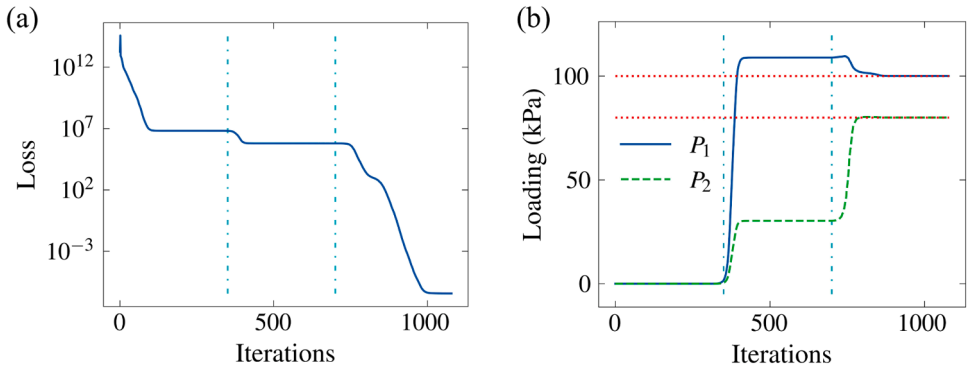


Fig. 19. Convergence using the PENN (DFEM) of the tunnel lining problem under heterogeneous loadings: (a) Loss and (b) Loadings with respect to training iterations.

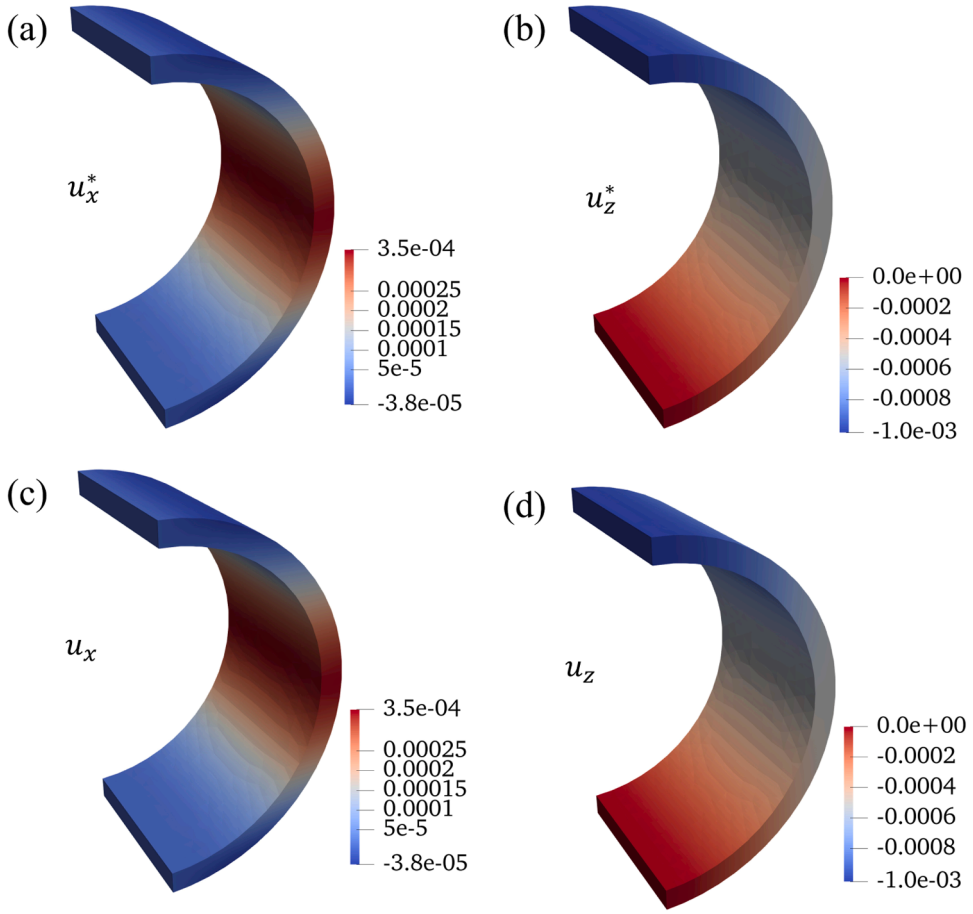


Fig. 20. Displacement components of the tunnel ring: FEM reference displacements (a) along x axis; (b) along y axis; the PENN (DFEM) displacements (c) along x axis; (d) along y axis.

Numerical Network (PENN) to revitalize classical numerical methods towards AI4Science.

CRediT authorship contribution statement

Xi Wang: Writing – review & editing, Writing – original draft, Software, Methodology, Conceptualization. **Zhen-Yu Yin:** Writing – review & editing, Project administration, Funding acquisition, Conceptualization. **Wei Wu:** Writing – review & editing, Project administration, Funding acquisition. **He-Hua Zhu:** Project administration, Funding acquisition.

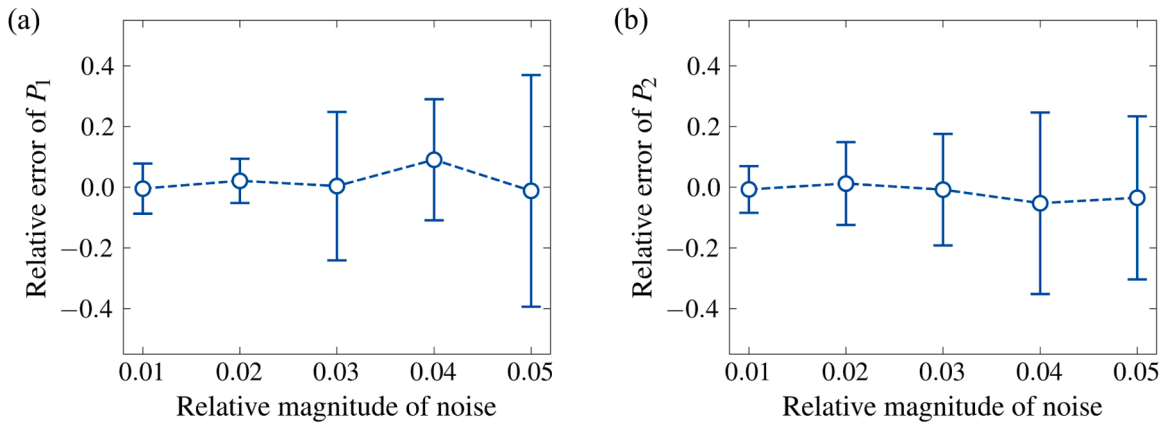


Fig. 21. Relative error of loadings with respect to the magnitude of noise: (a) Relative error of P_1; (b) Relative error of P_2.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by Hong Kong Polytechnic University; Research Grants Council (RGC) of the Hong Kong Special Administrative Region Government (HKSARG) of China (grant no 15220221; 15227923, 15229223); National Natural Science Foundation of China [42272338].

Data availability

Data will be made available on request.

References

[1] S.S. Rao, The Finite Element Method in Engineering, Butterworth-Heinemann, 2017.
[2] Q. Zhang, Z.Y. Wang, Z.Y. Yin, Y.F. Jin, A novel stabilized NS-FEM formulation for anisotropic double porosity media, Comput. Methods Appl. Mech. Eng. 401 (2022) 115666.
[3] C. Yang, F. Zhu, J. Zhao, A multi-horizon fully coupled thermo-mechanical peridynamics, J. Mech. Phys. Solid. (2024) 105758, https://doi.org/10.1016/j.jmps.2024.105758.
[4] C. Yang, F. Zhu, J. Zhao, Coupled total- and semi-Lagrangian peridynamics for modelling fluid-driven fracturing in solids, Comput. Meth. Appl. Mech. Eng. 419 (2024) 116580, https://doi.org/10.1016/j.cma.2023.116580.
[5] W. Liang, H. Fang, Z.Y. Yin, J. Zhao, A mortar segment-to-segment frictional contact approach in material point method, Comput. Methods Appl. Mech. Eng. 431 (2024) 117294.
[6] J. Yu, J. Zhao, S. Zhao, W. Liang, Thermo-hydro-mechanical coupled material point method for modeling freezing and thawing of porous media, Int. J. Numeric. Analyt. Methods Geomech. (2024).
[7] X. Wang, W. Wu, H. Zhu, H. Zhang, Three-dimensional discontinuous deformation analysis derived from the virtual work principle with a simplex integral on the boundary, Comput. Geotech. 146 (2022) 104710, https://doi.org/10.1016/j.compgeo.2022.104710.
[8] X. Wang, W. Wu, H. Zhu, H. Zhang, J.-S. Lin, A. Bobet, A global direct search method for high-fidelity contact detection between arbitrarily shaped three-dimensional convex polyhedral blocks, Comput. Geotech. 150 (2022) 104891, https://doi.org/10.1016/j.compgeo.2022.104891.
[9] X. Wang, W. Wu, H. Zhu, H. Zhang, J.-S. Lin, The last entrance plane method for contact indeterminacy between convex polyhedral blocks, Comput. Geotechn. 117 (2020) 103283, https://doi.org/10.1016/j.compgeo.2019.103283.
[10] J. Zhao, S. Zhao, S. Luding, The role of particle shape in computational modelling of granular matter, Nat. Rev. Phys. (2023) 1–21, https://doi.org/10.1038/s42254-023-00617-9.
[11] M. Wu, J. Wang, B. Pan, Z.Y. Yin, Particle tracking-aided digital volume correlation for clay–sand soil mixtures, Géotechnique (2024) 1–13.
[12] Y. Kong, X. Li, J. Zhao, M. Guan, Load–deflection of flexible ring-net barrier in resisting debris flows, Géotechnique 74 (5) (2022) 486–498.
[13] Y. Kong, M. Guan, Hydro-mechanical simulations aid demand-oriented design of slit dams for controlling debris flows, debris avalanches and rock avalanches, Eng. Geol. 326 (2023) 107314.
[14] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neur. Netw. 2 (1989) 359–366, https://doi.org/10.1016/0893-6080(89)90020-8.
[15] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707, https://doi.org/10.1016/j.jcp.2018.10.045.
[16] G.E. Karniadakis, I.G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, L. Yang, Physics-informed machine learning, Nat. Rev. Phys. 3 (2021) 422–440, https://doi.org/10.1038/s42254-021-00314-5.
[17] F. Lehmann, M. Fahs, A. Alhubail, H. Hoteit, A mixed pressure-velocity formulation to model flow in heterogeneous porous media with physics-informed neural networks, Adv. Water Resour. 181 (2023) 104564, https://doi.org/10.1016/j.advwatres.2023.104564.
[18] H. Guo, Z.-Y. Yin, A novel physics-informed deep learning strategy with local time-updating discrete scheme for multi-dimensional forward and inverse consolidation problems, Comput. Method. Appl. Mech. Eng. 421 (2024) 116819, https://doi.org/10.1016/j.cma.2024.116819.

[19] T. Qu, J. Zhao, S. Guan, Y. Feng, Data-driven multiscale modelling of granular materials via knowledge transfer and sharing, Int. J. Plast. 171 (2023) 103786, https://doi.org/10.1016/j.ijplas.2023.103786.
[20] T. Qu, S. Guan, Y.T. Feng, G. Ma, W. Zhou, J. Zhao, Deep active learning for constitutive modelling of granular materials: From representative volume elements to implicit finite element modelling, Int. J. Plastic. 164 (2023) 103576.
[21] P. Borate, J. Rivière, C. Marone, A. Mali, D. Kifer, P. Shokouhi, Using a physics-informed neural network and fault zone acoustic monitoring to predict lab earthquakes, Nat. Commun. 14 (2023) 3693, https://doi.org/10.1038/s41467-023-39377-6.
[22] S. Goswami, C. Anitescu, S. Chakraborty, T. Rabczuk, Transfer learning enhanced physics informed neural network for phase-field modeling of fracture, Theoret. Appl. Fract. Mech. 106 (2020) 102447, https://doi.org/10.1016/j.tafmec.2019.102447.
[23] X.-X. Chen, P. Zhang, Z.-Y. Yin, Physics-Informed neural network solver for numerical analysis in geoengineering, Georisk 0 (2024) 1–19, https://doi.org/10.1080/17499518.2024.2315301.
[24] K.Q. Li, H.L. He, Towards an improved prediction of soil-freezing characteristic curve based on extreme gradient boosting model, Geosci. Front. 15 (6) (2024) 101898.
[25] Y. Li, Z. Zhao, L. Sun, T. Guo, B. Yang, A geometrical morphology-enhanced computer vision approach for structural health assessment, Struct. Health Monitor. (2025), 14759217241307575.
[26] E. Haghighat, M. Raissi, A. Moure, H. Gomez, R. Juanes, A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics, Comput. Meth. Appl. Mech. Eng. 379 (2021) 113741, https://doi.org/10.1016/j.cma.2021.113741.
[27] D.W. Abueidda, Q. Lu, S. Koric, Meshless physics-informed deep learning method for three-dimensional solid mechanics, Int. J. Numer. Method. Eng. 122 (2021) 7182–7201, https://doi.org/10.1002/nme.6828.
[28] M. Vahab, E. Haghighat, M. Khaleghi, N. Khalili, A physics-informed neural network approach to solution and identification of biharmonic equations of elasticity, J. Eng. Mech. 148 (2022) 04021154, https://doi.org/10.1061/(ASCE)EM.1943-7889.0002062.
[29] A.M. Roy, R. Bose, V. Sundararaghavan, R. Arróyave, Deep learning-accelerated computational framework based on Physics Informed Neural Network for the solution of linear elasticity, Neur. Netw. 162 (2023) 472–489, https://doi.org/10.1016/j.neunet.2023.03.014.
[30] V.M. Nguyen-Thanh, X. Zhuang, T. Rabczuk, A deep energy method for finite deformation hyperelasticity, Eur. J. Mech. - A/Solid. 80 (2020) 103874, https://doi.org/10.1016/j.euromechsol.2019.103874.
[31] W. E, B. Yu, The deep ritz method: a deep learning-based numerical algorithm for solving variational problems, Commun. Math. Stat. 6 (2018) 1–12, https://doi.org/10.1007/s40304-018-0127-z.
[32] W. Li, M.Z. Bazant, J. Zhu, A physics-guided neural network framework for elastic plates: comparison of governing equations-based and energy-based approaches, Comput. Method. Appl. Mech. Eng. 383 (2021) 113933, https://doi.org/10.1016/j.cma.2021.113933.
[33] V.M. Nguyen-Thanh, C. Anitescu, N. Alajlan, T. Rabczuk, X. Zhuang, Parametric deep energy approach for elasticity accounting for strain gradient effects, Comput. Method. Appl. Mech. Eng. 386 (2021) 114096, https://doi.org/10.1016/j.cma.2021.114096.
[34] J. Wang, Y.L. Mo, B. Izzuddin, C.-W. Kim, Exact Dirichlet boundary physics-informed neural network EPINN for solid mechanics, Comput. Method. Appl. Mech. Eng. 414 (2023) 116184, https://doi.org/10.1016/j.cma.2023.116184.
[35] X. Wang, Z.-Y. Yin, W. Wu, H.-H. Zhu, Neural network-augmented differentiable finite element method for boundary value problems, Int. J. Mech. Sci. 285 (2025) 109783, https://doi.org/10.1016/j.ijmecsci.2024.109783.
[36] H. Gao, M.J. Zahr, J.-X. Wang, Physics-informed graph neural Galerkin networks: a unified framework for solving PDE-governed forward and inverse problems, Comput. Meth. Appl. Mech. Eng. 390 (2022) 114502, https://doi.org/10.1016/j.cma.2021.114502.
[37] S. Motiwale, W. Zhang, R. Feldmeier, M.S. Sacks, A neural network finite element approach for high speed cardiac mechanics simulations, Comput. Meth. Appl. Mech. Eng. 427 (2024) 117060, https://doi.org/10.1016/j.cma.2024.117060.
[38] X.-L. Yu, X.-P. Zhou, A nonlocal energy-informed neural network based on peridynamics for elastic solids with discontinuities, Comput. Mech. 73 (2024) 233–255, https://doi.org/10.1007/s00466-023-02365-0.
[39] X.-L. Yu, X.-P. Zhou, A nonlocal energy-informed neural network for peridynamic correspondence material models, Eng. Anal. Bound. Elem. 160 (2024) 273–297, https://doi.org/10.1016/j.enganabound.2024.01.004.
[40] X.-L. Yu, X.-P. Zhou, A nonlocal energy-informed neural network for isotropic elastic solids with cracks under thermomechanical loads, Int. J. Numer. Method. Eng. 124 (2023) 3935–3963, https://doi.org/10.1002/nme.7296.
[41] X.-P. Zhou, X.-L. Yu, Transfer learning enhanced nonlocal energy-informed neural network for quasi-static fracture in rock-like materials, Comput. Method. Appl. Mech. Eng. 430 (2024) 117226, https://doi.org/10.1016/j.cma.2024.117226.
[42] B. Feng, X.-P. Zhou, The novel graph transformer-based surrogate model for learning physical systems, Comput. Method. Appl. Mech. Eng. 432 (2024) 117410, https://doi.org/10.1016/j.cma.2024.117410.
[43] E. Haghighat, A.C. Bekar, E. Madenci, R. Juanes, A nonlocal physics-informed deep learning framework using the peridynamic differential operator, Comput. Meth. Appl. Mech. Eng. 385 (2021) 114012, https://doi.org/10.1016/j.cma.2021.114012.
[44] J.N. Fuhg, N. Bouklas, The mixed Deep Energy Method for resolving concentration features in finite strain hyperelasticity, J. Comput. Phys. 451 (2022) 110839, https://doi.org/10.1016/j.jcp.2021.110839.
[45] D.W. Abueidda, S. Koric, E. Guleryuz, N.A. Sobh, Enhanced physics-informed neural networks for hyperelasticity, Int. J. Numer. Method. Eng. 124 (2023) 1585–1601, https://doi.org/10.1002/nme.7176.
[46] C. Chadha, J. He, D. Abueidda, S. Koric, E. Guleryuz, I. Jasiuk, Improving the accuracy of the deep energy method, Acta Mech. 234 (2023) 5975–5998, https://doi.org/10.1007/s00707-023-03691-3.
[47] I. Jeong, M. Cho, H. Chung, D.-N. Kim, Data-driven nonparametric identification of material behavior based on physics-informed neural network with full-field data, Comput. Method. Appl. Mech. Eng. 418 (2024) 116569, https://doi.org/10.1016/j.cma.2023.116569.
[48] Y. Wang, J. Sun, W. Li, Z. Lu, Y. Liu, CENN: conservative energy method based on neural networks with subdomains for solving variational problems involving heterogeneous and complex geometries, Comput. Method. Appl. Mech. Eng. 400 (2022) 115491, https://doi.org/10.1016/j.cma.2022.115491.
[49] Y. Diao, J. Yang, Y. Zhang, D. Zhang, Y. Du, Solving multi-material problems in solid mechanics using physics-informed neural networks based on domain decomposition technology, Comput. Method. Appl. Mech. Eng. 413 (2023) 116120, https://doi.org/10.1016/j.cma.2023.116120.
[50] E. Zhang, M. Dao, G.E. Karniadakis, S. Suresh, Analyses of internal structures and defects in materials using physics-informed neural networks, Sci. Adv. 8 (2022) eabk0644, https://doi.org/10.1126/sciadv.abk0644.
[51] S. Rezaei, A. Harandi, A. Moeineddin, B.-X. Xu, S. Reese, A mixed formulation for physics-informed neural networks as a potential solver for engineering problems in heterogeneous domains: comparison with finite element method, Comput. Method. Appl. Mech. Eng. 401 (2022) 115616, https://doi.org/10.1016/j.cma.2022.115616.
[52] A. Harandi, A. Moeineddin, M. Kaliske, S. Reese, S. Rezaei, Mixed formulation of physics-informed neural networks for thermo-mechanically coupled systems and heterogeneous domains, Num. Method. Eng. (2023) e7388, https://doi.org/10.1002/nme.7388.
[53] X. Ren, X. Lyu, Mixed form based physics-informed neural networks for performance evaluation of two-phase random materials, Eng. Appl. Artif. Intell. 127 (2024) 107250, https://doi.org/10.1016/j.engappai.2023.107250.
[54] W. Ouyang, G. Li, L. Chen, S.-W. Liu, Physics-informed neural networks for large deflection analysis of slender piles incorporating non-differentiable soil-structure interaction, Int. J. Numer. Anal. Method. Geomech. 48 (2024) 1278–1308, https://doi.org/10.1002/nag.3679.
[55] M. Vahab, B. Shahbodagh, E. Haghighat, N. Khalili, Application of physics-informed neural networks for forward and inverse analysis of pile–soil interaction, Int. J. Solid. Struct. 277–278 (2023) 112319, https://doi.org/10.1016/j.ijsolstr.2023.112319.
[56] Z. Zhang, Q. Pan, Z. Yang, X. Yang, Physics-informed deep learning method for predicting tunnelling-induced ground deformations, Acta Geotech. 18 (2023) 4957–4972, https://doi.org/10.1007/s11440-023-01874-9.
[57] S.-Y. He, C. Tang, W.-H. Zhou, Settlement prediction of immersed tunnel considering time-dependent foundation modulus, Tunnell. Undergr. Space Technol. 144 (2024) 105562, https://doi.org/10.1016/j.tust.2023.105562.

[58] S.-Y. He, W.-H. Zhou, C. Tang, Physics-informed neural networks for settlement analysis of the immersed tunnel of the Hong Kong–Zhuhai–Macau Bridge, Int. J. Geomech. 24 (2024) 04023241, https://doi.org/10.1061/IJGNAI.GMENG-8689.
[59] W. Ouyang, G. Li, L. Chen, S.-W. Liu, Machine learning-based soil–structure interaction analysis of laterally loaded piles through physics-informed neural networks, Acta Geotech. (2024), https://doi.org/10.1007/s11440-023-02179-7.
[60] C. Xu, B.T. Cao, Y. Yuan, G. Meschke, Transfer learning based physics-informed neural networks for solving inverse problems in engineering structures under different loading scenarios, Comput. Method. Appl. Mech. Eng. 405 (2023) 115852, https://doi.org/10.1016/j.cma.2022.115852.
[61] G. Wang, Q. Fang, J. Wang, Q.M. Li, J.Y. Chen, Y. Liu, Estimation of load for tunnel lining in elastic soil using physics-informed neural network, Comput.-Aid. Civil Infrastruct. Eng. (2024), https://doi.org/10.1111/mice.13208 n/a.
[62] L.D. McClenny, U.M. Braga-Neto, Self-adaptive physics-informed neural networks, J. Comput. Phys. 474 (2023) 111722, https://doi.org/10.1016/j.jcp.2022.111722.
[63] M.A. Nabian, R.J. Gladstone, H. Meidani, Efficient training of physics-informed neural networks via importance sampling, Comput.-Aid. Civ. Infrastruct. Eng. 36 (2021) 962–977, https://doi.org/10.1111/mice.12685.
[64] R. Cipolla, Y. Gal, A. Kendall, Multi-task learning using uncertainty to weigh losses for scene geometry and semantics, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 7482–7491, https://doi.org/10.1109/CVPR.2018.00781.
[65] A. Arzani, L. Yuan, P. Newell, B. Wang, Interpreting and generalizing deep learning in physics-based problems with functional linear models, Eng. Comput. (2024), https://doi.org/10.1007/s00366-024-01987-z.
[66] S. Wang, Y. Teng, P. Perdikaris, Understanding and mitigating gradient flow pathologies in physics-informed neural networks, SIAM J. Sci. Comput. 43 (2021) A3055–A3081, https://doi.org/10.1137/20M1318043.
[67] C. Rao, P. Ren, Q. Wang, O. Buyukozturk, H. Sun, Y. Liu, Encoding physics to learn reaction–diffusion processes, Nat. Mach. Intell. 5 (2023) 765–779, https://doi.org/10.1038/s42256-023-00685-7.
[68] Y. Lu, A. Zhong, Q. Li, B. Dong, Beyond finite layer neural networks: bridging deep architectures and numerical differential equations, (2020). https://doi.org/10.48550/arXiv.1710.10121.
[69] L. Ruthotto, E. Haber, Deep neural networks motivated by partial differential equations, J. Math. Imaging Vis. 62 (2020) 352–364, https://doi.org/10.1007/s10851-019-00903-1.
[70] P. Karnakov, S. Litvinov, P. Koumoutsakos, Solving inverse problems in physics by optimizing a discrete loss: fast and accurate learning without neural networks, PNAS Nexus 3 (2024) 005, https://doi.org/10.1093/pnasnexus/pgae005.
[71] X. Wang, Z.-Y. Yin, Interpretable physics-encoded finite element network to handle concentration features and multi-material heterogeneity in hyperelasticity, Comput. Meth. Appl. Mech. Eng. 431 (2024) 117268, https://doi.org/10.1016/j.cma.2024.117268.
[72] N. Rahaman, A. Baratin, D. Arpit, F. Draxler, M. Lin, F. Hamprecht, Y. Bengio, A. Courville, On the spectral bias of neural networks, in: Proceedings of the 36th International Conference on Machine Learning, PMLR, 2019, pp. 5301–5310, in: https://proceedings.mlr.press/v97/rahaman19a.html (accessed May 2, 2024).
[73] K.-J. Bathe, Finite Element Procedures, Prestice Hall, 1996.
[74] P. Rathore, W. Lei, Z. Frangella, L. Lu, M. Udell, Challenges in training PINNs: a loss landscape perspective, (2024). https://doi.org/10.48550/arXiv.2402.01868.