Physics-informed neural networks for modeling rate- and temperature-dependent plasticity

Rajat Arora*† Pratik Kakkar† Biswadip Dey† Amit Chakraborty†

Abstract

This work presents a physics-informed neural network based framework to model the strain-rate and temperature dependence of the deformation fields (displacement, stress, plastic strain) in elastic-viscoplastic solids. A detailed discussion on the construction of the physics-based loss criterion along with a brief outline on ways to avoid unbalanced back-propagated gradients during training is also presented. We also present a simple strategy with no added computational complexity for choosing scalar weights that balance the interplay between different terms in the composite loss. Moreover, we also highlight a fundamental challenge involving selection of appropriate model outputs so that the mechanical problem can be faithfully solved using neural networks. Finally, the effectiveness of the proposed framework is demonstrated by studying two test problems modeling the elastic-viscoplastic deformation in solids at different strain-rates and temperatures, respectively.

1 Introduction

Modeling the elastic-plastic response of materials using conventional numerical methods, such as finite element method, Isogeometric analysis, or mesh-free methods, has always been computationally expensive due to the inherent iterative nature of discretization algorithms used in such methods. Furthermore, multitude of 'fundamentally accurate' theories for the high-fidelity modeling of dislocation mediated plastic deformation at different scales [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11] or fracture modeling in materials [12, 13, 14, 15, 16], is bringing these numerical solvers to their limits. Therefore, attempts are being made to explore the use of artificial intelligence, specifically Deep Neural Networks (DNN), to speed up (nonlinear) mechanical modeling of materials. The recent flurry of research involving the use of DNN to model physical systems has also been facilitated by the notable speed enhancements in the current computer architecture systems, and availability of computationally efficient open-source machine learning frameworks (PyTorch [17], Tensorflow [18], and Keras [19] among others).

*Corresponding author: rajat.arora@siemens.com.
†Siemens Corporation, Siemens Technology.

In fact, the idea of using the extraordinary approximating capability of DNN, established by the Universal approximation theorem [20], to obtain the solution of Partial Differential Equations (PDEs) by minimizing the network's loss function, comprising the residual error of governing PDEs and its initial/boundary conditions, has been around for some time [21, 22]. More recently, Raissi et. al [23, 24] has extended this concept to develop a general Physics-Informed Neural Network (PINN) framework for solving the forward and inverse problems involving general nonlinear PDEs by relying on small or even zero labeled datasets. Several applications of PINNs can be found in the literature ranging from modeling of fluid flows and Navier Stokes equations [25, 26, 27, 28], cardiovascular systems [29, 30], and material modeling [31, 32, 33, 34, 35, 36], among others. In addition to the PINN based approaches mentioned above, several data-driven approaches [37, 38, 39, 40] have also been proposed to generate surrogate models for modeling the solution of PDEs governing the behavior of physical systems. However, data-driven methods usually require extensive amount of experimentally or computationally generated data to learn a reliable model which may fail to satisfy physics-based constraints. On the other hand, the embedding of physics in the PINN based framework enforces physics-based constraints on the neural network outputs, thus enabling the generation of a high-fidelity model while simultaneously reducing (or even eliminating) the need for bigger training datasets.

Recurrent Neural Networks (RNN) and its variants Gated Recurrent Units [41] and Long Short-Term Memory (LSTM) [42] are another family of neural network models that are specifically designed to handle sequential data – making them an attractive choice to learn and predict (path-dependent) plastic behavior in metals. The presence of history-dependent hidden states in these models enable them to re-use relevant information from previous inputs to make future predictions which warrants their role similar to those of history-dependent internal variables (such as equivalent plastic strains) in computational plasticity. Various approaches using RNNs [43, 44, 45, 46, 47] have successfully demonstrated the predictive capability of neural networks to learn the path-dependent constitutive behavior in metals with great accuracy. These works showcase the potential of RNN based constitutive modeling to speedup existing numerical solvers by permitting direct evaluation of stresses at the integration points without the need for iterative return mapping algorithms. However, developing a physics-informed neural network to model the spatio-temporal variation of deformation in elastic-plastic solids, along with its dependence on strain-rate and temperature, poses several technical challenges. Through this work, we take a first step in highlighting these challenges and demonstrate the strength of PINNs for modeling elastic-viscoplastic deformation in materials.

The goal of the present work is to demonstrate the (physics-based) predictive capability of DNN models to model spatio-temporally varying deformation fields within elastic-plastic solids under monotonous loading conditions. In particular, two feedforward DNNs are used as the global approximator to study the dependence of spatio-temporally varying deformation fields (displacement, stress, and plastic strain) on strain rate (applied loading rate) and temperature, respectively. We provide several (but non-exhaustive) comparisons of the choice of the DNN architecture (number of hidden layers and number of neurons per layer) chosen to approximate the solution. A detailed discussion on the construction of (physics-based) composite loss along with a brief summary on ways to avoid unbalanced back-propagated

(exploding) gradients during model training is also presented. We also present a strategy with no added computational complexity for choosing the scalar weights that balance the interplay between different terms in composite loss. Although the current work focuses on the scenarios with monotonic loading paths, we note that the deformation of an elastic-viscoplastic solid is a highly nonlinear function of temperature, strain rate, spatial coordinates, and strain. This *real-time* stress predictive capability for elastic-viscoplastic materials enjoys special use in energy storage devices such as the design and development of lithium metal solid-state-batteries. Specifically, the study conducted here corresponds to analyzing the effect of impact (i.e. crash) and heat to the solid Li Anode in the solid state batteries.

The rest of this paper is organized as follows: After introducing notation and terminology immediately below, Sec. 2 presents a brief review of the equations for mechanical equilibrium for elastic-viscoplastic solids. The details of the neural network architecture and construction of the loss function are presented in Sections 3 and 4, respectively. Section 5 presents the results that evaluate the neural network models and demonstrate their predictive capabilities in modeling the temperature and strain rate dependence of deformation fields along with their spatio-temporal evolution for the two test cases discussed. Finally, conclusion of the current work, along with outlook of the future work, is presented in Sec. 6.

Notation and Terminology

Cartesian coordinate system is invoked for the ambient space and all (vector) tensor components are expressed with respect to the basis $(\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3)$ of this coordinate system. Vectors and tensors are represented by bold face lower- and upper-case letters, respectively. The symbol '$\cdot$' denotes single contraction of adjacent indices of two tensors (i.e. $\boldsymbol{a} \cdot \boldsymbol{b} = a_i b_i$ or $\boldsymbol{A} \cdot \boldsymbol{n} = A_{ij} n_j$). The symbol ':' denotes double contraction of adjacent indices of two tensors of rank two or higher (i.e. $\boldsymbol{A} : \boldsymbol{B} = A_{ij} B_{ij}$ or $\mathbb{C} : \boldsymbol{A} = \mathbb{C}_{ijkl} A_{kl}$). The following list describes the recurring mathematical symbols used in this paper.

Mathematical symbols	
$\mathbb{C}$	Fourth order elasticity tensor
$tr(\boldsymbol{\phi})$	trace of the vector/tensor quantity $\boldsymbol{\phi}$
$\|\|\boldsymbol{\phi}\|\|$	L_2 norm of the vector/tensor quantity $\boldsymbol{\phi}$
Div	Divergence operator
∇	Gradient operator
$\boldsymbol{I}$	Second order Identity tensor
$\boldsymbol{A}'$	Deviatoric part of a second order tensor $\boldsymbol{A}$ i.e. $\boldsymbol{A}' = \boldsymbol{A} - tr(\boldsymbol{A})\boldsymbol{I}$
$\boldsymbol{\sigma}$	Stress in the material
$\hat{\Gamma}$	Strain rate
t	Time
Γ	Scalar strain defined as $\hat{\Gamma} \times t$
Ω	Volumetric domain
$\boldsymbol{x} \equiv (x_1, x_2)$	Spatial coordinates in the domain Ω
θ	Temperature of the domain Ω
$\partial\Omega$	External boundary of the domain Ω

$\boldsymbol{n}$	External unit outward normal to $\partial\Omega$
$\boldsymbol{\epsilon}^e$	Elastic strain tensor
$\boldsymbol{\epsilon}^p$	Plastic strain tensor
$\dot{\boldsymbol{\epsilon}}^p$	Plastic strain rate tensor
$\boldsymbol{\epsilon}$	Total strain tensor
$\boldsymbol{u}$	Displacement vector
E	Young's Modulus
μ	Shear Modulus
ν	Poisson's ratio
$\dot{\bar{\epsilon}}^p$	Equivalent plastic strain rate
$\boldsymbol{t}_{bc}$	Traction vector on Neumann boundary $\partial\Omega_N$
$\boldsymbol{u}_{bc}$	Displacement vector on Dirichlet boundary $\partial\Omega_D$
A	Pre-exponential factor
Q	Activation energy
R	Gas constant
$m \in (0, 1]$	Strain-rate-sensitivity parameter
$\text{sign}(\cdot)$	Sign of the scalar quantity $(\cdot)$
S	Material strength
S_s	Saturation value of S for a given strain rate and temperature
$\{H_0, S_*, a, n\}$	Strain-hardening parameters
S_0	Initial $(t = 0)$ value of the material strength S

2 Governing equations

We briefly recall the system of nonlinear PDEs governing the behavior of elastic-viscoplastic solids under loads at small deformation. The reader is referred to standard textbooks [48] for a detailed discussion on the thermodynamics and mechanics of continuous media. The equilibrium equation, in the absence of body and inertial forces, is written as

$$Div\,\boldsymbol{\sigma} = \mathbf{0} \quad \text{in} \quad \Omega. \tag{1}$$

The above PDE, together with the boundary conditions

$$\boldsymbol{\sigma} \cdot \boldsymbol{n} = \boldsymbol{t}_{bc} \quad \text{on} \quad \partial\Omega_N \quad \text{and} \quad \boldsymbol{u} = \boldsymbol{u}_{bc} \quad \text{on} \quad \partial\Omega_D, \tag{2}$$

describe the strong form of mechanical equilibrium. The total strain $\boldsymbol{\epsilon}$ is given by the symmetric part of the displacement gradient, i.e. $\boldsymbol{\epsilon} = 0.5(\nabla\boldsymbol{u} + (\nabla\boldsymbol{u})^T)$. $\boldsymbol{\epsilon}$ is decomposed into the sum of elastic and plastic strain components, i.e. $\boldsymbol{\epsilon} = \boldsymbol{\epsilon}^e + \boldsymbol{\epsilon}^p$. The stress is given by the Hooke's law

$$\boldsymbol{\sigma} = \mathbb{C} : \boldsymbol{\epsilon}^e. \tag{3}$$

The plastic strain evolution is governed by

$$\dot{\boldsymbol{\epsilon}}^p = \sqrt{\frac{3}{2}}\,\dot{\bar{\epsilon}}^p\,\frac{\boldsymbol{\sigma}'}{||\boldsymbol{\sigma}'||}. \tag{4}$$

The equivalent plastic strain rate $\dot{\epsilon}^p$ is assumed to be given by a power law of the form

$$\dot{\epsilon}^p = A \exp\left(-\frac{Q}{R\theta}\right)\left(\sqrt{\frac{3}{2}}\frac{||\boldsymbol{\sigma}'||}{S}\right)^{1/m}. \tag{5}$$

The flow strength S is assumed to evolve according to

$$\dot{S} = h(S, \boldsymbol{\sigma}). \tag{6}$$

Following [49], the hardening function $h(S, \boldsymbol{\sigma})$ is taken as

$$h(S,\boldsymbol{\sigma}) = \left[H_0\left|1-\frac{S}{S_s}\right|^a \text{sign}\left(1-\frac{S}{S_s}\right)\right]\dot{\epsilon}^p, \quad \text{with}$$
$$S_s = S_*\left[\frac{\dot{\epsilon}^p}{A\exp\left(-\frac{Q}{R\theta}\right)}\right]^n.$$

In this work, we use the values of material parameters listed in Table 2 which correspond to an elastic-viscoplastic model for Lithium [49]. These parameters have been calibrated using the experimental data from direct tension tests on polycrystalline lithium specimens [50].

A s^{-1}	Q kJ/mol	m	S_0 MPa	H_0 MPa	S_* MPa	a	n	E MPa	ν
4.25×10^4	37	0.15	0.95	10.0	2.0	2.0	0.05	7810	0.38

Table 2: Material parameters for the elastic-viscoplastic model of Lithium.

3 Model Architecture

The central idea is to use a fully connected DNN to approximate the nonlinear mapping (for deformation fields) $\Psi : \boldsymbol{X} \to \boldsymbol{Y}$, where $\boldsymbol{X} \in \mathbb{R}^{N_{in}}$ and $\boldsymbol{Y} \in \mathbb{R}^{N_{out}}$ denote the N_{in}-dimensional input and N_{out}-dimensional output arrays for the mapping, respectively. In particular, we design and train two specific fully connected neural networks for modeling the elastic-viscoplastic behavior in two-dimensional solids and its dependence on strain rate, and temperature, respectively. To this end, we choose the inputs and the outputs of the model for the two cases before moving on to describing the model architecture.

Inputs of the model: The inputs ($N_{in} = 4$) for both the cases are described in the table below:

Case I : Strain rate dependence	$\{x_1, x_2, \Gamma, \hat{\Gamma}\}$
Case II : Temperature dependence	$\{x_1, x_2, \Gamma, \theta\}$.

Outputs of the model: The elastic-plastic deformation can be uniquely characterized by determining displacement vector $\boldsymbol{u}$ and any one of the two tensor fields, plastic strain $\boldsymbol{\epsilon}^p$ or

stress $\boldsymbol{\sigma}$ (since they are related by (3)), along with the determination of internal variables. Therefore, a preferred choice of outputs for the neural network model would include $\boldsymbol{u}$ and one of the two fields ($\boldsymbol{\epsilon}^p$ or $\boldsymbol{\sigma}$) plus internal variables. However, as demonstrated later in Appendix A, the DNN model with such choice of outputs suffers from degraded accuracy and convergence issues. Therefore, we propose a mixed-variable formulation, i.e., displacement, stress, plastic strain and the internal variables as the DNN outputs in this work. This formulation is found to be superior to the other formulation with regard to the trainability of the network as discussed later in appendix A. The model outputs for the proposed mixed-variable formulation for two-dimensional plane-strain conditions are $\{u_1, u_2, \sigma_{11}, \sigma_{22}, \sigma_{33}, \sigma_{12}, \epsilon^p_{11}, \epsilon^p_{22}, \epsilon^p_{12}, S\}$ implying $N_{out} = 10$. In doing so, we have used the facts that the stress tensor is symmetric and the plastic strain tensor is symmetric and deviatoric (i.e. $\text{tr}(\boldsymbol{\epsilon}^p) = 0$).

The hyper-parameters completely defining the neural network model comprise the number of hidden layers N_l, the number of neurons per layer N_n, and the activation function(s). The fully connected neural network algorithm representing the mapping Ψ also includes the following main elements:

- Normalization of the inputs: Each input component X_i is individually scaled between -1 and 1 before being used as the network input. The normalization is done by linearly projecting it within a range of $[-1, 1]$ as follows

$$\bar{X}_i = -1 + 2\frac{\psi_i(X_i) - \min(\psi_i(X_i))}{\max(\psi_i(X_i)) - \min(\psi_i(X_i))}, \tag{7}$$

  where ψ_i is a (known) transformation based on the physical concepts in the process of elastic-viscoplastic deformation. The use of such transformations have previously been shown to improve neural network training in terms of speed and accuracy [51, 45].

- Activation of the l^{th} hidden layer: $\boldsymbol{H}^{(l)}$, the output array of any hidden layer $1 \leq l \leq N_l$, is calculated as

$$\begin{aligned} \boldsymbol{H}^{(l)} &= \Phi^{(l)}\left(\boldsymbol{W}^{(l)}\boldsymbol{H}^{(l-1)} + \boldsymbol{b}^{(l)}\right) \\ \implies H^{(l)} &= \Phi^{(l)}\left(\sum_{k=1}^{N_{pre}} W^{(l)}_{ik} H^{(l-1)}_k + b^{(l)}_i\right), \quad 1 \leq i \leq N_n. \end{aligned} \tag{8}$$

  In the above, $\Phi^{(l)}$ denotes the activation function for the hidden layer l which operates elementwise on its arguments. $\boldsymbol{W}^{(l)}$ and $\boldsymbol{b}^{(l)}$ denote the weight matrix and bias array for the hidden layer l, respectively. N_{pre} denotes the number of the neurons in the layer $l-1$. For $l = 1$: $N_{pre} = N_{in}$ and $\boldsymbol{H}^{(0)} = \bar{\boldsymbol{X}}$

- Computation of the outputs: The normalized output array $\bar{\boldsymbol{Y}}$ is similarly calculated as

$$\bar{Y}_i = \left(\sum_{k=1}^{N_n} W^{(o)}_{ik} H^{(N_l)}_k + b^{(o)}_i\right), \quad 1 \leq i \leq N_{out}, \tag{9}$$

  $\boldsymbol{W}^{(o)}$ and $\boldsymbol{b}^{(o)}$ denote the weight matrix and bias for the output layer, respectively.

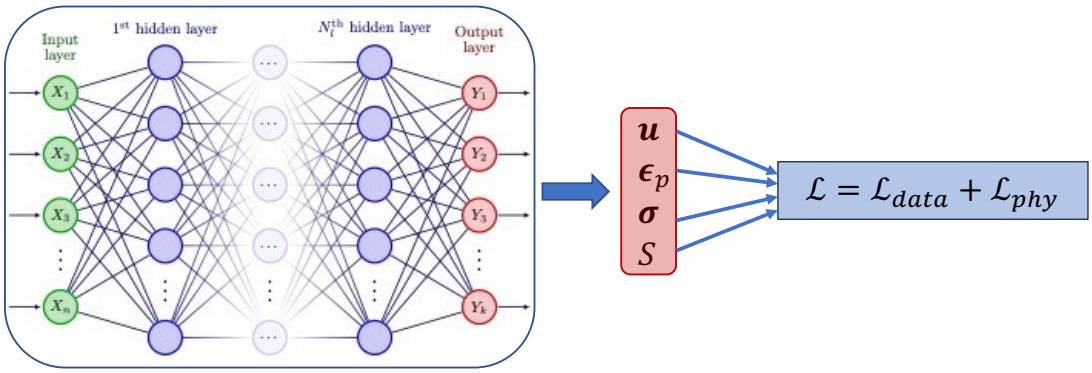

Figure 1: Schematic showing the geometry and the applied boundary conditions.

- Denormalization of the network outputs: Each component $\bar{Y}_i$ is then individually denormalized to yield the output array $\boldsymbol{Y}$ as

$$Y_i = \min(Y_i) + \frac{\bar{Y}_i + 1}{2} \left(\max(Y_i) - \min(Y_i)\right). \tag{10}$$

Henceforth, we denote the set of all weight matrices and bias arrays of the neural network by $\boldsymbol{W}$ and $\boldsymbol{b}$, respectively. In the present work, the hyperbolic tangent function is used as activation function for all the hidden layers, i.e. $\Phi^{(l)} = tanh \; \forall \; l \in [1, N_l]$. The transformation (see (7)) for all inputs is taken to be identity except for the strain rate, which is taken to be $\log_{10}$ function since strain rates may vary over multiple orders of magnitudes.

4 Construction of the loss function

The development of a PINN based approach to solve a system of nonlinear PDEs renders us an optimization problem which involves solving for $(\boldsymbol{W}, \boldsymbol{b})$ that minimizes network's total loss. The composite loss $\mathcal{L}$ in semi-supervised approach comprises the summation of supervised data loss $\mathcal{L}_{data}$ and the physics-based loss $\mathcal{L}_{phy}$ i.e. $\mathcal{L} = \mathcal{L}_{data} + \mathcal{L}_{phy}$. The (nondimensional) supervised data loss $\mathcal{L}_{data}$ measures the discrepancy between the normalized ground truth data $\bar{\boldsymbol{T}}$ and the neural network outputs $\bar{\boldsymbol{Y}}$ and is given by

$$\mathcal{L}_{data} = \frac{1}{N_{data}} \sum_{j=1}^{N_{data}} \sum_{i=1}^{N_{out}} ||\bar{Y}_i^{(j)} - \bar{T}_i^{(j)}||^2, \tag{11}$$

where N_{data} denotes the number of ground truth samples.

To evaluate the physics-based loss $\mathcal{L}_{phy}$, we introduce a finite set of randomly distributed collocation points discretizing the (normalized) input space. The whole set of collocation points is denoted by $\mathcal{P} = \{\mathcal{P}_\omega, \mathcal{P}_{\zeta,D}, \mathcal{P}_{\zeta,N}, \mathcal{P}_\eta\}$ where $\mathcal{P}_\omega$ denotes the collocation points in the entire input space $[-1, 1]^{N_{in}}$. $\mathcal{P}_{\zeta,D}$ and $\mathcal{P}_{\zeta,N}$ denote the subset of $\mathcal{P}$ that intersects with the $\partial\Omega_D$ and $\partial\Omega_N$, respectively. $\mathcal{P}_\eta$ denotes the subset of $\mathcal{P}$ that intersects with $\Gamma = 0$ (or $t = 0$).

To this end, we construct the physics-based loss $\mathcal{L}_{phy}$ with seven components i) PDE loss $\mathcal{L}_{pde}$, ii) Dirichlet boundary condition loss $\mathcal{L}_{Dbc}$, iii) Neumann boundary condition loss $\mathcal{L}_{Nbc}$, iv) initial condition loss $\mathcal{L}_{ic}$, v) constitutive loss $\mathcal{L}_c$ corresponding to the satisfaction of constitutive law, vi) plastic strain rate loss $\mathcal{L}_{psr}$ corresponding to the equation governing the evolution of plastic strain, and vii) strength loss $\mathcal{L}_s$ enforcing the material strength evolution equation. Each component of $\mathcal{L}_{phy}$ is individually calculated as follows:

$$
\begin{aligned}
\mathcal{L}_{pde} &= ||Div\,\boldsymbol{\sigma}||^2_{\mathcal{P}_\omega} \quad ; \quad \mathcal{L}_{Dbc} = \; ||\boldsymbol{u} - \boldsymbol{u}_{bc}||^2_{\mathcal{P}_{\zeta,D}} \\
\mathcal{L}_{Nbc} &= ||\boldsymbol{\sigma}\cdot\boldsymbol{n} - \boldsymbol{t}_{bc}||^2_{\mathcal{P}_{\zeta,N}} \quad ; \quad \mathcal{L}_{ic} = \; ||\boldsymbol{Y} - \mathcal{I}_0||^2_{\mathcal{P}_\eta} \\
\mathcal{L}_c &= ||\boldsymbol{\sigma} - \mathbb{C} : (\nabla\boldsymbol{u} - \boldsymbol{\epsilon}^p)||^2 \quad ; \quad \mathcal{L}_s = \; ||\dot{S} - h(S,\boldsymbol{\sigma})||^2 \\
\mathcal{L}_{psr} &= \text{MMSE}\left(\dot{\boldsymbol{\epsilon}}_p - \sqrt{\frac{3}{2}}\,A\exp\left(-\frac{Q}{R\theta}\right)\left(\sqrt{\frac{3}{2}}\frac{||\boldsymbol{\sigma}'||}{S}\right)^{1/m}\frac{\boldsymbol{\sigma}'}{||\boldsymbol{\sigma}'||}\right)
\end{aligned}
\tag{12}
$$

In the above, $\mathcal{I}_0$ denotes the initial state of the system i.e. outputs at $t = 0$. The loss criterion MMSE is discussed in detail below. $\mathcal{L}_{phy}$ is then given as the weighted sum of these loss components

$$
\mathcal{L}_{phy} = \lambda_1\mathcal{L}_{pde} + \lambda_2\mathcal{L}_{ic} + \lambda_3\mathcal{L}_{psr} + \lambda_4\mathcal{L}_s + \lambda_5\mathcal{L}_c + \lambda_6\mathcal{L}_{Dbc} + \lambda_7\mathcal{L}_{Nbc},
$$

where $\lambda_i (i = 1...6)$ are the scalar weights.

Next, we briefly discuss the two main difficulties that hinder the training of DNNs for elastic-viscoplastic modeling applications.

1. The power law dependence of the equivalent plastic strain rate $\dot{\epsilon}^p$ (Eq. (5)) leads to large values of L^2 norm of $\mathcal{L}_{psr}$ loss ($\geq O(10^{18})$) which causes unstable imbalance in the magnitude of the back-propagated gradients during the training when using common loss criterions such as Mean-Squared-Error (MSE). Therefore, in this work, we use a novel Modified Mean Squared Error (MMSE) loss criterion to reduce the numerical stiffness associated with equation (5) and allow stable gradients to be used during the training

$$
\text{MMSE}(A) = log_{10}(1 + ||A||). \tag{13}
$$

   In the above, A denotes the residual value. The loss criterion is equivalent to the Mean Squared Error (MSE) criterion when the discrepancy between the residual values are small.

2. The relative coefficients λ_i $(i = 1..6)$ for all the losses comprising $\mathcal{L}_{phy}$ play an important role in mitigating the gradient pathology issue during the training [52]. There are competing effects between these different loss components which can lead to convergence issues during the minimization of the composite loss $\mathcal{L}$ (see [25, Sec. 4.1]). While the recent advances in mitigating gradient pathologies [52, 53] might improve predictive accuracy, they introduce additional computational and memory overhead because of the calculation of an adaptive factor for each loss component. In this work, we devise a simple strategy, with no added computational complexity, to evaluate the coefficients which remain constant during the course of training. The strategy is outlined as follows:

- The Dirichlet boundary condition and initial condition losses ($\mathcal{L}_{ic}$ and $\mathcal{L}_{Dbc}$) are calculated in a normalized manner (scaled between $[-1, 1]$). So, we take $\lambda_2 = \lambda_6 = 1$.
- The other loss components are nondimensionalized using appropriate scales as shown in Table 3. μ_c is a constant chosen to scale quantities with units of stress. Based on the observation that stress is often nondimensionalized by Shear Modulus μ in conventional numerical methods, we choose $\mu_c = 0.01\mu$ to achieve tight tolerance on the equilibrium equation and traction boundary conditions.
- Since material strength S and μ differ by orders of magnitude, S is nondimensionalized by S_0.
- We nondimensionalize time by using strain rate $\hat{\dot{\Gamma}}$, since $\hat{\dot{\Gamma}}$ sets the time scale for the problem.
- The length is nondimensionalized by the characteristic length of the domain, chosen to be H in this work.

Loss component	Scaling
$\mathcal{L}_{pde}$	$\lambda_1 = \frac{H^2}{{\mu_c}^2}$
$\mathcal{L}_{ic}$	$\lambda_2 = 1$
$\mathcal{L}_{psr}$	$\lambda_3 = \frac{1}{\hat{\dot{\Gamma}}^2}$
$\mathcal{L}_{s}$	$\lambda_4 = \frac{1}{(S_0\,\hat{\dot{\Gamma}})^2}$
$\mathcal{L}_{c}$	$\lambda_5 = \frac{1}{{\mu_c}^2}$
$\mathcal{L}_{Dbc}$	$\lambda_6 = 1$
$\mathcal{L}_{Nbc}$	$\lambda_7 = \frac{1}{{\mu_c}^2}$

Table 3: Scaling constants for different physics-based loss components.

4.1 Training the network

The neural network is implemented and trained using PyTorch framework [17]. Before the training, the trainable parameters $(\boldsymbol{W}, \boldsymbol{b})$ of the neural network are initialized using Xavier initialization [54] technique. Adam optimizer [55] is used as the optimization algorithm with an initial learning rate $\eta = 10^{-3}$ with the other hyper-parameters set to their default value. We continue training for around 8000 epochs (complete passes through the whole training dataset) during which the learning rate is monitored and adaptively reduced by using `ReduceLROnPlateau` scheduler with the value of `patience` set to 30. The dataset collection strategy, and splitting into the training/validation/test datasets are discussed in greater detail in Section 5.

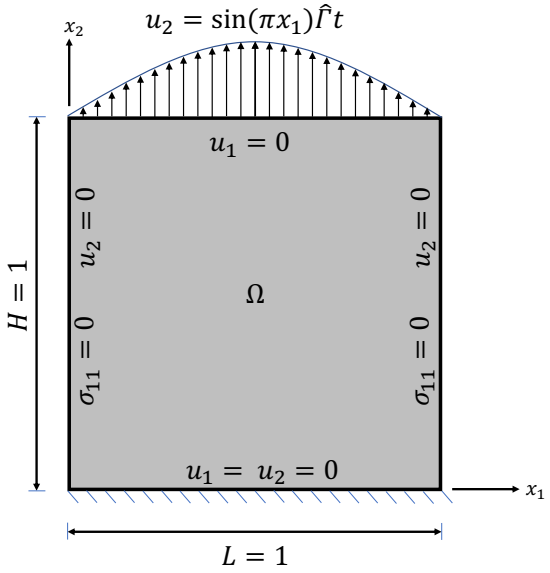

Figure 2: Schematic showing the geometry and the applied boundary conditions.

5 Results & Discussion

To illustrate the application of the proposed approach, we design and train specific neural networks for the two test cases focusing on reproducing the complex behavior of an elastic-viscoplastic material under loads along with its dependence on a) strain rate and b) temperature. Fig. 2 shows a schematic of the body along with the applied boundary conditions for both the cases. Without loss of generality, the body is assumed to deform quasistatically under plane strain conditions in the absence of body forces. An in-house code is developed using deal.II [56] to generate the ground truth data by solving the system of equations (1) - (6) up to strain $\Gamma = 0.04$. The code uses Finite Element Method with bi-linear elements on a grid size of 32×32. To qualitatively assess the accuracy of the predictions, we define Root-Mean-Squared Error (RMSE) e as

$$e(f) = \sqrt{\frac{\sum_{j=1}^{M} ||f_{pred}^{j} - f_{ref}^{j}||^2}{\sum_{j=1}^{M} ||f_{ref}^{j}||^2}}. \tag{14}$$

where f is the (normalized) quantity of interest and $M = 1024\,(= 32 \times 32)$ denotes the number of reference points in the spatial domain. The averaged-RMSE $\mathcal{E}$ over all output quantities is then defined as

$$\mathcal{E} = \frac{1}{N_{out}} \sum_{i=1}^{N_{out}} e(f_i). \tag{15}$$

In the following, a neural network architecture with N_l number of layers and N_n number of neurons per layer will be referred by $N_l \times N_n$.

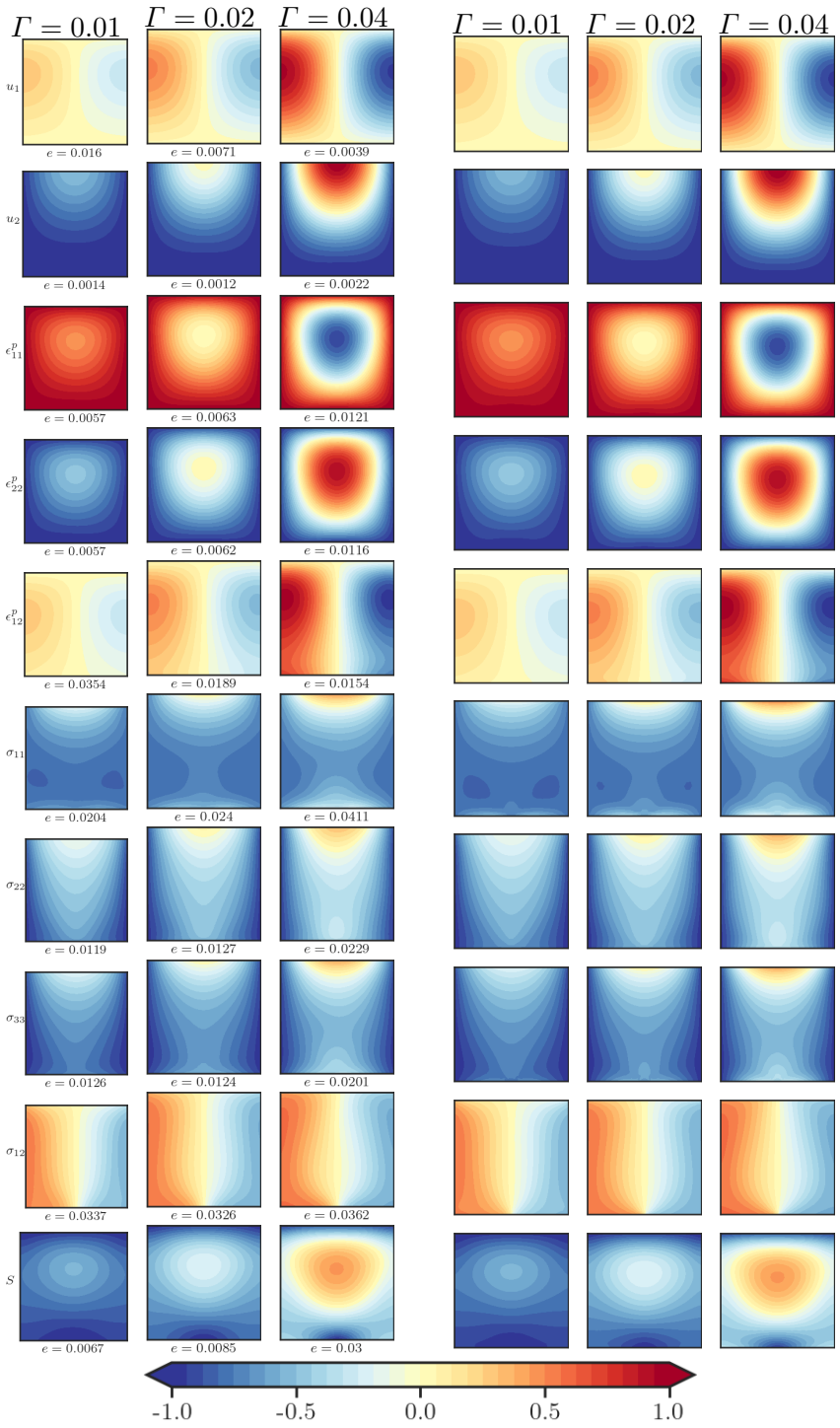

Figure 3: Comparison of the results obtained from physics-informed neural network model (left block) with the ground truth reference data (right block) for $\hat{\Gamma} = 10^{-2.5}\, s^{-1}$.

5.1 Case I: Strain rate dependence

Data collection strategy: To study the effect of strain rate $\hat{\Gamma}$ on the spatio-temporal evolution of deformation fields in the body, we generate the numerical data at the centroid of each

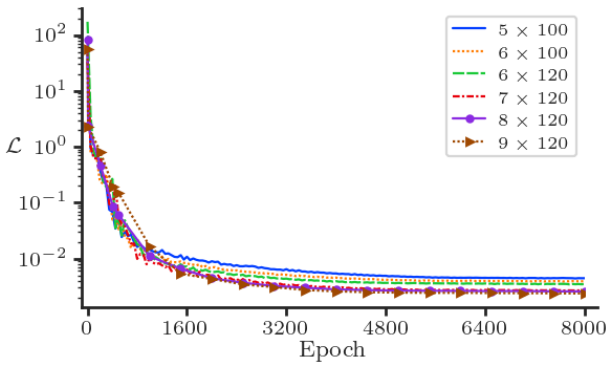

Figure 4: Comparison of the composite loss $\mathcal{L}$ for different neural network architectures ($N_l \times N_n$). N_l and N_n denotes the number of layers and number of nodels per layer, repectively.

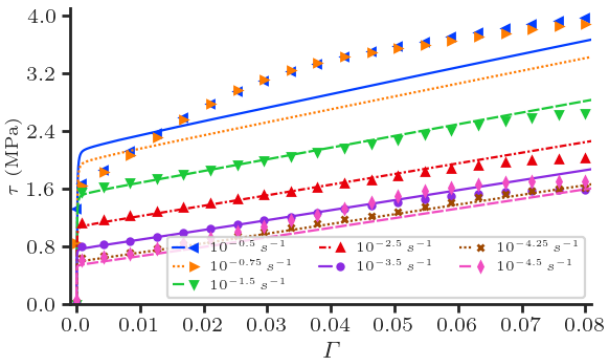

Figure 5: Stress-strain responses for different strain rates in and outside of the training data range.

grid element for the following strain rates $\hat{\dot{\Gamma}}(s^{-1})$: $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$ at a constant temperature $T = 298K$ upto strain $\Gamma = 0.04$. The dataset is then randomly split into a $80:20:0$ ratio for training, validation, and test purposes. We note that the testing data is set to zero as the model testing will be performed separately for several strain rates $\hat{\dot{\Gamma}}$ that are different from those used for generating the training data.

First, we perform a (non-exhaustive) parametric study to identify a suitable number of hidden layers N_l and number of neurons per layer N_n needed to model the deformation field with an acceptable accuracy. We train 6 neural networks with the following architectures: i) 5×100, ii) 6×100, iii) 6×120, iv) 7×120, v) 8×120, and vi) 9×120. Figure 4 presents the training history for each of these architectures. As expected, we see a merit in increasing both N_n and N_l initially but the final value of the composite loss stops improving when the number of layers are increased from 7 to 9 keeping N_n fixed at 120. These three network architectures ($7 \times 120, 8 \times 120$, and 9×120) reduce the nondimensional composite loss by almost five orders of magnitude (from 10^2 to $\sim 10^{-3}$). The values of the corresponding validation losses are monitored to notice any overfitting issues. We use the neural network with architecture 9×120 for results presented in this subsection.

Next, we compare the predicted values of the stress, plastic strain, and displacement fields in the domain with a test dataset for $\hat{\dot{\Gamma}} = 10^{-2.5} s^{-1}$ at 3 different strains i.e. at $\Gamma = 0.01, 0.02$, and 0.04. Figure 3 presents the predicted values of these deformation field components (left block) along with the FEM reference solution (right block). We can observe that the predicted values have no visible artifacts and are in great agreement with the FEM reference results. We also calculate the value of the RMSE e (see (14)) for each output quantity and report it underneath the corresponding field plot. The small values of e further confirm our observation that the neural network predictions match the FEM reference results remarkably well.

Next, we test the predictive capabilities of the trained neural network for values of inputs that lie outside of the subspace spanned by the training data. Specifically, we calculate the

averaged-RMSE (see (15)) $\mathcal{E}$ for 7 different strain rate values $\hat{\dot{\Gamma}}(s^{-1}) = \{10^{-0.5}, 10^{-.75}, 10^{-1.5}, 10^{-2.5}, 10^{-3.5}, 10^{-4.25}, 10^{-4.5}\}$ and multiple strain values in the range $[0, 0.08]$. We recall that training data spanned $\hat{\dot{\Gamma}} \in [10^{-1}, 10^{-4}]s^{-1}$ and $\Gamma \in [0, 0.04]$. Figure 7 shows the variation of the error $\mathcal{E}$ with strain Γ at different strain rates $\hat{\dot{\Gamma}}$. We make two important observations from this plot:

1. For the values of strain rate $\hat{\dot{\Gamma}}$ within the training range, the error $\mathcal{E}$ is very small ($\approx 1\%$) upto strain $\Gamma = 0.04$. However, in the region $\Gamma \in (0.04, 0.08]$, the error $\mathcal{E}$ steadily increases to $\approx 10\%$.

2. For values of $\hat{\dot{\Gamma}}$ outside of the training data range, the error $\mathcal{E}$ is large at all strains which implies that the predicted values do not match well with the actual FEM reference data.

Both these observations are consistent with the obtained stress-strain responses plotted in Figure 5. The stress-strain response (τ vs. Γ) is obtained by plotting the total force on the top surface divided by its length.

We therefore conclude that a fully connected DNN is successfully able to learn the highly nonlinear dependence of the deformation fields on the applied strain rate along with their spatio-temporal evolution. The predictions match remarkably well with the ground truth reference data for inputs within the training data range. However, as expected, the accuracy of the predictions degrade for strains and strain rate outside the training data range.

5.2 Temperature dependence

Data collection strategy: To approximate the nonlinear dependence of the (spatio-temporally varying) deformation fields (stress, plastic strain and displacement) on temperature and strain, we generate the numerical data at the centroid of each grid element for the following temperatures (K): $\{298, 318, 358, 378\}$ at a constant strain rate $\hat{\dot{\Gamma}} = 10^{-1}s^{-1}$ upto strain $\Gamma = 0.04$. In the similar spirit as before (see Sec. 5.1), the data set is split into $80 : 20 : 0$ ratio for training, validation, and test purposes. Test dataset size is set to zero as the model testing will be performed separately for several temperatures T that are different from those used during model training.

As before, we first conduct a study to gain insight into the effect of N_l and N_n on the composite loss $\mathcal{L}$ and train the six aforementioned neural network architectures (see Sec. 5.1). Figure 8 presents the training history for each of these architectures which shows similar trend as in Figure 4. Therefore, we use the neural network with architecture 9×120 for results presented in this subsection.

Next, we compare the predicted values of the stress, plastic strain, and displacement fields in the domain with a test dataset for $T = 328K$ at 3 different strains i.e. at $\Gamma = 0.01, 0.02$, and 0.04. Figure 3 presents the predicted values of these deformation field components (left block) along with the FEM reference solution (right block). We can observe that the error e (see (14)) has small values and the predictions are in great agreement with the FEM reference results.

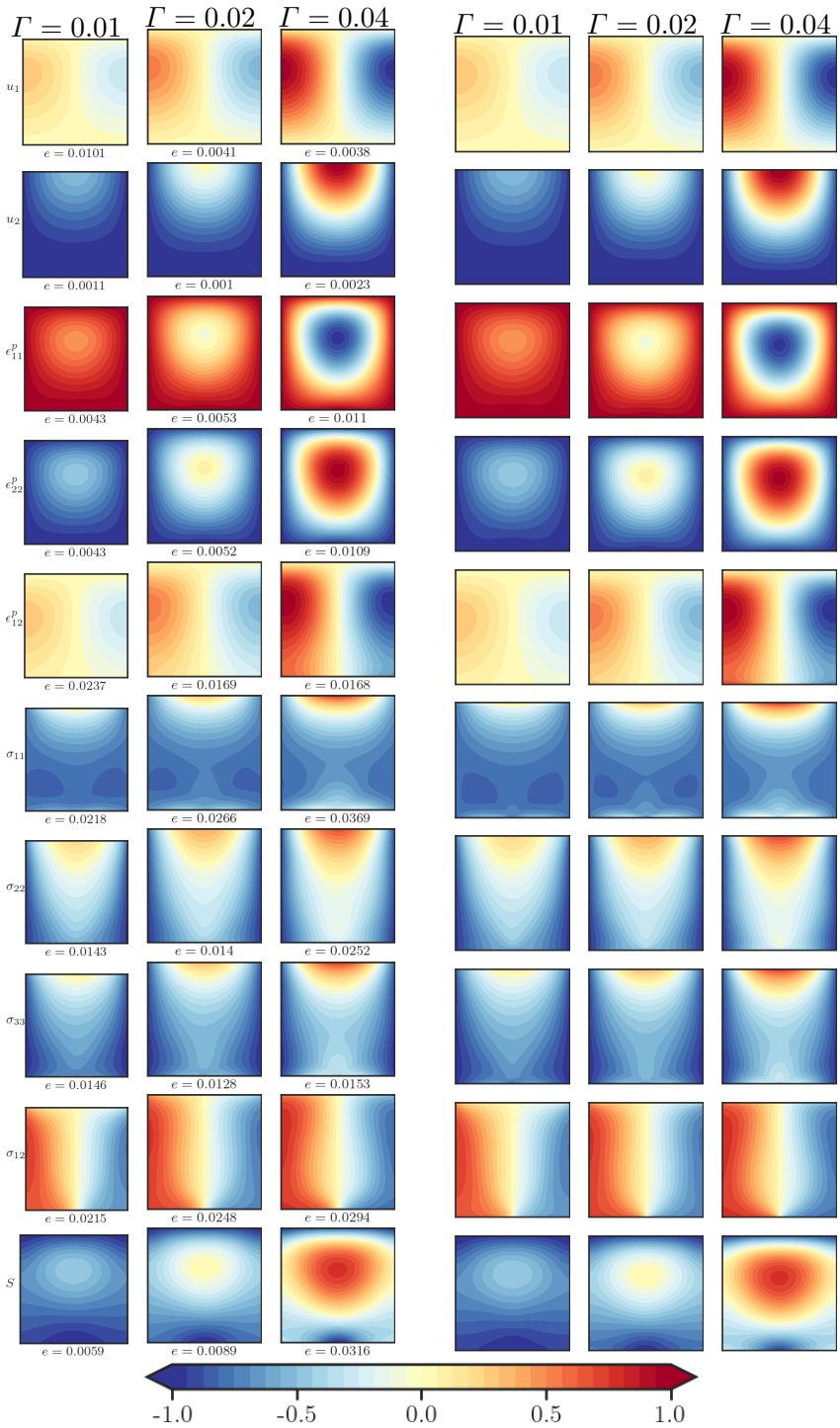

Figure 6: Comparison of the results obtained from physics-informed neural network model (left block) and the ground truth reference data (right block) for $T = 328K$.

Figure 9 shows the variation of the error $\mathcal{E}$ with strain Γ for different temperatures. We note that the error rises to $\geq 10\%$ as the strains go beyond the training data range. On the other hand, for values of temperature outside the training data range, the errors are still

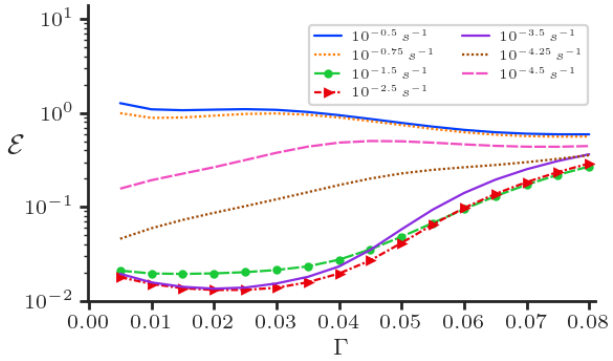

Figure 7: Variation of error $\mathcal{E}$ with strain Γ for different strain rates in and outside of the training data range.

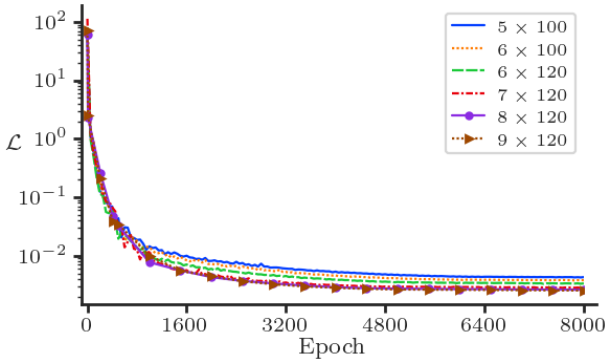

Figure 8: Comparison of the composite loss $\mathcal{L}$ for different neural network architectures $(N_l \times N_n)$. N_l and N_n denotes the number of layers and number of nodels per layer, repectively.

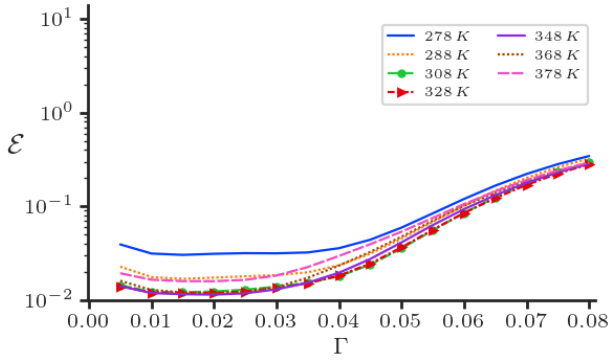

Figure 9: Variation of error $\mathcal{E}$ with strain Γ for different temperatures in and outside of the training data range.

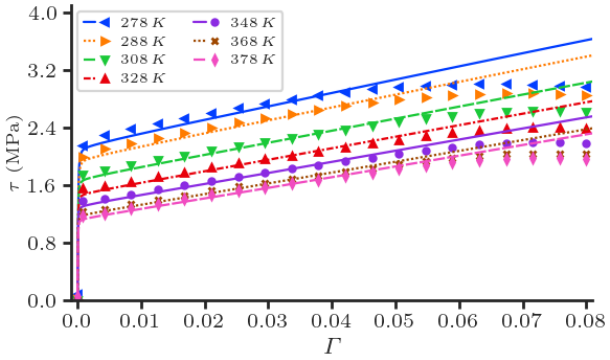

Figure 10: Stress-strain responses for different temperatures in and outside of the training data range.

$< 10\%$ which is in contrast with the error trend observed in Figure 7 when strain-rate $\hat{\Gamma}$ was outside the training data range. The stress-strain response plotted in Figure 10 further confirm these observations.

We therefore conclude that a fully connected neural network is successfully able to learn the nonlinear dependence of the deformation field on the temperature along with their spatio-temporal evolution.

6 Conclusion

This work demonstrates the strength of physics-informed neural networks in the context of problems dealing with the evolution of highly nonlinear deformation field in elastic-viscoplastic materials under monotonous loading. In particular, we trained specific physics-informed neural networks and applied them to two test cases of modeling the spatio-temporally

varying deformation in elastic-viscoplastic materials at different strain rates, and temperatures, respectively. We obtained results that are in great agreement with the ground truth reference data for both the test cases discussed in this work.

This work also discusses the construction of composite loss function, comprising the data loss component and physics-based loss components, in great detail. We also discuss a novel physics-based strategy for evaluation of nondimensional scalar constants that weigh each component in the physics-based loss function without any added computational complexity. Moreover, a novel loss criterion for residual calculation corresponding to plastic strain rate equation is proposed to alleviate issues related to unbalanced back-propagated (exploding) gradients during model training.

We also highlighted a fundamental challenge involving selection of appropriate model outputs so that the mechanical problem can be faithfully solved using neural networks. We present and compare two potential choice of outputs for the model in Appendix A and present detailed reasoning for preferring one choice over the other. The real-time stress field prediction in such highly nonlinear mechanical system paves the way for many new applications, such as design and optimization of lithium ion batteries or inverse modeling problems which were previously computationally intractable.

Future work will also focus on extending the framework to account for the path dependency of the loading by using recurrent architectures such as long short-term memory (LSTM) [42] and gated recurrent unit (GRU) [41] to detect history-dependent features. It would also be interesting to investigate the effect of enforcing boundary conditions in a *hard* manner [57] in the current framework.

Appendix A Comparison between two models

This section compares the results obtained from two PINN models a) Model I with displacement, stress, plastic strain, and strength $(\boldsymbol{u}, \boldsymbol{\sigma}, \boldsymbol{\epsilon}^p, S)$ as outputs and b) Model II with displacement, plastic strain, and stress $(\boldsymbol{u}, \boldsymbol{\epsilon}^p, S)$ as outputs. For Model II, the physics-based loss $\mathcal{L}_{phy}$ is obtained from the set of equations (12) with the following important changes: i) The stress is directly calculated from the displacements and plastic strains which are outputs of the neural network, i.e. $\boldsymbol{\sigma} = \mathbb{C} : (\nabla \boldsymbol{u} - \boldsymbol{\epsilon}^p)$. This implicitly leads to satisfaction of constitutive law so the loss component $\mathcal{L}_c$ is ignored. ii) The data loss $\mathcal{L}_{data}$ is also modified to account for the current model outputs.

The study conducted here corresponds to case I: Understanding the effect of strain rate on the spatio-temporal evolution of deformation in an elastic-viscoplastic material. The learning rate for Model II is taken to be 10^{-4} while keeping the collocation points and all other hyperparameters the same for both the architectures as described in Section 5.1.

The convergence of the training loss for both the models is presented in Fig. 11. It can be seen that loss reaches a stagnation value of $\approx$1.9 for model II at around 3000 epochs which is approximately hundred times larger than the converged loss value obtained for model I. We can conclude that model I does not suffer from any such degraded accuracy or convergence

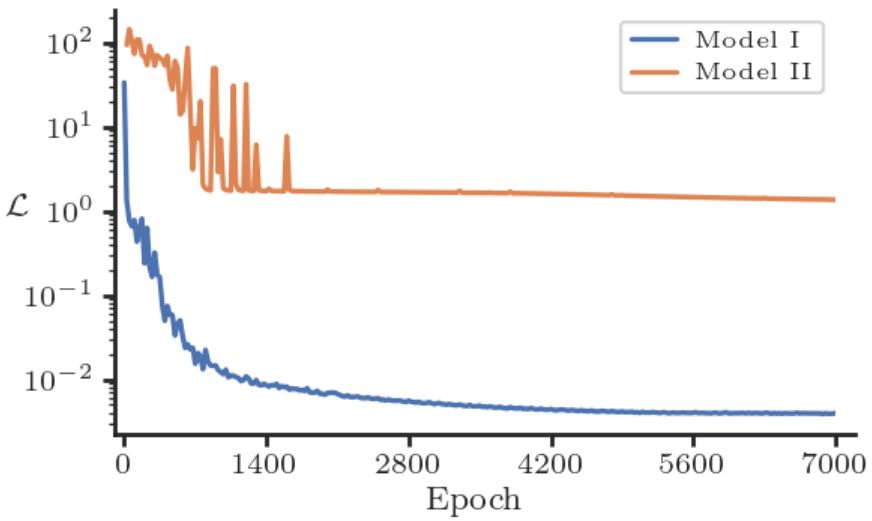

Figure 11: Comparison of training history for Model I and II differing only in the model outputs.

issue as indicated by Figure 11. This result is an extension of the similar observation for the purely linear elastic calculations presented in [57] to the general elastic-plastic modeling case discussed here.

While the exact reasons for such a behavior are still unclear, we highlight the main differences between the two models. First, the stress calculated in model II is sensitive to the noise in the gradients of $\boldsymbol{u}$. Second, we note that highest order of the spatial derivatives occurring in the composite loss function is one and two for models I and II, respectively. Moreover, in elastic/elastic-plastic deformations the order of displacement field magnitudes in the x_1 and x_2 direction can be vastly different because of the loading setup and Poisson's effect. We believe that these factors combine together to give rise to convergence issue and degraded accuracy when using model II. The use of improved training technique [58], which also approximates target derivatives along with target values, may alleviate these issues for model II but that may involve added computational complexity and remains the subject of future investigation.

References

[1] K. L. Nielsen and C. F. Niordson. A finite strain fe-implementation of the fleck-willis gradient theory: Rate-independent versus visco-plastic formulation. *European Journal of Mechanics / A Solids*, 75:389–398, 2019.

[2] Rajat Arora, Xiaohan Zhang, and Amit Acharya. Finite element approximation of finite deformation dislocation mechanics. *Computer Methods in Applied Mechanics and Engineering*, 367:113076, 2020.

[3] N. A. Fleck, G. M. Muller, M. F. Ashby, and J. W. Hutchinson. Strain gradient plasticity: Theory and experiment. *Acta Metallurgica et Materialia*, 42(2):475–487, 1994.

[4] Rajat Arora and Amit Acharya. A unification of finite deformation J2 Von-Mises plasticity and quantitative dislocation mechanics. *Journal of the Mechanics and Physics of Solids*, 143:104050, 2020.

[5] C. F. Niordson and V. Tvergaard. A homogenized model for size-effects in porous metals. *Journal of the Mechanics and Physics of Solids*, 123:222–233, 2019.

[6] J. Lynggaard, K. L. Nielsen, and C. F. Niordson. Finite strain analysis of size effects in wedge indentation into a face-centered cubic (fcc) single crystal. *European Journal of Mechanics / A Solids*, 76:193–207, 2019.

[7] M. Kuroda and V. Tvergaard. A finite deformation theory of higher-order gradient crystal plasticity. *Journal of the Mechanics and Physics of Solids*, 56(8):2573–2584, 2008.

[8] L.P. Evers, W.A.M. Brekelmans, and M.G.D. Geers. Non-local crystal plasticity model with intrinsic SSD and GND effects. *Journal of the Mechanics and Physics of Solids*, 52(10):2379–2401, 2004.

[9] Rajat Arora and Amit Acharya. Dislocation pattern formation in finite deformation crystal plasticity. *International Journal of Solids and Structures*, 184:114–135, 2020.

[10] A. L. Greer, Y. Q. Cheng, and E. Ma. Shear bands in metallic glasses. *Materials Science and Engineering: R: Reports*, 74(4):71–132, 2013.

[11] Rajat Arora. *Computational Approximation of Mesoscale Field Dislocation Mechanics at Finite Deformation*. PhD thesis, Carnegie Mellon University, 2019.

[12] Christian Miehe, Fabian Welschinger, and Martina Hofacker. Thermodynamically consistent phase-field models of fracture: Variational principles and multi-field fe implementations. *International journal for numerical methods in engineering*, 83(10):1273–1311, 2010.

[13] Michael J Borden, Thomas JR Hughes, Chad M Landis, and Clemens V Verhoosel. A higher-order phase-field model for brittle fracture: Formulation and analysis within the isogeometric analysis framework. *Computer Methods in Applied Mechanics and Engineering*, 273:100–118, 2014.

[14] Gao Yingjun, Luo Zhirong, Huang Lilin, Mao Hong, Huang Chuanggao, and Lin Kui. Phase field crystal study of nano-crack growth and branch in materials. *Modelling and Simulation in Materials Science and Engineering*, 24(5):055010, 2016.

[15] Pedro Areias, Timon Rabczuk, and MA3574977 Msekh. Phase-field analysis of finite-strain plates and shells including element subdivision. *Computer Methods in Applied Mechanics and Engineering*, 312:322–350, 2016.

[16] Charlotte Kuhn and Ralf Müller. A continuum phase field model for fracture. *Engineering Fracture Mechanics*, 77(18):3625–3634, 2010.

[17] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[18] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[19] François Chollet et al. Keras. https://keras.io, 2015.

[20] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[21] Isaac E Lagaris, Aristidis Likas, and Dimitrios I Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE transactions on neural networks*, 9(5):987–1000, 1998.

[22] Isaac E Lagaris, Aristidis C Likas, and Dimitris G Papageorgiou. Neural-network methods for boundary value problems with irregular boundaries. *IEEE Transactions on Neural Networks*, 11(5):1041–1049, 2000.

[23] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10561*, 2017.

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

[25] Luning Sun, Han Gao, Shaowu Pan, and Jian-Xun Wang. Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data. *Computer Methods in Applied Mechanics and Engineering*, 361:112732, 2020.

[26] Chengping Rao, Hao Sun, and Yang Liu. Physics-informed deep learning for incompressible laminar flows. *Theoretical and Applied Mechanics Letters*, 10(3):207–212, 2020.

[27] Xiaowei Jin, Shengze Cai, Hui Li, and George Em Karniadakis. Nsfnets (navier-stokes flow nets): Physics-informed neural networks for the incompressible navier-stokes equations. *Journal of Computational Physics*, 426:109951, 2021.

[28] Han Gao, Luning Sun, and Jian-Xun Wang. Phygeonet: Physics-informed geometry-adaptive convolutional neural networks for solving parametric pdes on irregular domain. *arXiv e-prints*, pages arXiv–2004, 2020.

[29] Georgios Kissas, Yibo Yang, Eileen Hwuang, Walter R Witschey, John A Detre, and Paris Perdikaris. Machine learning in cardiovascular flows modeling: Predicting arterial blood pressure from non-invasive 4d flow mri data using physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 358:112623, 2020.

[30] Francisco Sahli Costabal, Yibo Yang, Paris Perdikaris, Daniel E Hurtado, and Ellen Kuhl. Physics-informed neural networks for cardiac activation mapping. *Frontiers in Physics*, 8:42, 2020.

[31] Ari Frankel, Kousuke Tachida, and Reese Jones. Prediction of the evolution of the stress field of polycrystals undergoing elastic-plastic deformation with a hybrid neural network model. *Machine Learning: Science and Technology*, 1(3), July 2020.

[32] Ramakrishna Tipireddy, Paris Perdikaris, Panos Stinis, and Alexandre Tartakovsky. A comparative study of physics-informed neural network models for learning unknown dynamics and constitutive relations. *arXiv preprint arXiv:1904.04058*, 2019.

[33] Enrui Zhang, Minglang Yin, and George Em Karniadakis. Physics-informed neural networks for nonhomogeneous material identification in elasticity imaging. *arXiv preprint arXiv:2009.04525*, 2020.

[34] Xuhui Meng and George Em Karniadakis. A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse pde problems. *Journal of Computational Physics*, 401:109020, 2020.

[35] Qiming Zhu, Zeliang Liu, and Jinhui Yan. Machine learning for metal additive manufacturing: predicting temperature and melt pool fluid dynamics using physics-informed neural networks. *Computational Mechanics*, 67(2):619–635, 2021.

[36] Rajat Arora. Machine learning-accelerated computational solid mechanics: Application to linear elasticity. *arXiv preprint arXiv:2112.08676*, 2021.

[37] Daniele Versino, Alberto Tonda, and Curt A Bronkhorst. Data driven modeling of plastic deformation. *Computer Methods in Applied Mechanics and Engineering*, 318:981–1004, 2017.

[38] Zhiyong Li, Huaibao Zhang, Sean CC Bailey, Jesse B Hoagg, and Alexandre Martin. A data-driven adaptive reynolds-averaged navier–stokes k–ω model for turbulent flow. *Journal of Computational Physics*, 345:111–131, 2017.

[39] Francisco J Montáns, Francisco Chinesta, Rafael Gómez-Bombarelli, and J Nathan Kutz. Data-driven modeling and learning in science and engineering. *Comptes Rendus Mécanique*, 347(11):845–855, 2019.

[40] Steven L Brunton and J Nathan Kutz. Methods for data-driven multiscale model discovery for materials. *Journal of Physics: Materials*, 2(4):044002, 2019.

[41] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[42] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[43] M Mozaffar, R Bostanabad, W Chen, K Ehmann, Jian Cao, and MA Bessa. Deep learning predicts path-dependent plasticity. *Proceedings of the National Academy of Sciences*, 116(52):26414–26420, 2019.

[44] Diab W. Abueidda, Seid Koric, Nahil A. Sobh, and Huseyin Sehitoglu. Deep learning for plasticity and thermo-viscoplasticity. *International Journal of Plasticity*, 136:102852, 2021.

[45] Lahouari Benabou. Development of LSTM Networks for Predicting Viscoplasticity With Effects of Deformation, Strain Rate, and Temperature History. *Journal of Applied Mechanics*, 88(7), 05 2021. 071008.

[46] Dengpeng Huang, Jan Niklas Fuhg, Christian Weißenfels, and Peter Wriggers. A machine learning based plasticity model using proper orthogonal decomposition. *Computer Methods in Applied Mechanics and Engineering*, 365:113008, 2020.

[47] Maysam B. Gorji, Mojtaba Mozaffar, Julian N. Heidenreich, Jian Cao, and Dirk Mohr. On the potential of recurrent neural networks for modeling path dependent plasticity. *Journal of the Mechanics and Physics of Solids*, 143:103972, 2020.

[48] Morton E. Gurtin, Eliot Fried, and Lallit Anand. *The Mechanics and Thermodynamics of Continua*. Cambridge University Press, 2010.

[49] Lallit Anand and Sooraj Narayan. An elastic-viscoplastic model for lithium. *Journal of The Electrochemical Society*, 166(6):A1092, 2019.

[50] William S LePage, Yuxin Chen, Eric Kazyak, Kuan-Hung Chen, Adrian J Sanchez, Andrea Poli, Ellen M Arruda, MD Thouless, and Neil P Dasgupta. Lithium mechanics: roles of strain rate and temperature and implications for lithium metal batteries. *Journal of The Electrochemical Society*, 166(2):A89, 2019.

[51] R Kapoor, D Pal, and JK Chakravartty. Use of artificial neural networks to predict the deformation behavior of zr–2.5 nb–0.5 cu. *Journal of Materials Processing Technology*, 169(2):199–205, 2005.

[52] Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient pathologies in physics-informed neural networks. *arXiv preprint arXiv:2001.04536*, 2020.

[53] Rafael Bischof and Michael Kraus. Multi-objective loss balancing for physics-informed deep learning. *arXiv preprint arXiv:2110.09813*, 2021.

[54] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

[55] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *ICLR*, 2015.

[56] W. Bangerth, R. Hartmann, and G. Kanschat. deal.II – a general purpose object oriented finite element library. *ACM Trans. Math. Softw.*, 33(4):24/1–24/27, 2007.

[57] Chengping Rao, Hao Sun, and Yang Liu. Physics-informed deep learning for computational elastodynamics without labeled data. *Journal of Engineering Mechanics*, 147(8):04021043, 2021.

[58] Wojciech Marian Czarnecki, Simon Osindero, Max Jaderberg, Grzegorz Świrszcz, and Razvan Pascanu. Sobolev training for neural networks. *arXiv preprint arXiv:1706.04859*, 2017.