J. Mech. Phys. Solids 172 (2023) 105177

Contents lists available at ScienceDirect

Journal of the Mechanics and Physics of Solids

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

Modeling finite-strain plasticity using physics-informed neural network and assessment of the network performance

Sijun Niu [a], Enrui Zhang [b], Yuri Bazilevs [a], Vikas Srivastava [a,c,*]

[a] School of Engineering, Brown University, Providence, RI 02912, USA
[b] Division of Applied Mathematics, Brown University, Providence, RI 02912, USA
[c] Center for Biomedical Engineering, Brown University, Providence, RI 02912, USA

ARTICLE INFO

Keywords:
Physics-informed neural networks (PINN)
Machine learning
Finite element method (FEM)
Finite-deformation
Plasticity

ABSTRACT

Physics-informed neural networks (PINN) can solve partial differential equations (PDEs) by encoding the mathematical information explicitly into the loss functions. In the context of plasticity, discussions of PINN have only focused on small-strain formulations. We present a framework of finite-strain elasto-plasticity for PINN, considering rate-independent isotropic hardening in this work. Details of the model architecture are discussed, including inputs and outputs of the neural network and the construction of the loss function that incorporates equilibrium equations, boundary conditions and constitutive relations. In addition, the overall architecture can learn the constitutive relations from discrete measurements on a stress–strain curve, hence eliminating the need for modeling hardening law in the formulation. We demonstrate the performance of PINN through a numerical example that includes a multi-step loading and unloading history. Moreover, we assess the performance of PINN in terms of its accuracy and robustness under mesh refinement and as a function of the network architecture design. Displacement, Cauchy stress and accumulated plastic strain fields are compared to the finite element results for the same problem for the purposes of this assessment, which is intended to provide insight and guidance for the future designs of PINN for plasticity and related problems in solid mechanics.

1. Introduction

In recent years, there has been a rapid growth in the use of machine learning for various disciplines in engineering such as biotechnology (Alipanahi et al., 2015), fluid mechanics (Kutz, 2017) and battery technology (Zhang et al., 2018). In the field of computational mechanics, machine learning and deep learning have been utilized for a variety of applications: Liu et al. (2020) used neural networks to determine the fracture toughness of a pre-notched microcantilever where empirical solutions are often inaccurate. Hoerig et al. (2019) used a Cartesian neural network to acquire accurate Young's modulus from elastic imaging through a data-driven approach. Mozaffar et al. (2019) demonstrated that material plasticity can be precisely and efficiently predicted by a deep learning method. Huang et al. (2020) proposed a data-driven machine-learning-based material modeling framework for both elasticity and plasticity constitutive relations. Masi et al. (2021) proposed a thermodynamics-based artificial neural network for constitutive modeling of strain-rate-independent elasto-plastic materials. Niu and Srivastava (Niu and Srivastava, 2022a,b) developed a methodology based on finite element simulations and convolutional neural networks that can quantify embedded cracks and classify interacting flaws in real structures with high accuracy. Bonatti and Mohr (2022) utilized a self-consistent recurrent

* Corresponding author at: School of Engineering, Brown University, Providence, RI 02912, USA.
E-mail address: vikas_srivastava@brown.edu (V. Srivastava).

https://doi.org/10.1016/j.jmps.2022.105177
Received 17 August 2022; Received in revised form 9 November 2022; Accepted 9 December 2022
Available online 15 December 2022

neural network architecture for a surrogate material model representing elasto-plastic materials. Jin et al. (2022) used a finite element simulation trained convolutional neural network to inversely determine the accurate dynamic cohesive properties from the interferometer fringe images. Li and Chen (2022) proposed an equilibrium-based convolutional neural network to model the constitutive behavior of hyperelastic materials. The ideas presented in the aforementioned references rely on the data-driven nature of machine learning models and take advantage of existing advanced numerical methods, such as the finite element method (FEM), to create large training data. Supplied with a sufficient amount of data, machine learning models can effectively and accurately solve the inverse problems where the underlying information, such as geometry or material constitutive laws, are inferred implicitly from the simulation results.

While machine learning has had tremendous success in purely data-driven approaches, many applications are not sufficiently rich in data to take advantage of these methods. To overcome the data sparsity issue, physics-informed neural networks (PINN) were introduced in Raissi et al. (2019) as a novel approach to encode the known partial differential equations (PDEs) explicitly into data-driven models. Since then, significant research effort has been exerted to use PINN as a deep learning algorithm to solve PDEs, largely enabled by the built-in ability of the neural networks to perform automatic differentiation (Baydin et al., 2015). For many boundary value problems (BVPs), PINN can be implemented as a forward solver in a fairly straightforward manner. It is able to solve the governing PDEs without invoking the training data due to the fact that a neural network deep enough can be regarded as a universal function approximator (Hornik et al., 1989). PINN has been used for solving many types of PDEs, including stochastic PDEs (Zhang et al., 2019) and fractional PDEs (Pang et al., 2019), as well as in the various engineering fields, including fluid mechanics (Raissi et al., 2020), biomechanics (Yin et al., 2021), nondestructive testing (Shukla et al., 2020), metal additive manufacturing (Zhu et al., 2021) and optics (Chen et al., 2020). On the computational front, solid mechanics, which often deals with BVPs that are posed in a weak form over complex geometry and are governed by mechanical equilibrium involving sophisticated kinematics and constitutive relations, is dominated by finite elements (Chester et al., 2015; Bai et al., 2021; Zhong and Srivastava, 2021; Srivastava et al., 2010). However, compared to finite element methods that require sophisticated mesh generation, PINN is "mesh-free" in that it only makes use of individual residual points, an attribute that is shared with meshfree methods and that makes the domain discretization simpler (Chen et al., 2017; Bessa et al., 2014). The implementation of new material models, which may also be done by means of neural networks, is relatively straightforward in PINN. Hence a number of forward and inverse problems in solid mechanics have been formulated and solved using PINN: Kadeethum et al. (2020) utilized PINN to solve nonlinear diffusivity and Biot's equations. Rao et al. (2021) presented a PINN structure that used a mixed-variable output to model elastodynamics problems without resorting to labeled data. Zhang et al. (2020) applied PINN for solving an inverse problem identifying the mechanical properties of hyperelastic soft tissues in the sense of elasticity imaging. Haghighat et al. (2021) studied a set of forward and inverse problems in solid mechanics using PINN for linear-elastic and small-strain elasto-plastic materials. Goswami et al. (2022) proposed a physics-informed variational neural network for predicting crack paths in quasi-brittle materials. Abueidda et al. (2021) used PINN to forward-solve a few exemplary BVPs in solid mechanics employing different constitutive laws including linear elasticity, hyperelasticity and small-strain elasto-plasticity. Zhang et al. (2022) presented a general framework based on PINN for identifying unknown geometric and material parameters for internal defects in a structure. Arora et al. (2022) demonstrated a physics-informed neural-network-based framework to model the strain-rate and temperature dependence of the deformation fields in elasto-viscoplastic solids. Several approaches to solve PDEs based on alternative formulations of the loss function were presented in the literature, such as, for example, the deep energy method that defines the loss function not based on the collocated strong-form PDE residual, but on the minimum energy principle (Samaniego et al., 2020; Abueidda et al., 2022; Fuhg and Bouklas, 2022).

References (Haghighat et al., 2021; Abueidda et al., 2021; Zhang et al., 2022; Arora et al., 2022) addressing the elasto-plastic response of materials using PINN were limited to problems with small strains and rotations. However, finite-strain plasticity is important in practice, with examples ranging from the development of large plastic zones near a notch tip to ductile metals undergoing significant plastic strains when subjected to manufacturing processes like rolling and forging (see, e.g., the application to highly plastically deformed stainless steel corrugated pipes used for the transport of liquefied natural gas (Srivastava et al., 2011; Kothari et al., 2019)). Hence, large-deformation and finite-strain plasticity formulations are needed for this important class of problems. In addition, in the limited work related to PINN for elasto-plastic BVPs, only monotonic loading conditions were discussed, while elastic unloading and subsequent plastic loading along with several other natural phenomena in the theory of plasticity were not presented. The present work addresses these gaps. We develop a PINN formulation that incorporates a theory of finite-strain plasticity with rate-independent, isotropic hardening. In addition, we consider multi-step loading, where the specimen is loaded beyond the initial yield point, unloaded to the elastic regime and reloaded to further develop plastic deformation. Moreover, we develop an auxiliary neural network to approximate the constitutive relations. Instead of directly supplying material parameters, the auxiliary network learns from individual stress–strain measurements. The material properties can be identified from the discrete measurements and the plastic response (hardening law) is fully approximated by the network. This approach, in principle, eliminates the need for empirical models of the material plastic response, as advocated in the works of Ortiz et al. (Kirchdoerfer and Ortiz, 2016; Eggersmann et al., 2019; Stainier et al., 2019).

The remainder of the paper is organized as follows. In Section 2 we provide a brief recap of the finite-strain plasticity constitutive model and the corresponding stress update algorithm. In Section 3 we discuss the PINN model architecture. In Section 4 we use a numerical example to demonstrate the capability of our PINN to solve a finite-strain elasto-plastic BVP. We also present an assessment of the PINN performance as a function of the network parameters and mesh density. We draw conclusions and provide a discussion of the results in Section 5.

2. Finite-strain elasto-plasticity

2.1. Kinematics and constitutive modeling

We briefly summarize an established theoretical framework for finite-strain, rate-independent plasticity with isotropic hardening (Gurtin et al., 2010). We define a deformation gradient and assume the Kröner-Lee multiplicative decomposition along with plastic incompressibility

$$\mathbf{F} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}} = \mathbf{F}^e\mathbf{F}^p, \qquad J = \det \mathbf{F}^e > 0, \qquad \det \mathbf{F}^p = 1. \tag{1}$$

The velocity gradient is given by

$$\mathbf{L} = \dot{\mathbf{F}}\mathbf{F}^{-1} = \mathbf{L}^e + \mathbf{F}^e\mathbf{L}^p\mathbf{F}^{e-1}. \tag{2}$$

The elastic and plastic distortion-rate tensors $\mathbf{L}^e$ and $\mathbf{L}^p$ are defined as

$$\mathbf{L}^e = \dot{\mathbf{F}}^e\mathbf{F}^{e-1}, \qquad \mathbf{L}^p = \dot{\mathbf{F}}^p\mathbf{F}^{p-1}. \tag{3}$$

The plastic stretching tensor can then be defined as the symmetric part of $\mathbf{L}^p$

$$\mathbf{D}^p = \frac{1}{2}(\mathbf{L}^p + \mathbf{L}^{pT}). \tag{4}$$

Polar decomposition of the elastic deformation tensor gives

$$\mathbf{F}^e = \mathbf{R}^e\mathbf{U}^e, \tag{5}$$

where $\mathbf{R}^e$ is the elastic rotation tensor and $\mathbf{U}^e$ is the right elastic stretching tensor. An elastic Hencky (logarithmic) strain can be subsequently defined as

$$\mathbf{E}^e = \log \mathbf{U}^e. \tag{6}$$

We assume a specific free energy density form that can be written in terms of the Hencky strain $\mathbf{E}^e$,

$$\psi = G\left|\mathbf{E}_0^e\right|^2 + \frac{K}{2}(\mathrm{tr}\mathbf{E}^e)^2, \tag{7}$$

where G and K are the shear and bulk modulus, respectively, and $\mathbf{E}_0^e$ represents the deviatoric part of the strain tensor. For the above free energy, the Mandel stress tensor is given by

$$\mathbf{M}^e = 2G\mathbf{E}_0^e + K(\mathrm{tr}\mathbf{E}^e)\mathbf{1}. \tag{8}$$

The Cauchy stress $\boldsymbol{\sigma}$ and the first Piola–Kirchhoff (PK) stress $\mathbf{P}$ may be expressed as

$$\boldsymbol{\sigma} = J^{-1}\mathbf{R}^e\mathbf{M}^e\mathbf{R}^{eT}, \tag{9}$$

$$\mathbf{P} = J\boldsymbol{\sigma}\mathbf{F}^{-T}. \tag{10}$$

We define an equivalent tensile stress in terms of the deviatoric Mandel stress as $\bar{\sigma} = \sqrt{3/2}|\mathbf{M}_0^e|$ and write the flow rule as

$$\dot{\mathbf{F}}^p = \mathbf{D}^p\mathbf{F}^p, \qquad \mathbf{D}^p = \frac{3}{2}\dot{\bar{\epsilon}}^p\frac{\mathbf{M}_0^e}{\bar{\sigma}}, \qquad \mathbf{N}^p = \sqrt{\frac{3}{2}}\frac{\mathbf{M}_0^e}{\bar{\sigma}}, \tag{11}$$

where $\dot{\bar{\epsilon}}^p$ is the equivalent tensile plastic strain rate and $\mathbf{N}^p$ is the plastic direction. The yield function is chosen as

$$f = \bar{\sigma} - Y(\bar{\epsilon}^p), \tag{12}$$

where $Y(\bar{\epsilon}^p)$ is the flow strength and $\bar{\epsilon}^p$ is the accumulated plastic strain. Lastly, we can define the no-flow condition

$$\mathbf{D}^p = 0 \quad \text{if} \quad \begin{cases} f < 0 & \text{or} \\ f = 0 & \text{and} \quad \dot{f} < 0, \end{cases} \tag{13}$$

and the consistency condition

$$\text{if} \quad \mathbf{D}^p \neq 0 \quad \text{then} \quad f = \dot{f} = 0. \tag{14}$$

2.2. Stress update

Here we briefly summarize a stress update method used that is based on the backward Euler time integration of the constitutive relations presented in the previous section and results in the classical elastic-predictor and plastic-corrector return mapping algorithm.

The problem statement is as follows: Given $\mathbf{F}_n, \mathbf{F}_n^p$ and $\bar{\epsilon}_n^p$ at time t_n and $\mathbf{F}_{n+1}$ at time $t_{n+1} = t_n + \Delta t$, compute $\boldsymbol{\sigma}_{n+1}$, $\mathbf{F}_{n+1}^p$ and $\bar{\epsilon}_{n+1}^p$ at time t_{n+1}. To carry out the stress update at each material point, first, a set of trial quantities is calculated assuming the response is fully elastic:

$$\begin{cases} \mathbf{F}_{tr}^e = \mathbf{F}_{n+1}\mathbf{F}_n^{p-1}, \\ \mathbf{C}_{tr}^e = \mathbf{F}_{tr}^{eT}\mathbf{F}_{tr}^e, \\ \mathbf{U}_{tr}^e = \sqrt{\mathbf{C}_{tr}^e}, \\ \mathbf{R}_{tr}^e = \mathbf{F}_{tr}^e\mathbf{U}_{tr}^{e-1}, \\ \mathbf{E}_{tr}^e = \log(\mathbf{U}_{tr}^e), \\ \mathbf{M}_{tr}^e = 2G\mathbf{E}_{tr,0}^e + K(\mathrm{tr}\mathbf{E}_{tr}^e)\mathbf{1}, \\ \boldsymbol{\sigma}_{tr} = (\det \mathbf{F}_{n+1})^{-1}\mathbf{R}_{tr}^e\mathbf{M}_{tr}^e\mathbf{R}_{tr}^{eT}, \\ \bar{\sigma}_{tr} = \sqrt{\frac{3}{2}}|\mathbf{M}_{tr,0}^e|, \\ \mathbf{N}_{tr}^p = \sqrt{\frac{3}{2}}\frac{\mathbf{M}_{tr,0}^e}{\bar{\sigma}_{tr}}. \end{cases} \tag{15}$$

The yield function is then evaluated using the trial state to determine if the response is elastic or plastic:

$$f_{tr} = \bar{\sigma}_{tr} - Y(\bar{\epsilon}_n^p). \tag{16}$$

If $f_{tr} \leq 0$, no plastic flow occurs and the trial stress is the actual stress at this step, namely,

$$\boldsymbol{\sigma}_{n+1} = \boldsymbol{\sigma}_{tr}, \quad \mathbf{F}_{n+1}^p = \mathbf{F}_n^p \quad \text{and} \quad \bar{\epsilon}_{n+1}^p = \bar{\epsilon}_n^p. \tag{17}$$

However, if $f_{tr} > 0$, plastic flow occurs and the plastic strain increment is computed by solving the nonlinear equation

$$g(\Delta\bar{\epsilon}^p) = \bar{\sigma}_{tr} - 3G\Delta\bar{\epsilon}^p - Y(\bar{\epsilon}_n^p + \Delta\bar{\epsilon}^p) = 0. \tag{18}$$

The Cauchy stress, plastic deformation gradient, and accumulated plastic strain are updated, accordingly, as

$$\mathbf{M}_{n+1}^e = \mathbf{M}_{tr}^e - \sqrt{6}G\Delta\bar{\epsilon}^p\mathbf{N}_{tr}^p, \quad \boldsymbol{\sigma}_{n+1} = (\det \mathbf{F}_{n+1})^{-1}\mathbf{R}_{tr}^e\mathbf{M}_{n+1}^e\mathbf{R}_{tr}^{eT}, \tag{19}$$

$$\mathbf{D}_{n+1}^p = \sqrt{\frac{3}{2}}\frac{\Delta\bar{\epsilon}^p}{\Delta t}\mathbf{N}_{tr}^p, \quad \mathbf{F}_{n+1}^p = \exp(\Delta t\mathbf{D}_{n+1}^p)\mathbf{F}_n^p, \tag{20}$$

$$\bar{\epsilon}_{n+1}^p = \bar{\epsilon}_n^p + \Delta\bar{\epsilon}^p. \tag{21}$$

This completes the stress update for a given material point and load step.

3. PINN for large-deformation elasto-plasticity

3.1. Dense neural network structure

We now present our procedure to construct a PINN for the approximate solution of large-deformation elasto-plastic problems on arbitrary geometric configurations. The PINN has a generic structure, with one input layer, one output layer and a number of hidden layers in between. We adopt the formulation for a plane-strain problem in 2D. A total Lagrangian description is used with $\mathbf{X} = (X_1, X_2)$ denoting the spatial coordinates of the reference configuration. We assume that the loading path can be arbitrarily defined, and that the final deformed configuration is path- and history-dependent. We use a quasi-static approach and let the time t be the pseudo-time, which may be interpreted as a parameterization of the applied loading. The loading path is strictly followed to account for the history dependency.

In order to construct the PINN, we adopt a mixed formulation of the elasto-plastic problem in which the neural network approximates not only the displacement field $\hat{\mathbf{u}}(\mathbf{X}, t)$, but also the first PK stress field $\hat{\mathbf{P}}(\mathbf{X}, t)$, which is natural for the total Lagrangian formulation employed in this work. In addition, the plastic deformation gradient $\hat{\mathbf{F}}^p(\mathbf{X}, t)$, which is regarded as a state variable, is approximated in the PINN. Finally, the plastic strain increment $\Delta\hat{\bar{\epsilon}}^p(\mathbf{X}, t)$ at each step is also a direct output of the neural network. We denote by $\boldsymbol{\theta}$ the vector of all the trainable PINN parameters, including the weights and biases in the neurons of the network. With these notations, we can write our PINN formulation as

$$(\hat{\mathbf{u}}, \hat{\mathbf{P}}, \hat{\mathbf{F}}^p, \Delta\hat{\bar{\epsilon}}^p) = \mathcal{N}\mathcal{N}(\mathbf{X}, t; \boldsymbol{\theta}), \tag{22}$$

where all the quantities denoted by the ^ symbol are the unknowns of a problem, and are the direct outputs of the neural network, with $\hat{\mathbf{u}} = (\hat{u}_1, \hat{u}_2)$, $\hat{\mathbf{P}} = (\hat{P}_{11}, \hat{P}_{22}, \hat{P}_{12}, \hat{P}_{21})$, and $\hat{\mathbf{F}}^p = (\hat{F}_{11}^p, \hat{F}_{22}^p, \hat{F}_{12}^p, \hat{F}_{21}^p)$. Note that there are a total of 11 outputs at each spatial location $\mathbf{X}$ and time t, including the scalar plastic strain increment.

The choice of a mixed formulation is motivated by the fact that the network only needs to compute the first partial derivative of the unknown quantities with respect to the spatial coordinates of the reference configuration. Approximating the momentum balance equation in the primal form would necessitate the computation of the second partial derivatives of the displacement vector, which are much more costly than the first derivatives. Also, displacement formulation suffers from degraded accuracy and convergence issues as observed in Haghighat et al. (2021), Arora et al. (2022), pointing to the fact that mixed formulations may become more prominent in PINN-based approaches in the future.

3.2. Construction of the loss function

The BVP for the static equilibrium of a continuum solid body may be stated as

$$\text{Div } \mathbf{P} + \mathbf{f}_b = \mathbf{0} \quad \text{in} \quad \Omega, \qquad \mathbf{u} = \bar{\mathbf{u}} \quad \text{on} \quad \Gamma^u, \qquad \mathbf{P}\mathbf{N} = \bar{\mathbf{T}} \quad \text{on} \quad \Gamma^t. \tag{23}$$

Here, Ω is the problem reference domain, $\mathbf{f}_b$ is the prescribed body force, $\mathbf{u}$ is the unknown displacement field, $\bar{\mathbf{u}}$ is the prescribed displacement field on the boundary Γ^u, $\mathbf{P}$ is the unknown PK stress field, $\mathbf{N}$ is the unit outward normal vector on the reference-configuration boundary and $\bar{\mathbf{T}}$ is the prescribed traction field on the boundary Γ^t. The key idea of PINN is the construction of a loss function that explicitly encodes the discrete version of the underlying BVP. For this, we discretize the problem reference domain Ω using N_R residual points placed at locations $\mathbf{X}_R^A, A = 1, \dots, N_R$. We also place N_U and N_T points at locations $\mathbf{X}_U^A, A = 1, \dots, N_U$ and $\mathbf{X}_T^A, A = 1, \dots, N_T$ on Γ^u and Γ^t, respectively. With the assumption of no body force, we collocate the static equilibrium equation at each one of the residual points as

$$\text{Div } \hat{\mathbf{P}}(\mathbf{X}_R^A) = \mathbf{0}. \tag{24}$$

In order to satisfy the displacement and traction boundary condition, we also collocate them as

$$\hat{\mathbf{u}}(\mathbf{X}_U^A) = \bar{\mathbf{u}}(\mathbf{X}_U^A), \tag{25}$$

and

$$\hat{\mathbf{P}}\mathbf{N}(\mathbf{X}_T^A) = \bar{\mathbf{T}}(\mathbf{X}_T^A). \tag{26}$$

at each one of the Dirichlet and Neumann boundary points, respectively. The first PK stress and displacement fields are connected through an auxiliary relation, which is the key idea behind mixed methods. For this, we define an auxiliary PK stress field that is computed from the unknown displacement field

$$\hat{\mathbf{P}}' = \hat{\mathbf{P}}'(\hat{\mathbf{u}}; \theta_{\text{aux}}). \tag{27}$$

where θ_{aux} are frozen network parameters of a trained auxiliary neural network, which will be discussed more in detail in Section 3.3. To evaluate the stress $\hat{\mathbf{P}}'$, the deformation gradient is computed from the gradient of $\hat{\mathbf{u}}$ taken with respect to the spatial coordinates of the reference configuration. The computed deformation gradient is employed to drive the stress update algorithm shown in the previous section inside the neural network, which makes the loss function cognizant of the constitutive relation. The two stress quantities are set equal to one another by collocating the following equation at each residual point:

$$\hat{\mathbf{P}}(\mathbf{X}_R^A) - \hat{\mathbf{P}}'(\hat{\mathbf{u}}; \theta_{\text{aux}})(\mathbf{X}_R^A) = \mathbf{0}. \tag{28}$$

Plastic flow only occurs at the material points that have met the yield criterion. Hence, in the elastic region, $\Delta\hat{\epsilon}^p = 0$. In the plastic region, however, the return mapping algorithm computes the plastic strain increment via Eq. (18). These conditions may be expressed by means of an *if* statement and also collocated at each residual point as follows:

$$\Delta\hat{\epsilon}^p(\mathbf{X}_R^A) = 0 \text{ if } \hat{f}_{tr}(\mathbf{X}_R^A) \leq 0; \quad \hat{g}(\Delta\hat{\epsilon}^p)(\mathbf{X}_R^A) = 0 \text{ if } \hat{f}_{tr}(\mathbf{X}_R^A) > 0, \tag{29}$$

where $\hat{f}_{tr}$ is the yield function evaluated using the trial state for a given load step. Finally, we introduce the plastic deformation gradient $\hat{\mathbf{F}}^p$ as the output of the neural network (i.e., the unknown variable) and make it consistent with the computed plastic strain increment by means of yet another equation collocated at each residual point as

$$\hat{\mathbf{F}}^p(\mathbf{X}_R^A) - \hat{\mathbf{F}}^{p\prime}(\Delta\hat{\epsilon}^p)(\mathbf{X}_R^A) = \mathbf{0}, \tag{30}$$

where the function $\hat{\mathbf{F}}^{p\prime}(\Delta\hat{\epsilon}^p)$ is evaluated using Eq. (20).

At this stage, we have all the ingredients to construct the loss function $\mathcal{L}(\boldsymbol{\theta})$. For each loading step t, we follow the traditional approach and define $\mathcal{L}(\boldsymbol{\theta})$ as the weighted mean squared error with contributions coming from the discretized PDE, boundary conditions, and constitutive relations as

$$\mathcal{L} = \alpha_{PDE}\mathcal{L}_{PDE} + \alpha_{BC}\mathcal{L}_{BC} + \alpha_C\mathcal{L}_C. \tag{31}$$

Here $\mathcal{L}_{PDE}, \mathcal{L}_{BC}, \mathcal{L}_C$ are the individual loss functions for PDE, boundary conditions (BCs), constitutive relations, respectively, and α's are the corresponding weights. The expanded form of the individual loss functions, which penalize, in the least-squares sense, the errors in the collocation equations presented earlier, may be written out as

$$\mathcal{L}_{PDE} = \frac{1}{N_R}\sum_{A=1}^{N_R}\left|\text{Div } \hat{\mathbf{P}}(\mathbf{X}_R^A, t; \boldsymbol{\theta})\right|^2 \tag{32}$$

$$\begin{aligned}\mathcal{L}_{BC} &= \alpha_U\mathcal{L}_U + \alpha_T\mathcal{L}_T \\ &= \frac{\alpha_U}{N_U}\sum_{A=1}^{N_U}\left|\hat{\mathbf{u}}(\mathbf{X}_U^A, t; \boldsymbol{\theta}) - \bar{\mathbf{u}}(\mathbf{X}_U^A)\right|^2 + \frac{\alpha_T}{N_T}\sum_{A=1}^{N_T}\left|\hat{\mathbf{P}}\mathbf{N}(\mathbf{X}_T^A, t; \boldsymbol{\theta}) - \bar{\mathbf{T}}(\mathbf{X}_T^A, t)\right|^2\end{aligned} \tag{33}$$

$$\begin{aligned}\mathcal{L}_C &= \alpha_S \mathcal{L}_S + \alpha_F \mathcal{L}_F + \alpha_P \mathcal{L}_P \\ &= \frac{\alpha_S}{N_R} \sum_{A=1}^{N_R} \left| \hat{\mathbf{P}}(\mathbf{X}_R^A, t; \boldsymbol{\theta}) - \hat{\mathbf{P}}'(\hat{\mathbf{u}})(\mathbf{X}_R^A, t; \boldsymbol{\theta}, \boldsymbol{\theta}_{\text{aux}}) \right|^2 \\ &+ \frac{\alpha_F}{N_R} \sum_{A=1}^{N_R} \left| \hat{\mathbf{F}}^p(\mathbf{X}_R^A, t; \boldsymbol{\theta}) - \hat{\mathbf{F}}^{p\prime}(\Delta\hat{\bar{\epsilon}}^p)(\mathbf{X}_R^A, t; \boldsymbol{\theta}) \right|^2 \\ &+ \frac{\alpha_P}{N_R} \sum_{A=1}^{N_R} \left[\mathbb{1}_{\{\hat{f}_{tr} \le 0\}} \left| \Delta\hat{\bar{\epsilon}}^p(\mathbf{X}_R^A, t; \boldsymbol{\theta}) \right|^2 + \mathbb{1}_{\{\hat{f}_{tr} > 0\}} \left| \hat{g}(\Delta\hat{\bar{\epsilon}}^p(\mathbf{X}_R^A, t; \boldsymbol{\theta})) \right|^2 \right].\end{aligned} \tag{34}$$

Here, the subscripts S, F, P are used to denote the stress, plastic deformation gradient and plastic strain increment terms in the constitutive loss and $\mathbb{1}_{\{Z\}}$ is the indicator function that returns 1 when the argument Z is true and 0 otherwise. The indicator function with a sharp binary transition allows enforcement of the physical conditions of zero plastic strain in the elastic region and the correct value of $\hat{g}(\Delta\hat{\bar{\epsilon}}^p)$ in the plastic region. Eqs. (31) to (34) are for individual loading step t. For multiple loading steps, the returned loss value will be the summation of the losses for all the steps ($\sum_t$).

The PINN training process can be considered as an optimization problem to find the network parameters $\tilde{\theta}$ that minimize the loss function, namely,

$$\tilde{\theta} = \underset{\theta}{\operatorname{argmin}}\ \mathcal{L}(\theta). \tag{35}$$

Once the loss function is minimized, the approximated PINN solution of the elasto-plastic problem may be evaluated by passing the reference-configuration spatial coordinates and pseudo-time into the network with the optimal parameters $\tilde{\theta}$ as

$$(\hat{\mathbf{u}}, \hat{\mathbf{P}}, \hat{\mathbf{F}}^p, \Delta\hat{\bar{\epsilon}}^p) = \mathcal{N}\mathcal{N}(\mathbf{X}, t; \tilde{\theta}). \tag{36}$$

3.3. Learning the material law from discrete measurements

Material properties are fundamental to the prediction of how an object behaves when subjected to external forces. In the context of plasticity, additional parameters are often needed to describe the yielding behavior. For an isotropic hardening material, the flow stress is usually formulated as a function of the accumulated plastic strain (Chaaba, 2010). Based on the shape of the stress–strain curve, various types of phenomenological hardening laws have been proposed such as power-law hardening and Voce-type hardening (Lu et al., 2018; Voce, 1948). These models attempt to capture the shape of the stress–strain curve by assuming a specific functional form with a few parameters. Earlier works related to plasticity hard-code these hardening laws into the loss functions as one would do for writing a user material subroutine (Abueidda et al., 2021; Arora et al., 2022). However, the material properties are acquired from the discrete measurements in the actual experimental tests. Inaccuracy in representing the true physics may occur when the proposed models are not able to capture and reflect the full response of the material.

To overcome this limitation, we present a different approach that is model-free, as advocated in Ortiz et al. (Kirchdoerfer and Ortiz, 2016; Eggersmann et al., 2019; Stainier et al., 2019), and motivated by Tang et al. (2019). Instead of using phenomenological material hardening laws and parameters, we use the individual data points from a stress–strain curve as the input. In this work we focus on a single curve from a representative uniaxial tension test. Prior to training of the PINN described earlier, we first define an auxiliary network to approximate the entire material response, from elastic to plastic. This auxiliary network is a four-layer artificial neural network with 30 neurons in each of the two hidden layers. Once it is trained from individual data points, it analyzes the stress–strain curve and automatically determines the Young's modulus from the slope in the elastic region and initial yield stress indicated by the change of slope from elastic response to a less stiff plastic response. In the plastic regime, the auxiliary network outputs the flow strength for a given accumulated plastic strain value which can be expressed as

$$\hat{Y}(\bar{\epsilon}^p) = \mathcal{N}\mathcal{N}_{\text{aux}}(\bar{\epsilon}^p; \theta_{\text{aux}}) \tag{37}$$

The flow strength is subsequently used for the calculation of auxiliary PK stress $\hat{\mathbf{P}}'$ in the return mapping algorithm. As a result, this auxiliary network entirely replaces the explicit modeling of the hardening law. The complete architecture of the PINN formulation for fine-strain elasto-plastic problems is illustrated in Fig. 1.

4. Numerical example

4.1. Boundary value problem setup

We consider a plane-strain, hole-in-a-plate square geometry that is subjected to uniaxial tensile loading (see Fig. 2(a)). The units and material parameters are specified dimensionless as standard in the PINN community. The plate has dimensions 2 × 2 and the hole of radius 0.5 is placed at the center of the plate. The traction load $\mathbf{T}$ is uniform in space and acts in the 2-direction ($T_1 = 0$) at the top and bottom edges of the plate. It is defined on the reference configuration as a dead load. The left and right edges are traction-free. Due to the symmetry of the problem, we only consider a (upper right) quarter of the geometry. A uniform mesh of 40 × 40 residual points is generated for the quarter plate and the points that fall into the hole region are subsequently deleted.

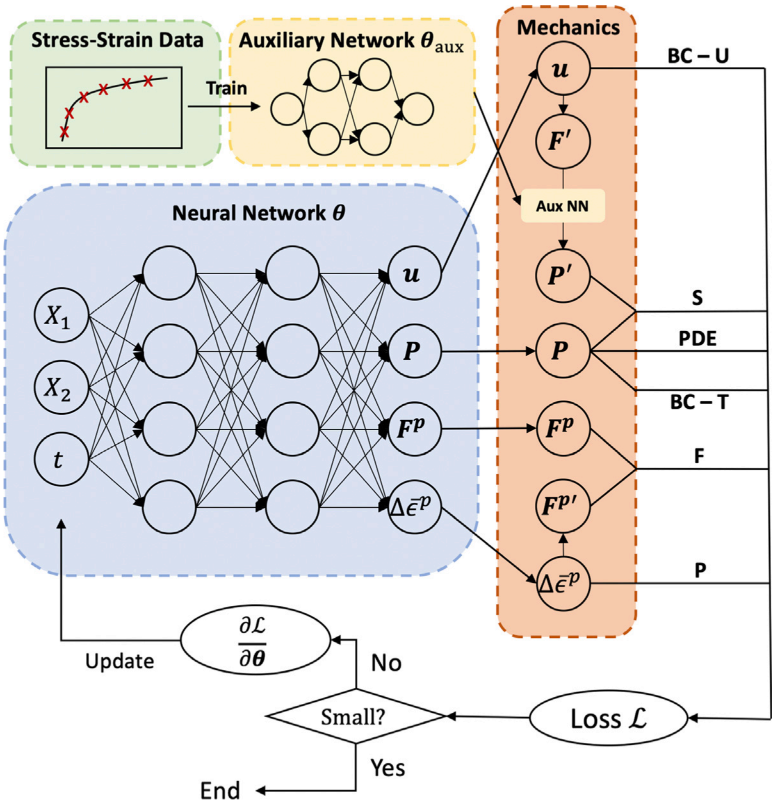

Fig. 1. Architecture of the PINN for finite-strain elasto-plasticity. The setup is based on the plane-strain formulation with multiple (pseudo)-time steps. The loss function is formulated based on the PDE, boundary conditions and constitutive relations. An auxiliary network is embedded in the architecture and used to model the material response by learning from discrete stress–strain data prior to the training of PINN.

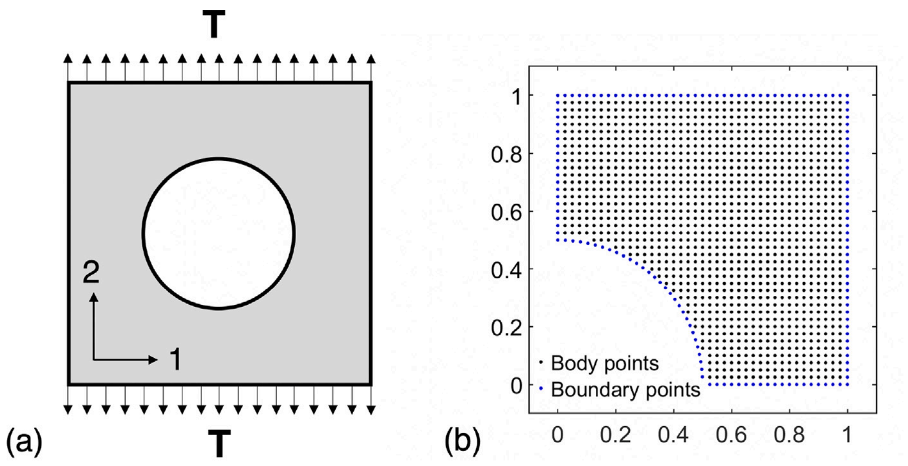

Fig. 2. (a) Schematic of the numerical example in which a plate with a hole is subject to uniaxial tensile loading in the 2-direction. (b) Distribution of the residual points in the reference configuration for a quarter of the geometry.

In addition, 30 residual points are placed uniformly on the circumference of the quarter hole. The placement of residual points is shown in Fig. 2(b).

For the loading condition, we consider three steps in total. Initially, the plate is free of stress and plastic strain with zero traction applied $\mathbf{T} = (0, 0)$. During step 1, the plate is subject to a traction load $\mathbf{T} = (0, 0.15)$ that causes yielding near the hole where the maximum stress concentration occurs. During step 2, the loading is reduced to $\mathbf{T} = (0, 0.1)$ for elastic unloading. In the last step 3 the final applied loading is $\mathbf{T} = (0, 0.18)$, which leads to more plastic straining in the plate. The Young’s modulus is chosen to be 2

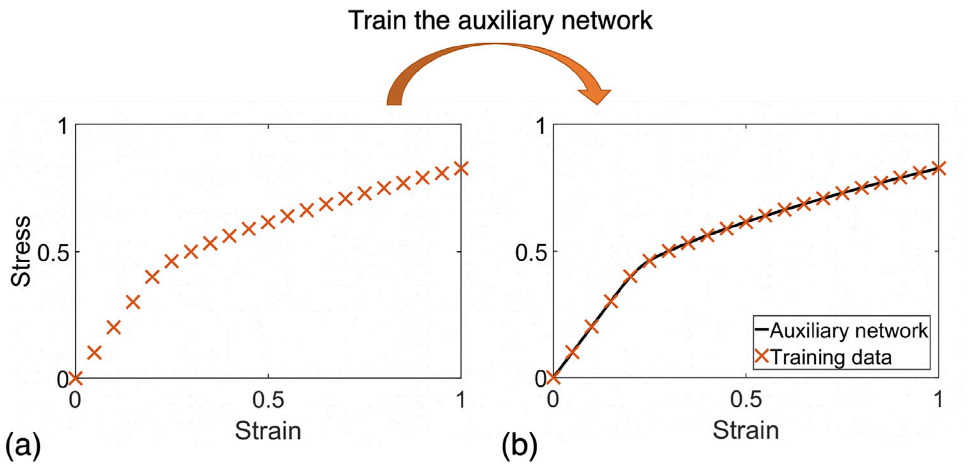

Fig. 3. (a) Discrete training data points from a stress–strain curve representing uniaxial tension test. The hardening law resembles a power-law form. (b) The auxiliary neural network learns from the training data and models the material response inside the PINN architecture.

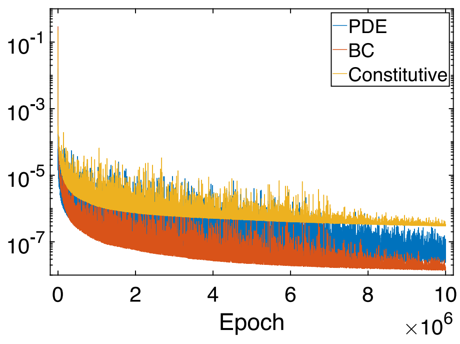

Fig. 4. Training losses for PDE, BC and constitutive relations. Y-axis is in *log* scale.

and Poisson's ratio is 0.3. We use a power-law hardening of the form

$$Y(\bar{\epsilon}^p) = Y_0 + H(\bar{\epsilon}^p)^n, \tag{38}$$

for the material response where $Y_0 = 0.4$, $H = 0.5$ and $n = 0.7$. This is only used to define the material characteristics but the material model is not provided to our model-free auxiliary network + PINN approach. The material response data points are generated using the material law specified above and then the stress–strain data is used to train the auxiliary network as discussed in the previous section. This process is summarized in Fig. 3 and we also show the approximated stress–strain curve by the auxiliary network after it has been trained. Because the Poisson's ratio cannot be directly acquired from a uniaxial tension test data, $\nu = 0.3$ is the only material parameter supplied directly to the neural network.

The PINN is implemented in the PyTorch framework and has 4 hidden layers with 30 neurons in each layer (Paszke et al., 2019). Adam optimizer is used as the optimization algorithm with an initial learning rate of 0.001 (Kingma and Ba, 2015). During the training, the learning rate is gradually decreased using an exponential decay scheduler. The activation function is the 'Tanh' function, except at the last layer, we have used 'Softplus' activation function. The training is performed on the computation cluster (Center for Computation and Visualization at Brown University) with GPU acceleration.

We use finite element software Abaqus to establish the reference solution and validate the results from PINN. A finite strain, rate independent isotropic hardening plasticity model presented in this paper is implemented in a user material subroutine (UMAT). We define pointwise and global L_2 errors to evaluate the accuracy of PINN solutions. The normalized L_2 errors for the displacement, Cauchy stress and accumulated plastic strain are defined as

$$\text{Displacement: } L_2^U = \frac{\|\hat{\mathbf{u}} - \mathbf{u}_{ref}\|_2}{\|\mathbf{u}_{ref}\|_2}, \quad \mathbf{u} = [u_1, u_2], \tag{39}$$

$$\text{Stress: } L_2^S = \frac{\|\hat{\boldsymbol{\sigma}} - \boldsymbol{\sigma}_{ref}\|_2}{\|\boldsymbol{\sigma}_{ref}\|_2}, \quad \boldsymbol{\sigma} = [\sigma_{11}, \sigma_{22}, 2\sigma_{12}], \tag{40}$$

$$\text{Accumulated plastic strain: } L_2^P = \frac{\|\hat{\bar{\epsilon}}^p - \bar{\epsilon}^p_{ref}\|_2}{\|\bar{\epsilon}^p_{ref}\|_2}, \tag{41}$$

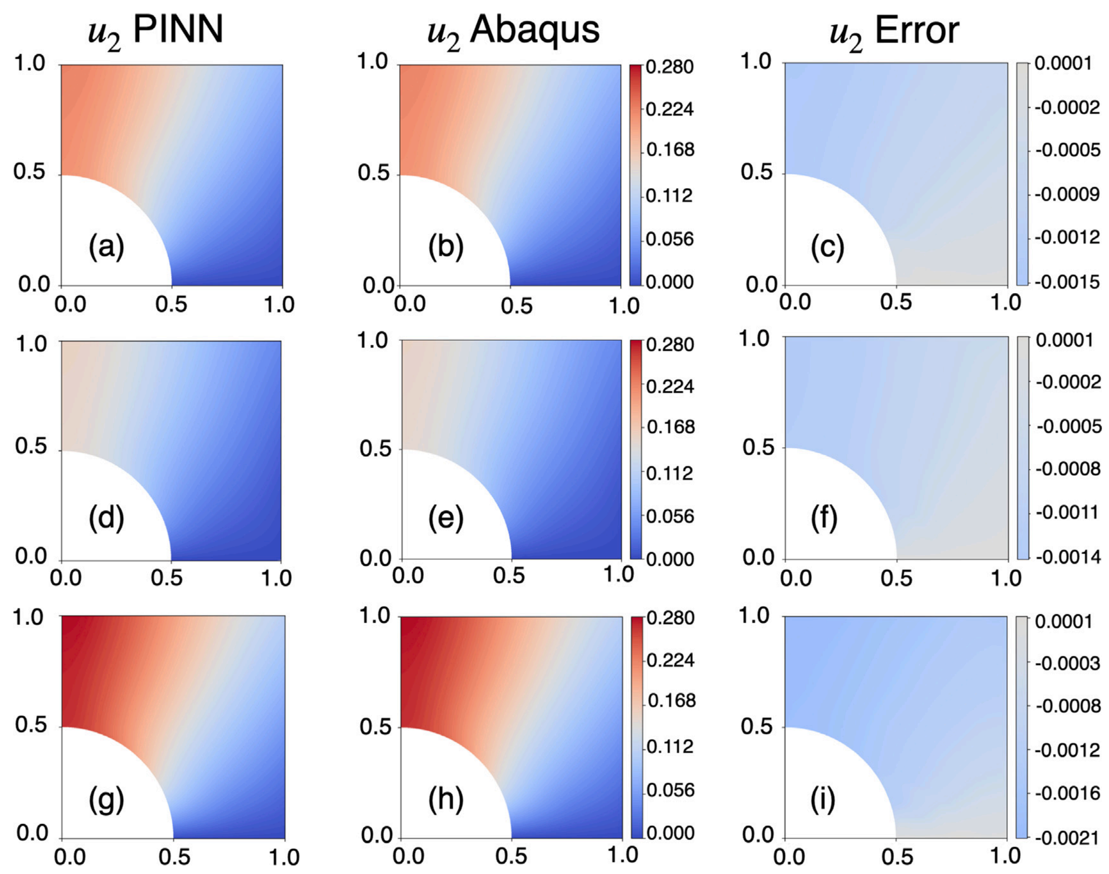

Fig. 5. Displacement in 2-direction from PINN and Abaqus and the error for the three load steps. Top row: Step 1 - loading. Middle row: Step 2 - elastic unloading. Bottom row: Step 3 - reloading to a higher load. (a)(d)(g): PINN, (b)(e)(h) Abaqus, (c)(f)(i) pointwise error.

where the reference quantities are from the benchmark finite element solutions computed on a numerically converged fine grid of approximately 150 × 150 bilinear elements in Abaqus. Note that we select a relatively simple power-law type hardening in this example to enable a straightforward comparison with the finite element solution. In principle, we could have selected a hardening law described by a curve of an arbitrary shape without any difficulties for PINN due to the model-free approach.

4.2. Multiple loading steps results

First, we show the approximate PINN solutions for the displacement, stress and plastic strain fields. We will use the term "plastic strain" instead of "accumulated plastic strain" from now on for brevity. Considering three pseudo-time steps in total, PINN has been trained for 10 million epochs. After the network has been trained, we created a testing grid consisting of 1000 × 1000 points as the input to the network. The training losses for each term (PDE, BCs and constitutive laws) are shown in Fig. 4.

The benchmark/reference finite element solutions are interpolated into the same testing grid points to calculate the errors. Since the geometry is subject to a traction type loading in the 2-direction, we show the displacement u_2, Cauchy stress σ_{22} and plastic strain in Figs. 5, 6, and 7, respectively, for the three steps of loading, elastic unloading (partial), and reloading to a higher load described previously. These figures show that the approximate solutions coming from PINN are comparable to the Abaqus results, with the errors being relatively small. From the displacement and stress contour, the elastic unloading is clearly shown in the middle row, which is step 2. Fig. 7 shows that the plastic strain is frozen in step 2 during the elastic unloading. The L_2 errors are given in Table 1. The errors for the displacement and stress are both below 1%. Plastic strain has a high error due to the fact that most of the plate is still behaving elastically and the reference L_2 norm for the plastic strain is small. Overall, we can see that PINN is able to fairly accurately approximate a classic finite strain elasto-plastic BVP. Multiple steps including plastic loading and elastic unloading as well as stress concentration features are well captured by the PINN at this level of resolution. We also emphasize that the plastic response (hardening law) was not given explicitly but in the form of discrete measurement data from an experimentally obtainable stress–strain curve that corresponds to a uniaxial tension test. The response is learned by an auxiliary network, which avoids the empiricism employed in classical material modeling.

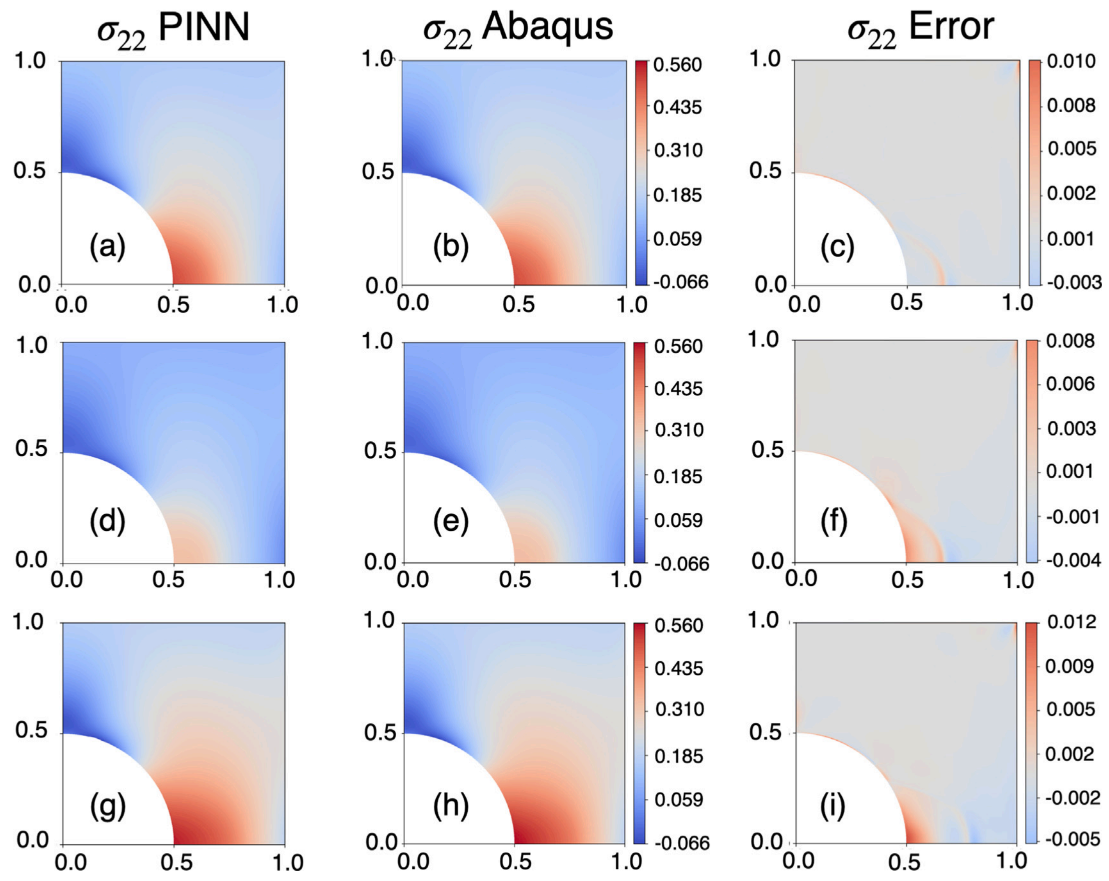

Fig. 6. Cauchy stress 22-component from PINN and Abaqus and the error for the three load steps. Top row: Step 1 - loading. Middle row: Step 2 - elastic unloading. Bottom row: Step 3 - reloading to a higher load. (a)(d)(g): PINN, (b)(e)(h) Abaqus, (c)(f)(i) pointwise error.

Table 1
PINN L_2 errors relative to the fine-mesh FEM solution for the displacement, stress and plastic strain at three loading steps.

Step	1	2	3
Displacement	0.68%	0.93%	0.81%
Stress	0.35%	0.74%	0.55%
Plastic strain	2.75%	3.58%	1.98%

4.3. Assessment of the neural network performance

PINN as a general PDE solver has been investigated extensively. For the special interest in solid mechanics, we would like to examine its unique performance for finite-strain elasto-plasticity and evaluate the performance with different network structures. We have demonstrated both plastic loading and elastic unloading in the previous section, so we turn our attention to the network itself and consider only one-step loading. In this case, the pseudo-time input is suppressed. Also, the auxiliary network for constitutive modeling is not used in order to simplify the computations. The geometry, BCs and mesh remain the same, and the traction prescribed is $\mathbf{T} = (0, 0.18)$, corresponding to step 3 in the previous section. PINN is trained for 3 million epochs for the study considered here.

When modeling material response, a wide class of material models have a discontinuity at the transition from elasticity to plasticity. This results in sharp elastic–plastic interfaces that develop in the computational domain. When computing the PDE residual (i.e., the discrete equilibrium equation), the derivatives of stress at the elastic–plastic boundary can be very large and numerically unfavorable for the network learning as well as for the approximation itself. The mixed formulation produces a smooth stress field and could lead to inaccuracy at the elastic–plastic boundary where the solution is supposed to be non-smooth. To control the smoothness, we investigate the effect of different activation functions, only at the last network layer, by considering 'Tanh', 'ReLU' and 'Softplus' options whose expressions are given by

$$Tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}, \tag{42}$$

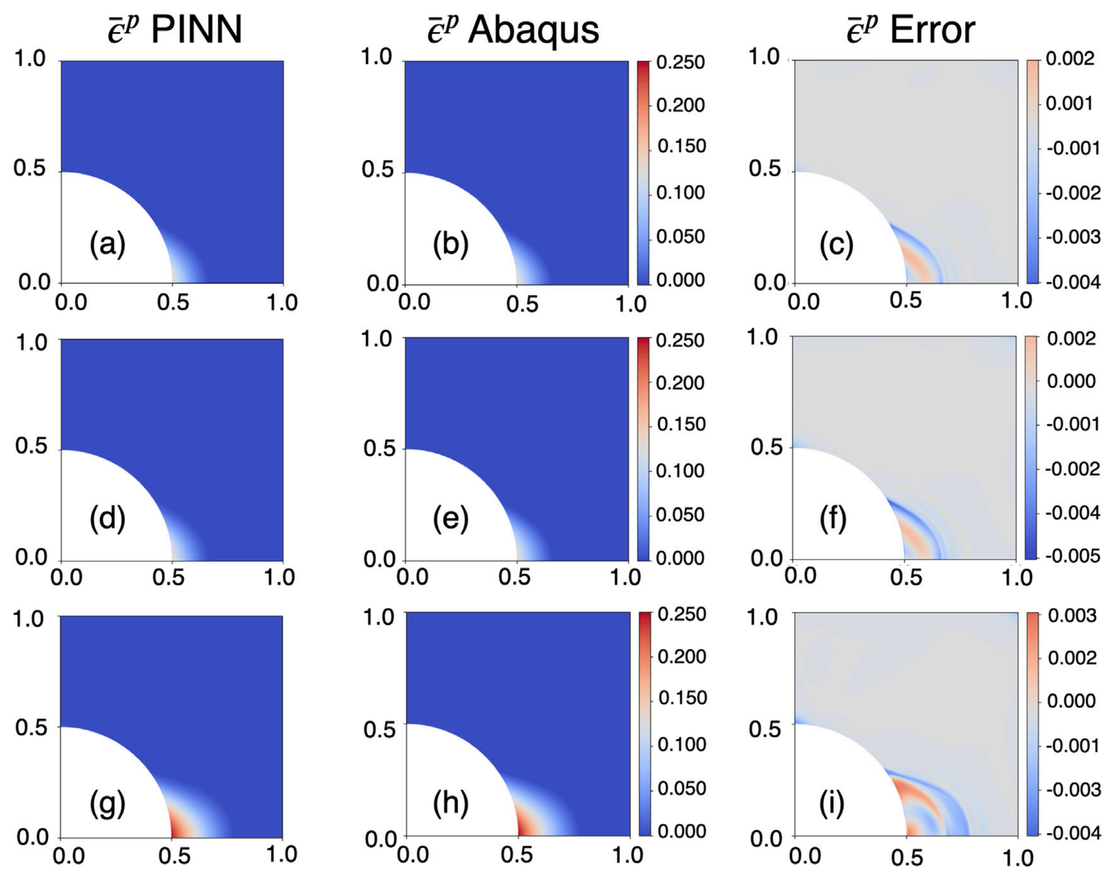

Fig. 7. Plastic strain from PINN and Abaqus and the error for the three load steps. Top row: Step 1 - loading. Middle row: Step 2 - elastic unloading. Bottom row: Step 3 - reloading to a higher load. (a)(d)(g): PINN, (b)(e)(h) Abaqus, (c)(f)(i) pointwise error.

Table 2
L_2 errors for the displacement, stress and plastic strain solutions for different activation functions at the last layer.

	Tanh	ReLU	Softplus
Displacement	0.08%	0.14%	0.02%
Stress	0.37%	0.37%	0.28%
Plastic strain	1.46%	1.44%	0.64%

$$ReLU(x) = max(0, x), \tag{43}$$

$$Softplus(x; \beta) = \frac{1}{\beta} \log[1 + e^{\beta x}], \tag{44}$$

where β is a user-defined parameter, which we set equal to 5. Softplus is a smooth approximation of the ReLU function and a larger β represents a sharper transition at $x = 0$. Fig. 8(a)–(c) plots the three activation functions.

We summarize the errors in Table 2. The solutions for displacement and stress fields are quite accurate using all three activation functions, thus we examine the pointwise error for the plastic strain, which is shown in Fig. 8(d)–(f). From the figures we can see that large errors indeed occur at the elastic–plastic boundary (see Fig. 7(i)). Tanh and ReLU activation functions have comparable error values. For Tanh, some regions that should remain strictly elastic have non-zero plastic strains due to the smooth approximation while the elastic–plastic boundary is very clear for ReLU owing to its functional form. Softplus has the smallest error of the three which is visually obvious from the figure. The above results indicate that the choice of network design requires the understanding of the problem physics and that the errors may be significantly reduced by designing the activation functions with the right balance of smoothness and sharpness.

The Fourier random feature proposed in Tancik et al. (2020) is able to capture the high frequency features in the solution from PINN (Wang et al., 2021b). Considering the elastic–plastic boundary, high frequency components are likely to exist and influence

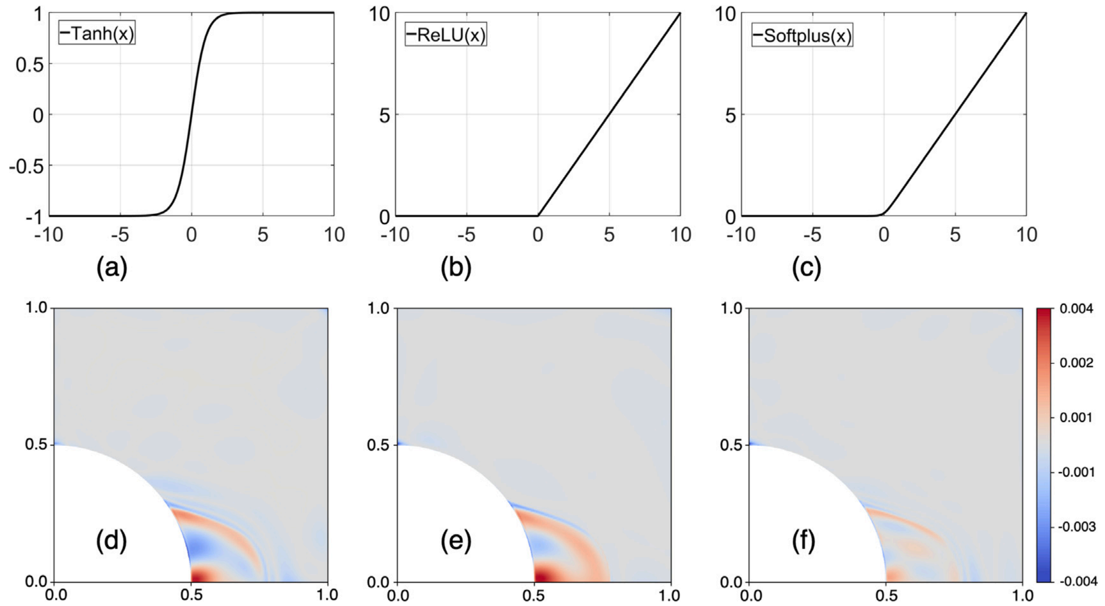

Fig. 8. Three activation functions: (a) Tanh, (b) ReLU, and (c) Softplus with $\beta = 5$. Pointwise error contours of the plastic strain with different activation functions: (d) Tanh, (e) ReLU, and (f) Softplus.

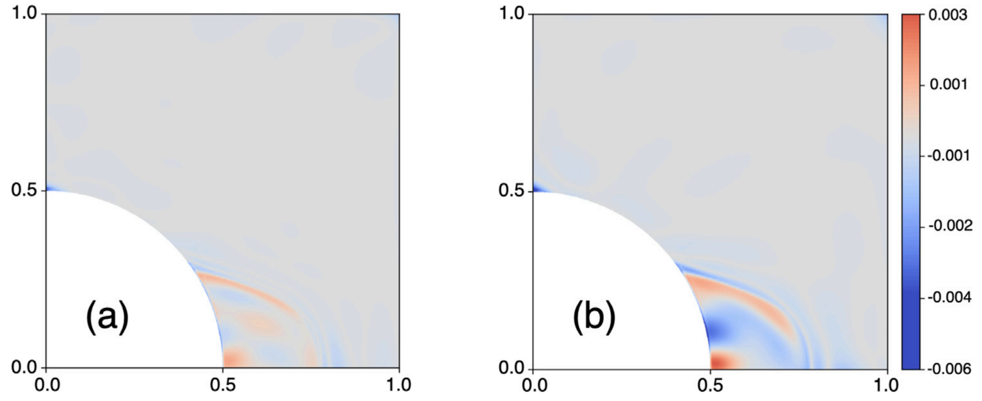

Fig. 9. Pointwise error contours of the plastic strain for the network (a) with and (b) without Fourier random feature matrix.

the final solutions. Hence we augment the PINN formulation by adopting this technique to minimize the errors associated with the discontinuity of the transition between the elastic and plastic behavior. A Fourier random feature mapping is performed on the neural network input $\mathbf{X} = (X_1, X_2)$, which can be expressed as

$$\boldsymbol{\gamma}(\mathbf{X}) = \begin{bmatrix} \cos(\mathbf{B}\mathbf{X}) \\ \sin(\mathbf{B}\mathbf{X}) \end{bmatrix}. \tag{45}$$

Here, $\mathbf{B}$ is the Fourier random feature matrix with dimensions $m\times 2$. Each entry of the matrix is sampled from a Gaussian distribution $N(0, \lambda^2)$, where $\lambda > 0$ is a user-specified hyper-parameter (Wang et al., 2021b). The vector $\boldsymbol{\gamma}(\mathbf{X})$ is used in place of $\mathbf{X}$ as the input to the PINN. We compare the performance of the two networks — one using the Fourier random feature matrix and one using the original input. The other network parameters remain the same and the last-layer activation function is Softplus. As indicated in Table 3, the Fourier random feature matrix clearly reduced the errors, especially for the displacement and plastic strain. Fig. 9 shows the pointwise error in the plastic strain for the two cases.

The network depth (i.e., the number of layers) and width (i.e., the number of neurons per layer) define the approximation capability for a fixed set of residual points. We have used a 4 × 30 network by default. Here, we present the results from networks with a few variations on the number of layers and neurons per layer. We considered four additional network configurations, namely, 4 × 20, 4 × 45, 2 × 30 and 6 × 30. The former two are for the investigation of the network width and the latter two are for the investigation of the network depth. Adopting the Softplus activation function and the Fourier random feature matrix while keeping other training parameters the same, the network performance is shown in Table 4. We observe that shallower and narrower networks

Table 3
L_2 errors for the displacement, stress and plastic strain solutions for network structure with and without Fourier random feature matrix.

	With **B**	Without **B**
Displacement	0.02%	0.16%
Stress	0.28%	0.34%
Plastic strain	0.64%	1.48%

Table 4
L_2 errors for the displacement, stress and plastic strain solutions for the network structures with different numbers of layers and neurons.

	Default	Layer width		Net depth	
	4 × 30	4 × 20	4 × 45	2 × 30	6 × 30
Displacement	0.02%	0.18%	0.03%	0.22%	0.03%
Stress	0.28%	0.45%	0.25%	0.40%	0.24%
Plastic strain	0.64%	2.14%	0.47%	1.85%	0.56%

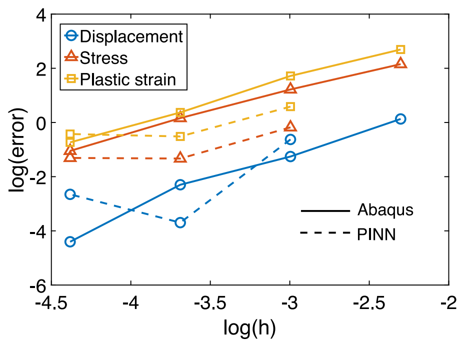

Fig. 10. Comparison of the errors between PINN and Abaqus for the residual-point mesh densities of 10 × 10, 20 × 20, 40 × 40 and 80 × 80. Errors are plotted on a log–log scale.

Table 5
L_2 errors for the displacement, stress, and plastic strain solutions for PINN and FEM for different mesh densities.

Mesh density	10 × 10		20 × 20		40 × 40		80 × 80	
Method	PINN	FEM	PINN	FEM	PINN	FEM	PINN	FEM
Displacement	–	1.14%	0.53%	0.28%	0.02%	0.10%	0.07%	0.01%
Stress	–	8.61%	0.84%	3.38%	0.26%	1.18%	0.27%	0.35%
Plastic strain	–	14.78%	1.80%	5.57%	0.60%	1.45%	0.65%	0.48%

are likely to have larger errors, while deeper and wider networks often have a negligible margin in improving the accuracy. A 4 × 30 network appears to be a good choice for the problem considered here.

Lastly, we evaluate the effect of residual point densities, which is analogous to mesh refinement/coarsening in FEM. By default we have used a mesh of 40 × 40 residual points on the entire computational domain before masking the points within the hole region. Three more mesh densities are considered here, which are (approximately) 10 × 10, 20 × 20 and 80 × 80 points. For a direct comparison with the finite element results, meshes corresponding to the above densities were first created in Abaqus. Then the exact locations of nodes for the meshes were used in PINN as the residual points for the training. Both the FEM and PINN solutions are linearly interpolated onto a 1000 × 1000 grid points to evaluate the errors, which are given in Table 5.

Fig. 10 shows the error convergence under mesh refinement and provides a comparison between PINN and FEM, using the data from Table 5. FEM results converge monotonically, with the stress and plastic strain converging at a slower rate than the displacement, as expected. The PINN convergence is less consistent. The 10 × 10 mesh in PINN is unable to generate a physically meaningful solution and is thus omitted from the plot. The 20 × 20- and 40 × 40-mesh PINN results show higher accuracy than the corresponding FEM results for the stress and plastic strains and appear to be converging at a similar rate. This may be a consequence of the fact that the "exact" solution is not sufficiently smooth due to the presence of plasticity, and thus the increase in the approximation power will not result in a higher convergence rate. However, the PINN error seems to saturate at the 40 × 40 mesh resolution and is seemingly increasing with subsequent refinement.

We take a closer look at the PINN solution on a coarse 10 × 10 mesh, which is reported in Fig. 11(a)–(f). The displacement and stress fields are both substantially smaller than in the FEM solution, and no plastic strain develops. Even with more training,

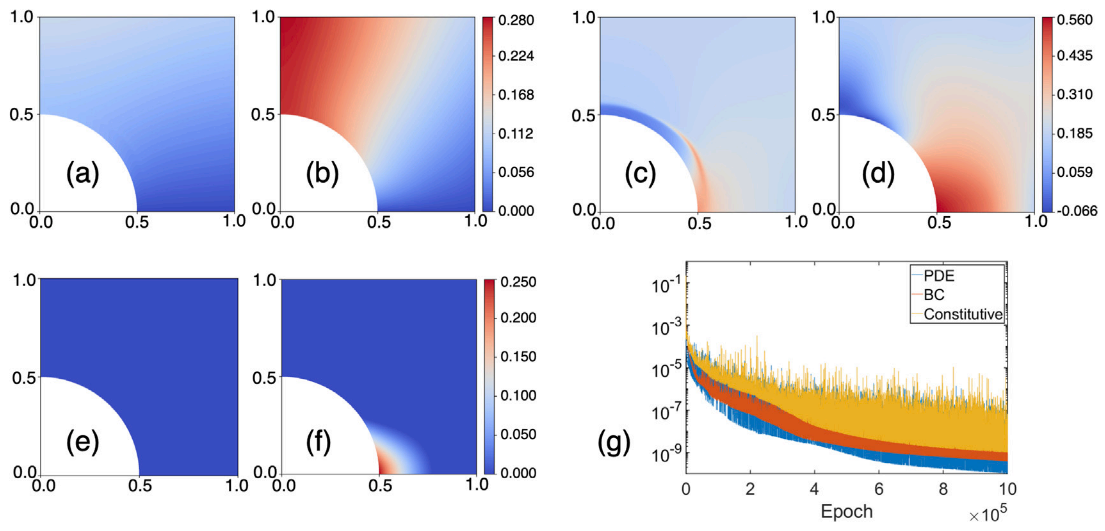

Fig. 11. Performance of PINN for the 10 × 10 mesh of residual points using a 4-layer-30-neuron network. (a) u_2 from PINN (b) u_2 from FEM (c) σ_{22} from PINN (d) σ_{22} from FEM (e) $\bar{\epsilon}^p$ from PINN (f) $\bar{\epsilon}^p$ from FEM (g) training loss.

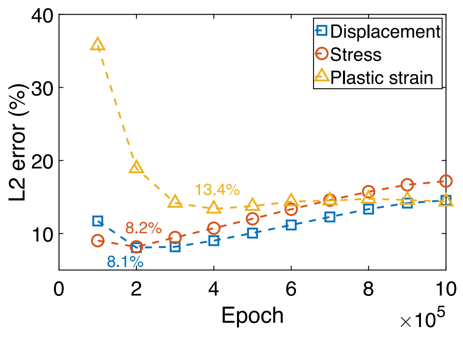

Fig. 12. L_2 error evolution for the mesh of 10 × 10 residual points using a smaller 3-layer-20-neuron network. Colored numbers mark the lowest error values and corresponding epoch for each variable.

while the overall loss continues to decrease (see Fig. 11(g)), the solution does not improve. We hypothesize that for a small number of residual points, our network structure (4 layers with 30 neurons each) may be too dense. To that end, we also tested a smaller network structure (3 layers with 20 neurons each) and trained the network for a million epochs. We were able to compute a meaningful solution on this mesh. The errors in the solutions at every 100k epochs are shown in Fig. 12. Although the smaller network is learning relatively better, the error evolution shows that the minimal errors, although still quite large, occur at 200k epoch for displacement (8.1%) and stress (8.2%) and at 400k for plastic strain (13.4%). Subsequently, the errors increase, likely showing a sign of over-fitting. These results suggest that in order to obtain a robust, FEM-like performance for PINN, one needs to have a good handle on the relationship between the mesh size, network structure (i.e., number of layers and neurons), and parameters employed in the definition of the loss function (not discussed here). These quantities are intrinsically linked and may not be chosen independently of one another if one expects accuracy and robustness.

The error increase for the 80 × 80 PINN solution may be attributed to small random fluctuation of the network parameters that are built into the neural network algorithms. When the loss function error is sufficiently small, changes in the solutions from the update of weights and biases in the network during each back-propagation step are large enough to cause the errors to increase for a given epoch. Another possible reason for the error increase is the insufficient network density. It is likely that more accurate results could be obtained by using a more dense network with this finer 80 × 80 mesh, however that would require significantly larger training epoch and longer training time.

5. Conclusions

In this work, we present the numerical formulation of PINN based on the theoretical framework for finite-strain elasto-plasticity. The PINN has a mixed formulation in which the displacement, first PK stress, plastic deformation gradient and plastic strain increment are approximated directly by the neural network. The loss function encodes the problem physics through the discretized elasto-plastic BVP consisting of the equilibrium equation, boundary conditions and constitutive relations. In addition, an auxiliary network is developed and implemented to replace explicit models of the hardening law. Through a numerical example of a plane-strain plate with a hole subject to uniaxial tensile loading we demonstrate that PINN is able to approximate the BVP solutions accurately for a general loading, unloading and reloading path. Plastic deformation, elastic unloading and stress concentrations are captured well in the method. We have considered quasi-static loading in our study; however, the method is general and not limited by this assumption. By utilizing modified PDEs, which include inertia terms in the momentum balance equation, the method presented can be extended to dynamic problems as well. The dynamic problems in PINN will need an additional temporal network input and appropriate initial conditions enforced through additional terms in the total loss function.

We consider a few configurations of the network structure to assess the PINN performance. FEM solution from a highly refined mesh is taken as a benchmark for the comparison. While traditional low-order FEM is a robust methodology where the quality of the solution primarily depends on the mesh resolution, the quality of the PINN results depends on a variety of factors. The design of the neural network needs careful attention and as shown in the present work, should not be independent of the residual point density, loss function design, and other material and BVP specific factors. In the context of finite-strain elasto-plasticity, we have shown that the choice of activation function at the last layer and input feature both affect the solution accuracy. Specifically, the Softplus activation function and Fourier random feature are useful in reducing the PINN approximation errors for our problem. We also show that PINN has difficulty finding physically meaningful solutions when the residual-point mesh is too coarse (although the solution quality may be improved by lowering the network density), but can achieve excellent performance when the mesh resolution is sufficiently fine. However, on finer meshes, factors such as network density (which may need to be increased) as well as stochasticity built into the neural network algorithms can lead to error saturation. Nevertheless, we feel that the meshfree nature of PINN and the ability of the network to implicitly approximate functions and their derivatives makes this methodology very appealing for computational solid and structural mechanics. Because of the multiple parameters involved in the construction of PINN and the inherent sensitivity of the PINN solution to these parameters, no universal guidelines can be suggested or quantitative conclusions can be made. Network design, at this stage, remains problem- and mesh-dependent and also requires the understanding of the mechanics and physics of the problem.

PINN remains an active research area in scientific machine learning, where studies are seeking to improve computational efficiency through domain decomposition XPINN (Jagtap and Karniadakis, 2021), parallel computing PINN (Shukla et al., 2021), and accelerating the convergence of the training process of PINN through Fourier random features (Wang et al., 2021b) and residual-based adaptive sampling (Wu et al., 2023). We have used the Fourier random features in our study since it is suitable for our problem and accelerates the convergence of the PINN. Active research is also being conducted to expand the utility of PINN beyond well-defined PDEs. Physics-Informed DeepONet (Wang et al., 2021a) expands solving an instance of PDEs into a class of PDEs, Physics-Informed GAN (Yang et al., 2020) is for stochastic differential equations, and Bayesian PINN (Yang et al., 2021) incorporate uncertainty into PINN.

Lastly, it is worth noting that inverse problems, such as material or geometry identification, are excellent candidates to harvest the full potential of PINN. Traditionally, these inverse problems are difficult to deal with using FEM and usually require additional external optimization algorithms. On the other hand, PINN formulation and network design for inverse problems can be obtained from the forward-problem design in a straightforward manner. Hence, integrating finite-deformation plasticity into the framework of PINN, with a complete forward-problem formulation presented in this work, we envision a promising path towards inverse-problem formulations in finite-strain elasto-plasticity and beyond.

CRediT authorship contribution statement

Sijun Niu: Investigation, Methodology, Formal analysis, Data curation, Software, Visualization, Writing – original draft. **Enrui Zhang:** Investigation, Methodology, Writing – review & editing. **Yuri Bazilevs:** Investigation, Methodology, Writing – review & editing. **Vikas Srivastava:** Conceptualization, Investigation, Methodology, Supervision, Funding acquisition, Writing – review & editing.

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.

Data availability

Data will be made available on request.

Acknowledgments

We gratefully acknowledge the support from the U.S. Department of Transportation Pipeline and Hazardous Materials Safety Administration (USDOT PHMSA) to V. Srivastava under grant numbers 693JK31950001CAAP and 693JK32050001CAAP. We also acknowledge the support from the Office of Naval Research (ONR), USA to Y. Bazilevs and V. Srivastava under grant number N00014-21-1-2670. The computations presented in this paper were carried out at Brown University's High Performance Computing facility — the Center for Computation and Visualization (CCV). The views and opinions expressed in this article are those of the authors and do not necessarily reflect the official policy or position of any agency of the U.S. government.

References

Abueidda, D.W., Koric, S., Al-Rub, R.A., Parrott, C.M., James, K.A., Sobh, N.A., 2022. A deep learning energy method for hyperelasticity and viscoelasticity. Eur. J. Mech. A Solids 95, 104639. http://dx.doi.org/10.1016/j.euromechsol.2022.104639.

Abueidda, D.W., Lu, Q., Koric, S., 2021. Meshless physics-informed deep learning method for three-dimensional solid mechanics. Internat. J. Numer. Methods Engrg. 122 (23), 7182–7201. http://dx.doi.org/10.1002/nme.6828.

Alipanahi, B., Delong, A., Weirauch, M.T., Frey, B.J., 2015. Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. Nature Biotechnol. 33 (8), 831–838. http://dx.doi.org/10.1038/nbt.3300.

Arora, R., Kakkar, P., Dey, B., Chakraborty, A., 2022. Physics-informed neural networks for modeling rate- and temperature-dependent plasticity. http://dx.doi.org/10.48550/ARXIV.2201.08363.

Bai, Y., Kaiser, N.J., Coulombe, K.L., Srivastava, V., 2021. A continuum model and simulations for large deformation of anisotropic fiber–matrix composites for cardiac tissue engineering. J. Mech. Behav. Biomed. Mater. 121, 104627. http://dx.doi.org/10.1016/j.jmbbm.2021.104627.

Baydin, A.G., Pearlmutter, B.A., Radul, A.A., Siskind, J.M., 2015. Automatic differentiation in machine learning: a survey. http://dx.doi.org/10.48550/ARXIV.1502.05767.

Bessa, M.A., Foster, J.T., Belytschko, T., Liu, W.K., 2014. A meshfree unification: Reproducing kernel peridynamics. Comput. Mech. 53 (6), 1251–1264. http://dx.doi.org/10.1007/s00466-013-0969-x.

Bonatti, C., Mohr, D., 2022. On the importance of self-consistency in recurrent neural network models representing elasto-plastic solids. J. Mech. Phys. Solids 158, 104697. http://dx.doi.org/10.1016/j.jmps.2021.104697.

Chaaba, A., 2010. Plastic collapse assessment of thick vessels under internal pressure according to various hardening rules. J. Pressure Vessel Technol. 132 (5), http://dx.doi.org/10.1115/1.4001272.

Chen, J.-S., Hillman, M., Chi, S.-W., 2017. Meshfree methods: Progress made after 20 years. J. Eng. Mech. 143 (4), http://dx.doi.org/10.1061/(asce)em.1943-7889.0001176.

Chen, Y., Lu, L., Karniadakis, G.E., Negro, L.D., 2020. Physics-informed neural networks for inverse problems in nano-optics and metamaterials. Opt. Express 28 (8), 11618–11633. http://dx.doi.org/10.1364/OE.384875.

Chester, S.A., Di Leo, C.V., Anand, L., 2015. A finite element implementation of a coupled diffusion-deformation theory for elastomeric gels. Int. J. Solids Struct. 52, 1–18. http://dx.doi.org/10.1016/j.ijsolstr.2014.08.015.

Eggersmann, R., Kirchdoerfer, T., Reese, S., Stainier, L., Ortiz, M., 2019. Model-free data-driven inelasticity. Comput. Methods Appl. Mech. Engrg. 350, 81–99. http://dx.doi.org/10.1016/j.cma.2019.02.016.

Fuhg, J.N., Bouklas, N., 2022. The mixed Deep Energy Method for resolving concentration features in finite strain hyperelasticity. J. Comput. Phys. 451, 110839. http://dx.doi.org/10.1016/j.jcp.2021.110839.

Goswami, S., Yin, M., Yu, Y., Karniadakis, G.E., 2022. A physics-informed variational DeepONet for predicting crack path in quasi-brittle materials. Comput. Methods Appl. Mech. Engrg. 391, 114587. http://dx.doi.org/10.1016/j.cma.2022.114587.

Gurtin, M.E., Fried, E., Anand, L., 2010. The Mechanics and Thermodynamics of Continua. Cambridge University Press, http://dx.doi.org/10.1017/CBO9780511762956.

Haghighat, E., Raissi, M., Moure, A., Gomez, H., Juanes, R., 2021. A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics. Comput. Methods Appl. Mech. Engrg. 379, 113741. http://dx.doi.org/10.1016/j.cma.2021.113741.

Hoerig, C., Ghaboussi, J., Insana, M.F., 2019. Data-driven elasticity imaging using cartesian neural network constitutive models and the autoprogressive method. IEEE Trans. Med. Imaging 38 (5), 1150–1160. http://dx.doi.org/10.1109/TMI.2018.2879495.

Hornik, K., Stinchcombe, M., White, H., 1989. Multilayer feedforward networks are universal approximators. Neural Netw. 2 (5), 359–366. http://dx.doi.org/10.1016/0893-6080(89)90020-8.

Huang, D., Fuhg, J.N., Weißenfels, C., Wriggers, P., 2020. A machine learning based plasticity model using proper orthogonal decomposition. Comput. Methods Appl. Mech. Engrg. 365, 113008. http://dx.doi.org/10.1016/j.cma.2020.113008.

Jagtap, A.D., Karniadakis, G.E., 2021. Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. In: AAAI Spring Symposium: MLPS.

Jin, H., Jiao, T., Clifton, R.J., Kim, K.S., 2022. Dynamic fracture of a bicontinuously nanostructured copolymer: A deep-learning analysis of big-data-generating experiment. J. Mech. Phys. Solids 164, http://dx.doi.org/10.1016/j.jmps.2022.104898.

Kadeethum, T., Jørgensen, T.M., Nick, H.M., 2020. Physics-informed neural networks for solving nonlinear diffusivity and Biot's equations. PLOS ONE 15 (5), 1–28. http://dx.doi.org/10.1371/journal.pone.0232683.

Kingma, D.P., Ba, J.L., 2015. Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings. pp. 1–15.

Kirchdoerfer, T., Ortiz, M., 2016. Data-driven computational mechanics. Comput. Methods Appl. Mech. Engrg. 304, 81–101. http://dx.doi.org/10.1016/j.cma.2016.02.001.

Kothari, M., Niu, S., Srivastava, V., 2019. A thermo-mechanically coupled finite strain model for phase-transitioning austenitic steels in ambient to cryogenic temperature range. J. Mech. Phys. Solids 133, 103729. http://dx.doi.org/10.1016/j.jmps.2019.103729.

Kutz, J.N., 2017. Deep learning in fluid dynamics. J. Fluid Mech. 814, 1–4. http://dx.doi.org/10.1017/jfm.2016.803.

Li, L.F., Chen, C.Q., 2022. Equilibrium-based convolution neural networks for constitutive modeling of hyperelastic materials. J. Mech. Phys. Solids 164, http://dx.doi.org/10.1016/j.jmps.2022.104931.

Liu, X., Athanasiou, C.E., Padture, N.P., Sheldon, B.W., Gao, H., 2020. A machine learning approach to fracture mechanics problems. Acta Mater. 190, 105–112. http://dx.doi.org/10.1016/j.actamat.2020.03.016.

Lu, X., Xu, H., Zhao, B., 2018. Effect of hardening exponent of power-law hardening elastic-plastic substrate on contact behaviors in coated asperity contact. Materials 11 (10), http://dx.doi.org/10.3390/ma11101965.

Masi, F., Stefanou, I., Vannucci, P., Maffi-Berthier, V., 2021. Thermodynamics-based artificial neural networks for constitutive modeling. J. Mech. Phys. Solids 147, 104277. http://dx.doi.org/10.1016/j.jmps.2020.104277.

Mozaffar, M., Bostanabad, R., Chen, W., Ehmann, K., Cao, J., Bessa, M.A., 2019. Deep learning predicts path-dependent plasticity. Proc. Natl. Acad. Sci. 116 (52), 26414–26420. http://dx.doi.org/10.1073/pnas.1911815116.

Niu, S., Srivastava, V., 2022a. Simulation trained CNN for accurate embedded crack length, location, and orientation prediction from ultrasound measurements. Int. J. Solids Struct. 242, 111521. http://dx.doi.org/10.1016/j.ijsolstr.2022.111521.

Niu, S., Srivastava, V., 2022b. Ultrasound classification of interacting flaws using finite element simulations and convolutional neural network. Eng. Comput. 38 (5), 4653–4662. http://dx.doi.org/10.1007/s00366-022-01681-y.

Pang, G., Lu, L., Karniadakis, G.E., 2019. fPINNs: Fractional physics-informed neural networks. SIAM J. Sci. Comput. 41 (4), A2603–A2626. http://dx.doi.org/10.1137/18M1229845.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S., 2019. PyTorch: An imperative style, high-performance deep learning library. Adv. Neural Inf. Process. Syst. 32.

Raissi, M., Perdikaris, P., Karniadakis, G.E., 2019. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J. Comput. Phys. 378, 686–707. http://dx.doi.org/10.1016/j.jcp.2018.10.045.

Raissi, M., Yazdani, A., Karniadakis, G.E., 2020. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. Science 367 (6481), 1026–1030. http://dx.doi.org/10.1126/science.aaw4741.

Rao, C., Sun, H., Liu, Y., 2021. Physics-informed deep learning for computational elastodynamics without labeled data. J. Eng. Mech. 147 (8), 4021043. http://dx.doi.org/10.1061/(ASCE)EM.1943-7889.0001947.

Samaniego, E., Anitescu, C., Goswami, S., Nguyen-Thanh, V.M., Guo, H., Hamdia, K., Zhuang, X., Rabczuk, T., 2020. An energy approach to the solution of partial differential equations in computational mechanics via machine learning: Concepts, implementation and applications. Comput. Methods Appl. Mech. Engrg. 362, 112790. http://dx.doi.org/10.1016/j.cma.2019.112790.

Shukla, K., Di Leoni, P.C., Blackshire, J., Sparkman, D., Karniadakis, G.E., 2020. Physics-informed neural network for ultrasound nondestructive quantification of surface breaking cracks. J. Nondestruct. Eval. 39 (3), 1–20. http://dx.doi.org/10.1007/s10921-020-00705-1.

Shukla, K., Jagtap, A.D., Karniadakis, G.E., 2021. Parallel physics-informed neural networks via domain decomposition. J. Comput. Phys. 447, 110683. http://dx.doi.org/10.1016/j.jcp.2021.110683.

Srivastava, V., Buitrago, J., Slocum, S.T., 2011. Stress analysis of a cryogenic corrugated pipe. In: Proceedings of the ASME 2011 30th International Conference on Ocean, Offshore and Arctic Engineering. 3, American Society of Mechanical Engineers, pp. 411–422. http://dx.doi.org/10.1115/OMAE2011-49852.

Srivastava, V., Chester, S.A., Anand, L., 2010. Thermally actuated shape-memory polymers: Experiments, theory, and numerical simulations. J. Mech. Phys. Solids 58 (8), 1100–1124. http://dx.doi.org/10.1016/j.jmps.2010.04.004.

Stainier, L., Leygue, A., Ortiz, M., 2019. Model-free data-driven methods in mechanics: material data identification and solvers. Comput. Mech. 64 (2), 381–393. http://dx.doi.org/10.1007/s00466-019-01731-1.

Tancik, M., Srinivasan, P.P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J.T., Ng, R., 2020. Fourier features let networks learn high frequency functions in low dimensional domains. Adv. Neural Inf. Process. Syst. 1–24.

Tang, S., Zhang, G., Yang, H., Li, Y., Liu, W.K., Guo, X., 2019. MAP123: A data-driven approach to use 1D data for 3D nonlinear elastic materials modeling. Comput. Methods Appl. Mech. Engrg. 357, 112587. http://dx.doi.org/10.1016/j.cma.2019.112587.

Voce, E., 1948. The relationship between stress and strain for homogeneous deformations. J. Inst. Met. 74, 537–562.

Wang, S., Wang, H., Perdikaris, P., 2021a. Learning the solution operator of parametric partial differential equations with physics-informed DeepONets. Sci. Adv. 7 (40), eabi8605. http://dx.doi.org/10.1126/sciadv.abi8605.

Wang, S., Wang, H., Perdikaris, P., 2021b. On the eigenvector bias of Fourier feature networks: From regression to solving multi-scale PDEs with physics-informed neural networks. Comput. Methods Appl. Mech. Engrg. 384, 113938. http://dx.doi.org/10.1016/j.cma.2021.113938.

Wu, C., Zhu, M., Tan, Q., Kartha, Y., Lu, L., 2023. A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks. Comput. Methods Appl. Mech. Engrg. 403, 115671. http://dx.doi.org/10.1016/j.cma.2022.115671.

Yang, L., Meng, X., Karniadakis, G.E., 2021. B-PINNs: Bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data. J. Comput. Phys. 425, 109913. http://dx.doi.org/10.1016/j.jcp.2020.109913.

Yang, L., Zhang, D., Karniadakis, G.E., 2020. Physics-informed generative adversarial networks for stochastic differential equations. SIAM J. Sci. Comput. 42 (1), A292–A317. http://dx.doi.org/10.1137/18M1225409.

Yin, M., Zheng, X., Humphrey, J.D., Karniadakis, G.E., 2021. Non-invasive inference of thrombus material properties with physics-informed neural networks. Comput. Methods Appl. Mech. Engrg. 375, 113603. http://dx.doi.org/10.1016/j.cma.2020.113603.

Zhang, E., Dao, M., Karniadakis, G.E., Suresh, S., 2022. Analyses of internal structures and defects in materials using physics-informed neural networks. Sci. Adv. 8 (7), eabk0644. http://dx.doi.org/10.1126/sciadv.abk0644.

Zhang, D., Lu, L., Guo, L., Karniadakis, G.E., 2019. Quantifying total uncertainty in physics-informed neural networks for solving forward and inverse stochastic problems. J. Comput. Phys. 397, 108850. http://dx.doi.org/10.1016/j.jcp.2019.07.048.

Zhang, Y., Xiong, R., He, H., Pecht, M.G., 2018. Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries. IEEE Trans. Veh. Technol. 67 (7), 5695–5705. http://dx.doi.org/10.1109/TVT.2018.2805189.

Zhang, E., Yin, M., Karniadakis, G.E., 2020. Physics-informed neural networks for nonhomogeneous material identification in elasticity imaging. http://dx.doi.org/10.48550/ARXIV.2009.04525.

Zhong, J., Srivastava, V., 2021. A higher-order morphoelastic beam model for tubes and filaments subjected to biological growth. Int. J. Solids Struct. 233, 111235. http://dx.doi.org/10.1016/j.ijsolstr.2021.111235.

Zhu, Q., Liu, Z., Yan, J., 2021. Machine learning for metal additive manufacturing: predicting temperature and melt pool fluid dynamics using physics-informed neural networks. Comput. Mech. 67 (2), 619–635. http://dx.doi.org/10.1007/s00466-020-01952-9.