Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 363 (2020) 112875
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

SO(3)-invariance of informed-graph-based deep neural network for anisotropic elastoplastic materials

Yousef Heider[a,b], Kun Wang[a], WaiChing Sun[a,*]

[a] *Department of Civil Engineering and Engineering Mechanics, Columbia University, New York, NY 10027, USA*
[b] *Institute of General Mechanics, RWTH Aachen University, Templergraben 64, 52062 Aachen, Germany*

Received 13 July 2019; received in revised form 28 October 2019; accepted 21 January 2020
Available online xxxx

Abstract

This paper examines the frame-invariance (and the lack thereof) exhibited in simulated anisotropic elasto-plastic responses generated from supervised machine learning of classical multi-layer and informed-graph-based neural networks, and proposes different remedies to fix this drawback. The inherent hierarchical relations among physical quantities and state variables in an elasto-plasticity model are first represented as informed, directed graphs, where three variations of the graph are tested. While feed-forward neural networks are used to train path-independent constitutive relations (e.g., elasticity), recurrent neural networks are used to replicate responses that depends on the deformation history, i.e. or path dependent. In dealing with the objectivity deficiency, we use the spectral form to represent tensors and, subsequently, three metrics, the Euclidean distance between the Euler Angles, the distance from the identity matrix, and geodesic on the unit sphere in Lie algebra, can be employed to constitute objective functions for the supervised machine learning. In this, the aim is to minimize the measured distance between the true and the predicted 3D rotation entities. Following this, we conduct numerical experiments on how these metrics, which are theoretically equivalent, may lead to differences in the efficiency of the supervised machine learning as well as the accuracy and robustness of the resultant models. Neural network models trained with tensors represented in component form for a given Cartesian coordinate system are used as a benchmark. Our numerical tests show that, even given the same amount of information and data, the quality of the anisotropic elasto-plasticity model is highly sensitive to the way tensors are represented and measured. The results reveal that using a loss function based on geodesic on the unit sphere in Lie algebra together with an informed, directed graph yield significantly more accurate rotation prediction than the other tested approaches.

Keywords: Recurrent neural network; Lie algebra; Anisotropic materials; Machine learning; Crystal plasticity; Special orthogonal group

1. Introduction

Constitutive or material laws that provide the explicit relations among strain history, state variables and stress are the key component to supplement hard constraints such as balance principles and thermodynamic laws for

* Correspondence to: Department of Civil Engineering and Engineering Mechanics, Columbia University, 614 SW Mudd, Mail Code: 4709, New York, NY 10027, USA.
E-mail address: wsun@columbia.edu (W. Sun).

https://doi.org/10.1016/j.cma.2020.112875
0045-7825/

single-phase solid mechanics problems [1–5]. These constitutive laws are often proposed based on abstractions and generalizations of phenomenological and experimental observations [6–11].

However, the recent success in machine learning and mining on big data has become an impetus for an alternative data-driven approach, where these manual abstractions and generalization processes are either bypassed through a new optimization procedure that minimizes the distance between points in dataset and the constraint (equilibrium and compatibility) (cf. [1,2]) or replaced by supervised machine learning via neural networks or other artificial intelligence (AI) tools (e.g., support vector machine) [12–15]. In the latter case, our intention to generate constitutive laws to predict stress–strain relations must be mathematically represented as objective (or loss) function such that the artificial intelligence may evaluate the success of the training.

In this work, our objective is to (1) explore how and why seemingly equivalent objective functions but with different tensor representations (e.g. components vs. spectral form) for input and output may lead to orders of difference in efficiency of the neural network training and the accuracy of the resultant models for anisotropic elasto-plastic materials, and (2) how these differences accumulate in the multi-step informed-graph-based neural network and cause significant performance difference.

1.1. Tensor representation and metrics for 3D rotations

Recall that anisotropic materials may exhibit principal stresses non-coaxial to the principal strains. This non-coaxiality may originate from inherent anisotropy of the micro-structure (e.g., a stack of paper). It may also continuously evolve during deformation (e.g., the plasticity-induced spin due to activation of slip systems of different directions in crystal grain [16]), rotation of the fabric of granular assembles under inelastic deformation [17–19]. To capture these anisotropic responses, previous works, such as [13], have extended the training database by expressing the same set of stress–strain data in coordinate systems such that the discrepancy in different coordinate systems is suppressed. Recent studies, such as [14,20,21], have introduced coordinate-free invariant metrics for the objective function such that the resultant neural network model provides consistent predictions under arbitrary rotations. However, there has not been yet any work that explores how different parametrization of discrepancy measure affects the performance of the data-driven models. To analyze the influence of parametrization of machine learning objective functions, we first represent all tensor quantities involved in the supervised learning in spectral form, then employ different distance metrics for 3D rotation in numerical experiments. We use three scalar-valued functions suitable as metrics for measuring the distance between tensors belonging to the special orthogonal group, i.e., (1) the Euclidean distance between the Euler angles, (2) deviation of rotation matrices from the identity tensor, and (3) the geodesic on the unit sphere within Lie Algebra. Such metrics are widely used in robotic and computer graphic for measuring difference in orientation. Note that changing the objective function leads to different landscape in the parametric space. By employing the most common first-order gradient optimization approach, i.e. ADAM, we explore how changing the objective function affects the speed of the training, the convergence rate, and the robustness of finding the optimal material laws.

1.2. Deep neural network and informed, directed graph

Once an objective function is defined, a suitable machine learning strategy must be selected such that the supervised machine learning produces a mapping that maps input(s) to output(s) while maximizing the performance metrics defined by an objective (loss) function. This mapping can be generated by classical approaches, such as response surface [22], support vector machine [23], probabilistic learning [24], and neural network [25–27]. While different parametrizations of objective functions will affect the training procedure of all different types of supervised machine learning, our numerical experiments will be focusing on the deep neural network , one of the most common technique to generate generic constitutive laws in recent years. To deal with path-dependent material responses, we will use a combination of classical feed-forward neural network and the recurrent neural network. In this context, to circumvent the lack of interpretability of predictions made by the classical deep neural network, we incorporate the concepts from graph theory to introduce intermediate predictions that aim to better explain mechanism that leads to stress predictions [3,20,28,29].

Recall that a neural network is essentially an informed, directed graph in which neurons are the vertices connected by directed edges. In this graph approach, a prediction on stress is made by considering the information flow from

the root (strain history) of the graph to the leaf (stress), where weights of the neurons in-between the upstream and downstream are adjusted based on the training data. However, unlike the classical approach, where the information flow is merely passed through neurons in the intermediate layers that have no physical meaning, the informed, directed graph also introduce a second type of intermediate vertices that are themselves physical quantities [14,30]. Thus, the machine learning model is trained to predict intermediate quantities (e.g. such as activated slip systems, trial stresses or slip-system-related plastic deformation in crystal plasticity or the fabric tensor and porosity of the DEM simulation) that improve both the accuracy (by incorporating more information) and robustness (by introducing new basis) [18,31–33].

Other challenges related to the training or failure to finish training of the NN model, such as high demand of training data, under- or over-fitting and vanishing gradients were thoroughly addressed in [14]. Apart from this, Liu et al. [21], Liu and Wu [4] introduced a data-driven multiscale material modeling approach, referred to as deep material network, in 2D and 3D settings. The basic idea in this approach is to apply the mechanistic building block scheme for RVE model reduction, as well as to apply homogenization and direct numerical simulations via LS-Dyna on RVEs topology to generate training data. In the context of nonlinear, inelastic time-dependent material behavior, such as the response of metals under shock waves, Stoffel et al. [15] presented an ANN-based material model embedded into a finite element code. Within structural dynamics and the computation of response statistics, Koeppe et al. [34] proposed the inclusion of RNN to speed up Monte Carlo method by replacing the iterative calculation procedure used to evaluate the nonlinear, inelastic, hysteresis structural behavior. In connection with employing machine learning in the context of mechanical behavior of polycrystalline aggregates, Frankel et al. [35] recently developed a new approach that incorporate the pre-deformed grain morphology and orientation in the sense of image data and using convolutional neural network. Meanwhile, Huang et al. [27] introduced uncertainty quantification to neural network models designed for elasticity constitutive laws and the corresponding boundary value problems that employs neural network constitutive laws. The incorporation of the image data and the uncertainty quantification, see, e.g., [36], are important topics that will be considered in the future but is out of the scope of this study.

1.3. Major objectives, organization of contents, and notations

The goal of this research is to gain new insight on (1) how parametrization of objective functions affects the training and prediction performance of the neural network constitutive laws for anisotropic materials and (2) estimate the sensitivity and robustness of neural network constitutive laws incorporating different amounts of human knowledge in the informed, directed graph (e.g. complete black box, frame-indifference models, and physically-informed models), see, [14,37].

The organization of the rest of the paper is as follows. First, we will provide a brief overview of the construction of graph-based neural network model for general elasto-plasticity problem at material point (Section 2). Then, we introduce and compare metrics that seemingly provide identical physical meaning for the training of constitutive laws, explain the properties of these metrics and the corresponding implications on the neural network training (Section 3). The recurrent neural network for path-dependent predictions are highlighted in Section 4, followed by numerical experiments that examine the pros and cons for each of the metrics for machine learning in Section 5. The major approaches and findings are then summarized in Section 6. Throughout this paper, the single-grain crystal plasticity problem is used as the test bed.

As for notations and symbols, bold-faced letters denote tensors (including vectors which are rank-one tensors); the symbol '$\cdot$' denotes a single contraction of adjacent indices of two tensors (e.g. $\boldsymbol{a}\cdot\boldsymbol{b} = a_i b_i$ or $\boldsymbol{c}\cdot\boldsymbol{d} = c_{ij}d_{jk}$); the symbol ':' denotes a double contraction of adjacent indices of tensor of rank two or higher (e.g. $\boldsymbol{C} : \boldsymbol{\epsilon^e} = C_{ijkl}\epsilon^e_{kl}$); the symbol '$\otimes$' denotes a juxtaposition of two vectors (e.g. $\boldsymbol{a}\otimes\boldsymbol{b} = a_i b_j$) or two symmetric second order tensors (e.g. $(\boldsymbol{\alpha}\otimes\boldsymbol{\beta})_{ijkl} = \alpha_{ij}\beta_{kl}$). Moreover, $(\boldsymbol{\alpha}\oplus\boldsymbol{\beta})_{ijkl} = \alpha_{jl}\beta_{ik}$ and $(\boldsymbol{\alpha}\ominus\boldsymbol{\beta})_{ijkl} = \alpha_{il}\beta_{jk}$. We also define identity tensors $(\boldsymbol{I})_{ij} = \delta_{ij}$, $(\boldsymbol{I}^4)_{ijkl} = \delta_{ik}\delta_{jl}$, and $(\boldsymbol{I}^4_{\text{sym}})_{ijkl} = \frac{1}{2}(\delta_{ik}\delta_{jl} + \delta_{il}\delta_{kj})$, where δ_{ij} is the Kronecker delta. As for sign conventions, unless specified otherwise, we consider the direction of the tensile stress and dilative pressure as positive.

2. Overview of models and informed, directed graph representation

In the machine learning material models in this contribution, we distinguish the direct or "black box" approach from the graph-based "physical-informed" approach. In the "black box" case that employs the recurrent neural

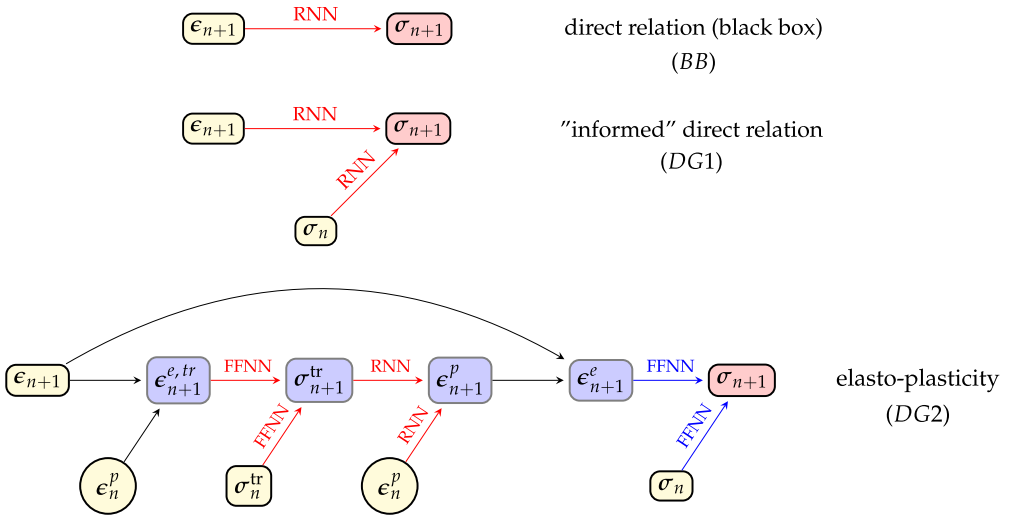

Fig. 1. Three variations of the informed, directed graph, representing the information flow in the machine learning-based crystal plasticity material model. The yellow nodes ($\boldsymbol{\epsilon}_{n+1}$, $\boldsymbol{\sigma}_n^{\text{tr}}$, $\boldsymbol{\epsilon}_n^p$ and $\boldsymbol{\sigma}_n$) represent the root nodes, the pink node ($\boldsymbol{\sigma}_{n+1}$) is the target node, whereas the blue nodes are the intermediate nodes. The black arrows represent definitions and the red arrows represent the machine learning models, which are either recurrent neural networks (RNN) or feed-forward regression neural networks (FFNN). The blue arrows in $DG2$ refer to ML models but without a new training. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

network, the total stress is predicted merely based on the total strain history. In the graph-based "physically-informed" approach, neural networks are making sequence of predictions not only on stress but also other intermediate quantities that can help making the stress predictions more accurate and robustness. In this latter case, we found that using feed-forward neural network (FFNN) to predict the path-independent responses, while use recurrent neural network (RNN) to predict the path-dependent behaviors may lead to higher accuracy of the stress predictions. In both case, we analyze (1) how representation (e.g. components vs. spectral form) of the tensors in the input, output, immediate vertices of the graph, and (2) how different ways to measure the discrepancies in the objective function affect the quality of the predictions.

We also discuss the representation of crystal plasticity model as an informed, directed graph, see, e.g. [38] for more discussion of the directed graph scheme. This way of representation is in line with the extraction of intermediate information (such as trial stresses and plastic strains in elasto-plasticity), which can be included as a part of the "physically-informed" machine learning approach [14]. Following this, the machine learning model can be thought as a procedure that employs organized knowledge and successive predictions that govern the final outcomes of predictions.

2.1. Representation of crystal plasticity in informed, directed graph

In this paper, we select the crystal plasticity problem as our test bed. Due to the existence of slip system, a single grain of crystal may exhibits isochoric plastic flow that depends on the combination of slip systems that are activation. Hence, the accuracy of the stress predictions highly depends on whether the machine learning model is able to replicate the plastic spin in each incremental step upon yielding. Note that, while the crystal plasticity is used as an example, the design of the directed graph is general such that it also work for other anisotropic elastoplastic materials where stress and elastic strain are not co-axial. In connection with the informed, directed graph representation, Fig. 1, top, illustrates the direct "black box" (BB) approach, in which RNN is applied to predict the next stress state $\boldsymbol{\sigma}_{n+1}$ based on the updated strain state $\boldsymbol{\epsilon}_{n+1}$ as well as the strain from previous time steps according to the RNN definition. In analogy to the traction-separation model discussed in [14], the crystal plasticity algorithm can be considered as an information flow in a directed graph. In this, two informed, directed graphs are discussed in the following with two levels of deepness: The first one, illustrated in Fig. 1, middle, represents an

"informed" direct relation ($DG1$) in which the predicted output history, here $\boldsymbol{\sigma}_n$, is used as an additional input, thus, contributing to the prediction of the next output $\boldsymbol{\sigma}_{n+1}$. This modification results in an incremental ANN, which is in line with the path-dependent material behavior subjected to time incremental loading. In this, all the attributes from previous time steps, such that of the strain and the stress tensors, are known. The second physically-informed approaches, $DG2$, illustrated in Fig. 1, bottom, is based on the classical elasto-plasticity computational framework [39,40] and does not incorporate lower-scale information.

Unlike $DG2$, both BB and $DG1$ graphs consider the overall behavior of the crystal without distinguishing between the elastic and the plastic responses in the training. However, $DG2$ variation hybridizes two machine-learning ANN approaches, i.e. RNN and FFNN, with the classical rate-independent computational elasto-plasticity model. In this, a number of the relations in the return-mapping algorithm are replaced by machine learning steps, where, similar to $DG1$, a time incremental framework is applied in $DG2$. Thus, having the time increment $\Delta t = t_{n+1} - t_n$ with t_{n+1} as the next time step and t_n as the current one, the root nodes are the strain tensor $\boldsymbol{\epsilon}_{n+1}$, the elastic trial strain $\boldsymbol{\epsilon}_n^{e,\,tr}$, the trial stress $\boldsymbol{\sigma}_n^{\mathrm{tr}}$, the plastic strain tensor $\boldsymbol{\epsilon}_n^p$, and the total stress $\boldsymbol{\sigma}_n$ from the time step t_n. The stress at next time step $\boldsymbol{\sigma}_{n+1}$ is considered as the target node. The intermediate edges $\boldsymbol{\epsilon}_{n+1}^{e,\,tr}$ and $\boldsymbol{\epsilon}_{n+1}^{e}$ are considered as definitions and replaced by mathematical expressions in the ANN model. In particular, $DG2$ in Fig. 1, bottom, involves the following steps:

1. elastic trial strain $\boldsymbol{\epsilon}_{n+1}^{e,\,tr} = \boldsymbol{\epsilon}_{n+1} - \boldsymbol{\epsilon}_n^p$ (Computed in $DG2$ as a definition)
2. trial elastic stress $\boldsymbol{\sigma}_{n+1}^{\mathrm{tr}} = \boldsymbol{C}^e : \boldsymbol{\epsilon}_{n+1}^{e,\,tr}$ (history-independent **elastic step**, replaced by **FFNN**)
3. plastic strain $\boldsymbol{\epsilon}_{n+1}^p = \boldsymbol{\epsilon}_{n+1}^p(\boldsymbol{\epsilon}_{n+1}^p,\, \boldsymbol{\sigma}_{n+1}^{\mathrm{tr}},\, \lambda_{n+1}^p)$ (history-dependent **plastic step**, replaced by **RNN**)
4. elastic strain $\boldsymbol{\epsilon}_{n+1}^e = \boldsymbol{\epsilon}_{n+1} - \boldsymbol{\epsilon}_{n+1}^p$ (Computed in $DG2$ as a definition)
5. total stress $\boldsymbol{\sigma}_{n+1} = \boldsymbol{C}^e : \boldsymbol{\epsilon}_{n+1}^e$ (history-independent **elastic step**, replaced by **FFNN** from step 2 without a new training)

2.2. Data acquisition within isotropic/transversely-isotropic-elastic crystal plasticity

The focus in the following is on the solution of a f.c.c. single crystal under plastic deformations within small-strain framework. In particular, we follow the "ultimate algorithm" procedure, introduced in [16,40], to identify the set of active slip systems, where this algorithm delivers an exact solution for crystal plasticity in which the crystals are subjected to tension, shear or mixed tension–shear deformations varying as a ramp function. An abstract summary of the procedure is introduced in Appendix. To promote the effect of rotations on the mechanical response and, thus, the rotation role in the accuracy of the machine learning model, we consider in the crystal plasticity two cases of elastic response; one with isotropic linear elastic and one with transversely-isotropic elastic behavior within small strain framework. In this, we adopt the transversely-isotropic model presented in [41] and later extended within invariant framework in [42]. Therefore, having $\boldsymbol{C}^e$ as the elasticity tensor of the f.c.c. lattice and without the need to change the general formulation between the stress tensor $\boldsymbol{\sigma}$ and the elastic strain $\boldsymbol{\epsilon}^e$ as $\boldsymbol{\sigma} = \boldsymbol{C}^e : \boldsymbol{\epsilon}^e$, the elasticity tensor can be expressed for the isotropic case as

$$\boldsymbol{C}^e = \lambda\, \boldsymbol{I} \otimes \boldsymbol{I} + 2\,\mu\, \boldsymbol{I}^4_{\mathrm{sym}}\,, \tag{1}$$

with λ and μ being the microscopic Lamé constants. For the transverse isotropic case, the material is assumed to have a single preferred direction, denoted as $\boldsymbol{a}$ with $\|\boldsymbol{a}\| = 1$. The material symmetry group $\mathcal{MG}_T$ is defined by

$$\mathcal{MG}_T := \left\{ \boldsymbol{I};\ \boldsymbol{Q}(\theta,\, \boldsymbol{a}) \mid 0 \le \theta \le 2\pi \right\}, \tag{2}$$

where $\boldsymbol{Q}(\theta,\, \boldsymbol{a})$ represent all the rotations about the $\boldsymbol{a}$-axis. Moreover, an anisotropy structural tensor $\boldsymbol{M}$, which characterizes the inherent anisotropic of the material as introduced in [43], can then be defined as $\boldsymbol{M} = \boldsymbol{a} \otimes \boldsymbol{a}$. Following the derivations given in [41], the elasticity tensor for transversely-isotropic material applied in the manuscript can be expressed as

$$\boldsymbol{C}^e = \lambda_T\, \boldsymbol{I} \otimes \boldsymbol{I} + 2\,\mu_T\, \boldsymbol{I}^4_{\mathrm{sym}} + \alpha\big[\boldsymbol{M} \otimes \boldsymbol{I} + \boldsymbol{I} \otimes \boldsymbol{M}\big] + 2\,(\mu_L - \mu_T)\, \boldsymbol{I}^4_a + \beta\, \boldsymbol{M} \otimes \boldsymbol{M}\,, \tag{3}$$

with λ_T, μ_T, μ_L, α, β being material parameters and the 4th-order tensor $(\boldsymbol{I}^4_a)_{ijkl}$ is expressed in index notation in terms of the direction $\boldsymbol{a}$ and the Kronecker delta δ as $(\boldsymbol{I}^4_a)_{ijkl} := a_i(\delta_{jk}a_l + \delta_{jl}a_k) + a_j(\delta_{il}a_k + \delta_{ik}a_l)$.

3. Objective function in spectral form

3.1. Metrics for 3D rotations

We briefly review three distance functions or metrics reported in the literature for measuring the distance between two 3D rotations, see, e.g., [44–46] for more details and references. The aim of this presentation is to use these metrics, which are widely applied in, e.g., Robotics, computer vision or graphics, in the design of new machine learning objective functions that minimize the discrepancy in the training and, thus, in the calibration, between the reference or true rotation and the predicted rotation. Thus, having $\boldsymbol{R}_1, \boldsymbol{R}_2$ as two rotation tensors belonging to the Special Orthogonal Group, $SO(3)$, finding the distance function ϕ_i, with i referring to the type of metric, can be formulated as follows:

$$\text{For} \quad \boldsymbol{R}_1, \boldsymbol{R}_2 \in SO(3) \quad \text{find} \quad \phi_i(\boldsymbol{R}_1, \boldsymbol{R}_2) \,:\, SO(3) \times SO(3) \rightarrow \mathbb{R}^+ . \tag{4}$$

In the training of the machine learning model, the aim is to minimize ϕ_i , where the rotation tensors are formed by concatenating the orthogonal, normalized eigenvectors of, e.g., the stress or the strain tensors as will be discussed in later sections. In general, the distance functions must possess the following properties [20,45,47],

1. The statement $\phi_i(\boldsymbol{R}_1, \boldsymbol{R}_2) = 0$ is equivalent to $\boldsymbol{R}_1 = \boldsymbol{R}_2$
2. $\phi_i(\boldsymbol{R}_1, \boldsymbol{R}_2) = \phi_i(\boldsymbol{R}_2, \boldsymbol{R}_1)$
3. $\phi_i(\boldsymbol{R}_1, \boldsymbol{R}_3) \leq \phi_i(\boldsymbol{R}_2, \boldsymbol{R}_1) + \phi_i(\boldsymbol{R}_3, \boldsymbol{R}_1) \quad$ with $\quad \boldsymbol{R}_1, \boldsymbol{R}_2, \boldsymbol{R}_3 \in SO(3)$
4. $\phi_i(\boldsymbol{R}_1, \boldsymbol{R}_2)$ is *bi-invariant*, i.e., $\phi_i(\boldsymbol{R}_1, \boldsymbol{R}_2) = \phi_i(\boldsymbol{R}_3 \cdot \boldsymbol{R}_1, \boldsymbol{R}_3 \cdot \boldsymbol{R}_2) = \phi_i(\boldsymbol{R}_1 \cdot \boldsymbol{R}_3, \boldsymbol{R}_2 \cdot \boldsymbol{R}_3)$, $\boldsymbol{R}_1, \boldsymbol{R}_2, \boldsymbol{R}_3 \in SO(3)$
5. *Boundedly-equivalent* distance functions ϕ_A and ϕ_B fulfill the restriction $(a_r\, \phi_A \leq \phi_B \leq b_r\, \phi_A)$, with a_r and b_r being positive real numbers.
6. *Functional equivalent* ϕ_A and ϕ_B require the existence of a strictly-increasing positive continuous function f_r so that $f_r\,(\phi_A) = \phi_B$
7. Two metrics are *topologically equivalent* if they are either *boundedly equivalent* or *functional equivalent*

The discussed metrics in the following differ in the way that rotations are represented, the invariance and the accuracy in measuring the rotation distances.

- *Distance between the Euler Angles:* Having $\{\varphi_i,\, \theta_i,\, \psi_i\} \in \mathbb{E} \subset \mathbb{R}^+$ as the set of Euler angles of $\boldsymbol{R}_i$, the metric for $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$ can be expressed as the Euclidean distance between the Euler Angles as

$$\phi_{Eu} \;=\; \sqrt{d(\varphi_1, \varphi_2)^2 + d(\theta_1, \theta_2)^2 + d(\psi_1, \psi_2)^2}\,. \tag{5}$$

  In this, the Euclidean distance for two scalar-valued quantities a_1, a_2 is expressed as $d(a_1, a_2) = \min\{|a_1 - a_2|,\, 2\pi - |a_1 - a_2|\} \in [0, \pi]$. Thus, ϕ_{Eu} belongs to the range $[0, \pi\sqrt{3}]$. A drawback of using Euler Angles in calculating the metric is the non-uniqueness under certain extreme cases, such as $(\pi,\, \pi,\, 0)$ and $(0,\, 0,\, \pi)$ can represent the same rotation, however, their metric is not zero, [45]. As a solution to overcome this deficiency, the Euler angles can be imposed to be in half-open ranges as $\varphi_i,\, \theta_i \in [-\pi,\, \pi)$ and $\psi_i \in [-\pi/2,\, \pi/2)$, which allows ϕ_{Eu} to be valid as a metric on $SO(3)$.
- *Distance from the identity matrix:* The distance between $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$ can be expressed by means of the Frobenius norm $\|\cdot\|_F$ and after reformulation as

$$\phi_{DI} \;=\; \|\boldsymbol{I} - \boldsymbol{R}_1 \boldsymbol{R}_2^T\|_F = \sqrt{2\left[3 - \operatorname{tr}(\boldsymbol{R}_1 \boldsymbol{R}_2^T)\right]}\,. \tag{6}$$

  In Eq. $(6)_2$, the Frobenius norm was replaced by the $2nd$-norm, thus, the range of ϕ_{DI} values becomes $[0,\, 2]$. A mathematical proof of the validity of ϕ_{DI} as metric on $SO(3)$ was discussed by Huynh [45].
- *Geodesic on the unit sphere:* Having $SO(3)$ as Lie group of the 3D rotations, the skew-symmetric counterpart matrices represent Lie Algebra $so(3)$, which is a convenient way to represent the rotation. For a rotation matrix $\boldsymbol{R} \in SO(3)$, the skew-symmetric counterpart in Lie Algebra $\boldsymbol{W} \in so(3)$ is computed explicitly via logarithmic mapping. Thus, having Θ as the angle of rotation defined as

$$\Theta := \arccos\left[\tfrac{1}{2}(\operatorname{tr}\boldsymbol{R} - 1)\right] \quad \text{with} \quad \Theta \in [0, \pi]\,, \tag{7}$$

the logarithmic mapping that yields the skew-symmetric rotation matrix is computed by

$$\boldsymbol{W} = \log \boldsymbol{R} = \begin{cases} \boldsymbol{0} & \text{if } \Theta = 0 \\ \dfrac{\Theta}{2\sin\Theta}(\boldsymbol{R} - \boldsymbol{R}^T) & \text{if } \Theta \in (0, \pi) \\ \pm\pi\,[\tilde{\boldsymbol{v}}] & \text{if } \Theta = \pi\,, \end{cases} \tag{8}$$

where $[\tilde{\boldsymbol{v}}] \in so(3)$ is a skew-symmetric tensor. In particular, following the discussion in [44], $\operatorname{tr}\boldsymbol{R} = -1$ is associated with two solutions of $\log \boldsymbol{R}$, i.e. $\pm\pi\,[\tilde{\boldsymbol{v}}]$, where $\tilde{\boldsymbol{v}}$ is a unit length eigenvector of $\boldsymbol{R}$ corresponding to an eigenvalue equals to 1. In analogy, the mapping from Lie Algebra to Lie Group succeeds via exponential mapping as explained in, e.g., [47,48]. In particular, $\boldsymbol{R} \in SO(3)$ can be computed as follows:

$$\boldsymbol{R} = \exp \boldsymbol{W} = \begin{cases} \boldsymbol{I} & \text{if } \Theta = 0 \\ \boldsymbol{I} + \dfrac{\sin\Theta}{\Theta}\boldsymbol{W} + \dfrac{(1-\cos\Theta)}{\Theta^2}\boldsymbol{W}^2 & \text{if } \Theta > 0 \end{cases} \quad \text{with} \quad \Theta := \sqrt{\frac{1}{2}(\boldsymbol{W} : \boldsymbol{W})}\,. \tag{9}$$

Having $\boldsymbol{W}_1, \boldsymbol{W}_2 \in so(3)$ as the skew-symmetric rotation tensors corresponding to $\boldsymbol{R}_1, \boldsymbol{R}_2 \in SO(3)$, the distance function between the two rotations can be expressed as

$$\phi_{Lie} := \|\log(\boldsymbol{R}_1\boldsymbol{R}_2^T)\| = \|\log(\boldsymbol{R}_1) - \log(\boldsymbol{R}_2)\| = \|\boldsymbol{W}_1 - \boldsymbol{W}_2\|\,, \tag{10}$$

As a consequence of Eqs. (8) and (10), the range of ϕ_{Lie} values is $[0,\ \pi)$.

In the aforementioned metrics on $SO(3)$, ϕ_{DI} is dimensionless, whereas ϕ_{Eu} and ϕ_{Lie} have units in radian, which is not considered as a problem, but merely a scaled variation of ϕ_i values. Moreover, the three metrics are bi-invariant w.r.t. to the topology on $SO(3)$, and ϕ_{Lie} and ϕ_{DI} are boundedly equivalent. A drawback related to ϕ_{Eu} is that it sometimes results in large values although the rotations are too close, i.e. in connection with the extreme limits of the angles as π and $-\pi$, see, [45].

3.2. $SO(3)$-Invariant objective function

In the context of constitutive modeling, a material model should obey the principle of material frame indifference, also known as material objectivity, which states that constitutive relations have to be invariant under rigid body rotations of the actual configuration, see, [49]. In similar fashion, machine learning material models should also preserve the principle of material objectivity or material frame indifference, in which the predicted quantities (e.g. eigenvalues and directions) should be independent of the current frame of reference, i.e. observer-invariance. The importance of objectivity becomes more significant when, e.g. studying deformation of anisotropic materials, associated with large rotations of the eigenvectors. In this, the objectivity fulfillment in machine learning is associated with good accuracy in predicting both the eigenvalues and the rotation-related quantities like the eigenvectors, the Euler angles or the components of the skew-symmetric rotations in Lie Algebra.

In the direct relation (BB) approach and using the components of the strain tensor $\boldsymbol{\epsilon}$ to predict the components of the stress tensor $\boldsymbol{\sigma}$ expressed in a certain coordinate system via Voigt notation as illustrated in Fig. 2, left, the applied objective "loss" function in the training is based on the mean squared error (MSE) between the true output data $\overline{s}_i$ and the corresponding machine-predicted data $\overline{s}_i^M$, related to the stress measures. In particular, the scalar-valued loss function can be expressed by

$$\mathcal{D}_0^{ML} = \frac{1}{N}\sum_{i=1}^{N}\left[\overline{s}_i - \overline{s}_i^M\right]^2, \tag{11}$$

where N is the number of output data points. It is worth mentioning that $\overline{s}_i$ are scaled to be in the range of [0, 1], whereas $\overline{s}_i^M$, due to use of sigmoid activation function, have real-values in the range [0, 1].

The aforementioned loss function with tensors in component form does not necessarily satisfy the objectivity constraint of the material model as was discussed in [14]. Therefore, while maintaining the ANN architecture, we intend in the current work to test and compare different objective "loss" functions, such that based on rotation-related quantities, and evaluate their influence on the training performance. The proposed approaches aim to enforce the objectivity in a strict sense without the need to extend the training sets through providing database that include rigid-body rotations, but merely through changing the way that data are presented in the input and the output layers.

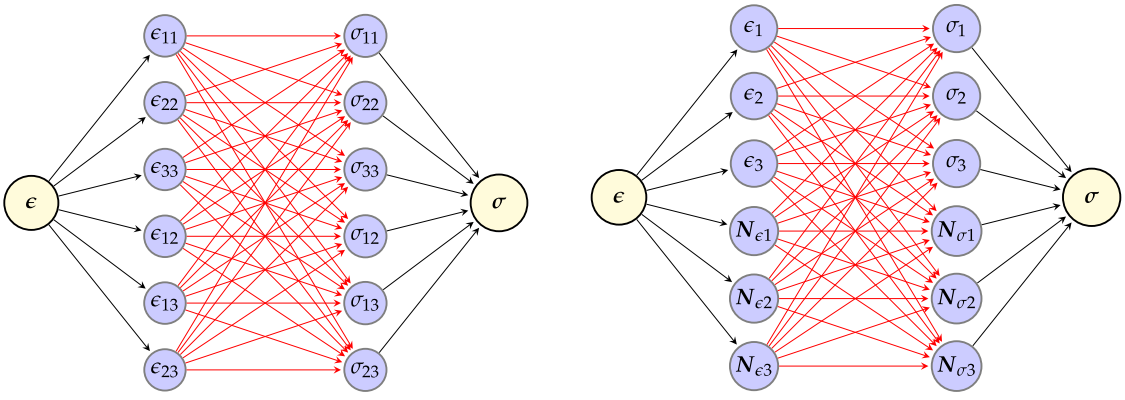

Fig. 2. Graph illustrating the strain and stress components (ϵ_{ij} and σ_{ij}) as input and output layers in the machine learning model (left). Illustration of the strain and stress principal components and the corresponding eigenvectors (right), where the red arrows are replaced by RNN model. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

The natural datasets resulting from the computational method in crystal plasticity are the strain tensor components and the corresponding components of the stress tensor. Thus, the first step in the underlying approach is the application of spectral decomposition to compute the eigenvalues and eigenvectors of the strain and stress tensors, which represent the input and the output of the machine learning model, i.e.,

$$\boldsymbol{\epsilon} = \sum_{i=1}^{3} \epsilon_i \, \boldsymbol{N}_{\epsilon i} \otimes \boldsymbol{N}_{\epsilon i} \qquad \text{and} \qquad \boldsymbol{\sigma} = \sum_{i=1}^{3} \sigma_i \, \boldsymbol{N}_{\sigma i} \otimes \boldsymbol{N}_{\sigma i} \, , \tag{12}$$

where ϵ_i and σ_i are the eigenvalues of $\boldsymbol{\epsilon}$ and $\boldsymbol{\sigma}$, respectively. Additionally, $\boldsymbol{N}_{\epsilon i} = (n_{\epsilon i})_j \, \boldsymbol{e}_j$ and $\boldsymbol{N}_{\sigma i} = (n_{\sigma i})_j \, \boldsymbol{e}_j$ are the corresponding normalized eigenvectors with the components $(n_{\epsilon i})_j$ and $(n_{\sigma i})_j$ and the index $j \in \{1, 2, 3\}$. The spectral decomposition allows to introduce alternative structure of the ML model by using the eigenvalues and eigenvectors of the strain to predict the eigenvalues and eigenvectors of the stress as illustrated in Fig. 2, right. Moreover, an alternative loss function based on the mean squared error (MSE) between the true output data $\overline{SN}_i$ of the eigenvalues and eigenvectors of the stress and the corresponding machine-predicted data $\overline{SN}_i^M$ can be introduced as

$$\mathcal{D}_{SN}^{ML} = \frac{1}{N} \sum_{i=1}^{N} \left[\overline{SN}_i - \overline{SN}_i^M \right]^2 . \tag{13}$$

where N is the number of output data points. From comparison with $\mathcal{D}_0^{ML}$, the computation of $\mathcal{D}_{SN}^{ML}$ at each iteration (ML-weights update) is more demanding as 12 components (3 for the eigenvalues and 9 for the eigenvectors) should be considered. A comparison with respect to the efficiency will be introduced in Section 5.

Inspired from the scalar-valued metrics that represent distances between two 3D rotations, discussed in the literature as in [44–46] and summarized briefly in Section 3.1, we introduce in the following three alternative loss functions that can be used in the training of the ML model in an attempt to improve the fulfillment of the material objectivity constraint. First of all, we consider the rotation tensors, formed by concatenating the orthogonal, normalized eigenvectors of the measured strain and stress tensors and expressed, respectively, as

$$\boldsymbol{R}_\epsilon = \begin{bmatrix} \boldsymbol{N}_{\epsilon 1} & \boldsymbol{N}_{\epsilon 2} & \boldsymbol{N}_{\epsilon 3} \end{bmatrix} \quad , \quad \boldsymbol{R}_\sigma = \begin{bmatrix} \boldsymbol{N}_{\sigma 1} & \boldsymbol{N}_{\sigma 2} & \boldsymbol{N}_{\sigma 3} \end{bmatrix} , \tag{14}$$

where $\boldsymbol{R}_\epsilon, \boldsymbol{R}_\sigma \in SO(3)$. For instance, in the BB-approach ML model, $\boldsymbol{R}_\epsilon$ beside the strain eigenvalues ϵ_i belong to the input layer, whereas $\boldsymbol{R}_\sigma$ beside the stress eigenvalues σ_i belong to the output layer. The rotation tensor can, however, be represented by equivalent quantities such as the Euler Angles or the entities of the skew-symmetric counterpart in Lie Algebra. In the training of the ANNs, we aim to minimize the metric $\Phi(\boldsymbol{R}_\sigma, \boldsymbol{R}_\sigma^M) : SO(3) \times SO(3) \rightarrow \mathbb{R}^+$ that measures the difference between the true rotation tensor $\boldsymbol{R}_\sigma$ (represents the measured orientation of the principal directions of the stress) and the predicted rotation tensor $\boldsymbol{R}_\sigma^M$. Thus, in ideal case,

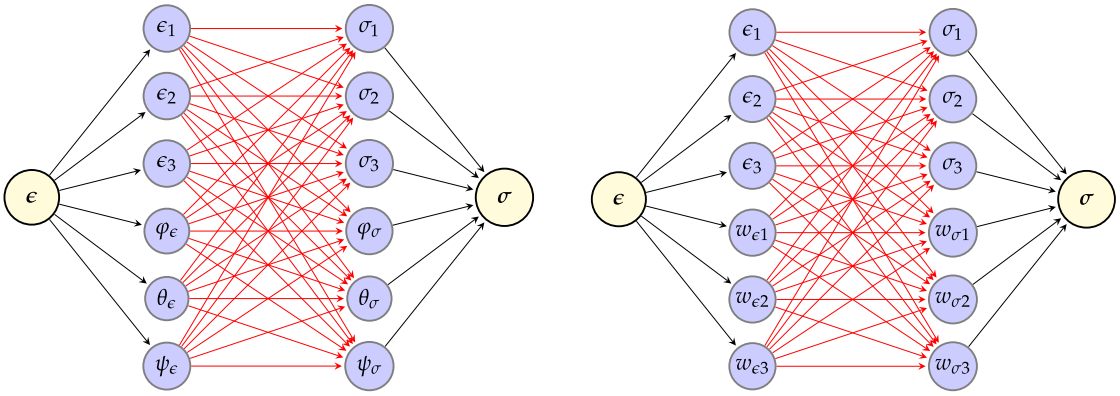

Fig. 3. Graph illustrating the strain and stress principal components and the corresponding Euler angles (left) and the strain and stress principal components and the corresponding components of the skew-symmetric rotation entities in Lie Algebra (right). The red arrows are replaced during the training by the RNN model.

$\Phi(\boldsymbol{R}_\sigma,\ \boldsymbol{R}_\sigma^M) = 0$ is equivalent to $\boldsymbol{R}_\sigma = \boldsymbol{R}_\sigma^M$. The proposed loss functions ($\mathcal{D}_i^{ML}$) can be seen as an average of two terms; one is related to the eigenvalues of the stress $\mathcal{D}_\sigma^{ML}$ and another term related to the rotation $\mathcal{D}_R^{ML}$, i.e.,

$$\mathcal{D}_{\sigma R}^{ML} = \tfrac{1}{2}(\mathcal{D}_\sigma^{ML} + \mathcal{D}_R^{ML}) \quad \text{with} \quad \mathcal{D}_\sigma^{ML} = \frac{1}{N}\sum_{i=1}^{N}\Big[\overline{\sigma}_i - \overline{\sigma}_i^M\Big]^2, \quad \text{and} \quad \mathcal{D}_R^{ML} = \frac{1}{N}\sum_{i=1}^{N}\overline{\Phi}_i\,, \tag{15}$$

where N is the number of output data points. $\overline{\sigma}_i$, $\overline{\sigma}_i^M$ are, respectively, the true and the predicted output data related to the eigenvalues of stress measure, whereas $\overline{\Phi}_i$ include the distance measures between the true and the predicted rotations. Fig. 3, left, illustrates the structure of the machine learning (BB-graph as example) for the case rotations represented via the Euler angles. In this, the input layer includes the principal strains and the Euler angles related to $\boldsymbol{R}_\epsilon$, which are used to predict the principal stresses and the Euler angles of the stress. Thus, having $\{\overline{\varphi}_\sigma, \overline{\theta}_\sigma, \overline{\psi}_\sigma\} \in \mathbb{E} \subset \mathbb{R}^+$ as the set of true Euler angles and $\{\overline{\varphi}_\sigma^M, \overline{\theta}_\sigma^M, \overline{\psi}_\sigma^M\} \in \mathbb{E} \subset \mathbb{R}^+$ are the predicted ones, adopting the *Euclidean distance between the Euler angles* from Section 3.1 yields for a certain test k

$$\overline{\Phi}_k = \sqrt{d(\overline{\varphi}_\sigma, \overline{\varphi}_\sigma^M)^2 + d(\overline{\theta}_\sigma, \overline{\theta}_\sigma^M)^2 + d(\overline{\psi}_\sigma, \overline{\psi}_\sigma^M)^2}\,. \tag{16}$$

Applying logarithmic mappings to the 3D, $SO(3)$ rotations (Lie group) yields an alternative representation of the rotation via the skew-symmetric counterpart matrices, i.e. Lie Algebra $so(3)$, as discussed in Section 3.1. Thus, instead of the Euler angles, the training of the RNNs can be applied to predict the three entities of the skew-symmetric rotation matrix $\boldsymbol{W}$ as illustrated in Fig. 3, right. With $\overline{w}_{\sigma i}$ as the true components of the skew-symmetric rotation tensor and $\overline{w}_{\sigma i}^M$ as the predicted ones, the distance function between the true and the predicted rotations, also called the *geodesic on the unit sphere*, can be expressed via the mean squared error as

$$\mathcal{D}_R^{ML} = \frac{1}{N}\sum_{i=1}^{N}\overline{\Phi}_i = \frac{1}{N}\sum_{i=1}^{N}\Big[\overline{w}_{\sigma i} - \overline{w}_{\sigma i}^M\Big]^2. \tag{17}$$

The last loss function is based on the *distance from the identity matrix*. The training in this case aims to predict the components of the rotation tensor, i.e. components of the stress eigenvectors. Unlike the loss function given in Eq. (13), which is based on the mean squared error, the aforementioned distance function can be calculated based on the components of the true rotation $\overline{\boldsymbol{R}}_\sigma$ and the predicted components of $\overline{\boldsymbol{R}}_\sigma^M$. In particular, for a certain test k, the rotation-related term in Eq. (15) can be expressed as

$$\overline{\Phi}_k = \|\boldsymbol{I} - \overline{\boldsymbol{R}}_\sigma\,(\overline{\boldsymbol{R}}_\sigma^M)^T\|_F = \sqrt{2\Big[3 - \mathrm{tr}\big[\overline{\boldsymbol{R}}_\sigma(\overline{\boldsymbol{R}}_\sigma^M)^T\big]\Big]} \tag{18}$$

with $\|\cdot\|_F$ being the Frobenius norm. In the following sections and applications, we use abbreviations related to each of the objective functions and summarized in Table 1.

Table 1
Summary of the considered objective "loss" functions.

Obj.	Description	Eq.
$\mathcal{D}_0^{ML}$	Based on the components of the true and predicted stress tensor	(11)
$\mathcal{D}_{SN}^{ML}$	Based on the eigenvalues and eigenvector entities of the true and predicted stress tensor	(13)
$\mathcal{D}_{\sigma Eu}^{ML}$	Based on the eigenvalues and Euler angles of the true and predicted stress tensor	(15), (16)
$\mathcal{D}_{\sigma Lie}^{ML}$	Based on the eigenvalues and the components of the skew-symmetric rotation tensor (Lie Algebra) of the true and predicted stress tensor	(15), (17)
$\mathcal{D}_{\sigma DI}^{ML}$	Based on the eigenvalues and the distance from the identity matrix of the predicted and true rotations	(15), (18)

3.3. ML implementation highlights

The machine learning model in this work is implemented in the high-level Python deep learning open-source code Keras (cf. [50]) with built-in LSTM neural networks together with several settings and options related, such that for layers, optimizers, objective functions and error measures. The tensorial operations in Keras are handled by the open-source symbolic tensor manipulation library Tensorflow, see [51], serving as the "backend engine" of Keras. After acquiring the database from the lower-scale crystal plasticity simulations in comma separated values (csv) format, the data are preprocessed in an open-source Python data analysis library Pandas [52]. In particular, we distinguish in the data preparation between training, validation and testing sets. Moreover, each set is further split into input and output subsets, where, for instance, the spectral decomposition is applied to obtain the eigenvalues and eigenvector and with that rotation-related quantities as the Euler angles or the skew-symmetric rotation matrix. This is followed by scaling (normalization) each sequence of input and output data from the original range to be within the range $[0, 1]$. This action is carried out using the MinMaxScaler class in sklearn.preprocessing toolkit [53]. After finishing the training, the predicted data can be rescaled to the original range using a similar scaling functionality. Following the preprocessing, the neural network is constructed, which consists of an input and output layer according to the description in Section 3. The number of hidden layers and the corresponding nodes have been varied to get the best possible fit for each case. In most of the cases, a choice of two hidden layers with ca. 100 nodes gave the best training.

4. Supervised machine learning with recurrent neural network

The focus in the underlying work is on history-dependent (path-dependent or inelastic) material mechanical behaviors, such as in crystal plasticity. This kind of behavior can be treated within the artificial neural networks (ANNs) though application of recurrent neural networks (RNN). In particular, the Long Short-Term Memory (LSTM) neural network as a part of the RNN approach, introduced by Hochreiter and Schmidhuber [54], is applied to consider the effect of history variables on the path-dependent behavior.

4.1. Overview and methods

The building of the machine learning models in this contribution involves the following steps:

1. Generation of synthetic datasets through mechanically-loaded (strain-controlled) f.c.c. single crystal to get the strain–stress components time history.
2. Applying of spectral decomposition to extract the principal components and the related eigenvectors (principal directions).
3. Formation of the rotation tensors through concatenating the eigenvectors, and then reformulate these to get simplified rotation-related quantities such as Euler Angles and the geodesic on the unit sphere (through logarithmic mapping within Lie Algebra).
4. Modify the loss function of the machine learning algorithm through application of metrics for 3D rotations as summarized in Table 1.
5. Comparison between the three approaches of BB, $DG1$ and $DG2$, which also consider a variety of loss functions.

Table 2
Definition of the considered datasets in Keras machine learning model.

Dataset	Description
Training	Represents the data that the ML model will be fit into by adjusting the weights of the neural networks.
Validation	Represents an independent dataset that uses the generated weights to measure the ML model performance during the training process. In this, it is evaluated at each epoch (weights update) during the training, i.e. its loss function and accuracy are calculated and compared to that related to the training dataset. Thus, it contributes to the tuning of the neural networks weights during the training.
Testing	Represents an unseen dataset, which is not a part of the training. It is used to calibrate the performance of a trained ML model, as testing the accuracy a given configuration of the neural networks and the applied deep learning models in Keras.

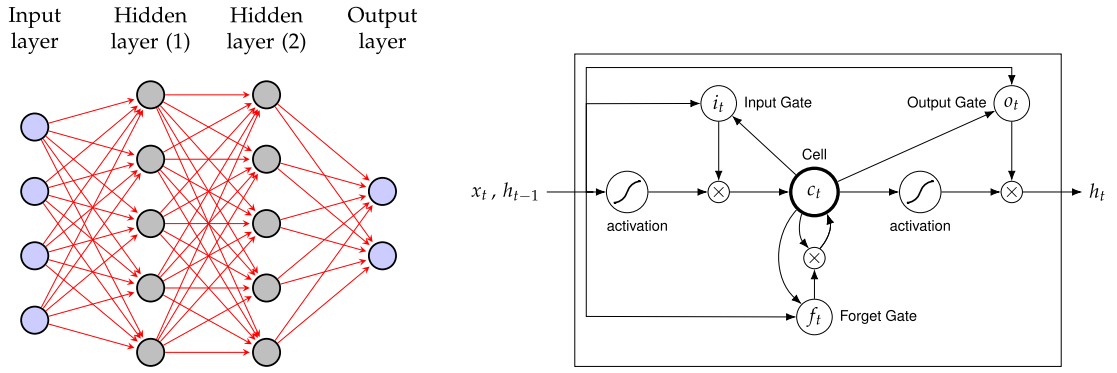

Fig. 4. Graph illustration of an ANN structure (left). A common architecture of a neuron of the LSTM variation with a cell (c_t) and input (i_t), output (o_t), and forget (f_t) gates beside the activation functions (right).

It is also worth mentioning that if experimental data are available, such that from laboratory experiments, there is no need to go through steps 1. and, instead, the modeler can directly start with step 2. In this work, a manual data splitting is carried out by specifying from the beginning three datasets, i.e. training, validation and testing, with brief definitions given in Table 2.[1]

Following the specification of the datasets, the artificial neural network (ANN) is constructed, which, in general, consists at least of three types of layers, the input, the output and the hidden layers, with equal or different number of neurons (vertices, nodes or units). As an example, Fig. 4, left, illustrates ANN with two hidden layers with 5 neurons for each, an input layer with 4 inputs and an output layer with 2 outputs.

In the training of the neural network, each neuron receives inputs (x_i) from other neurons or from the input layer and computes an output (y) via a non-linear activation function (f). In particular, we have

$$y = f(w_i\, x_i + b) \quad \text{with, e.g.,} \quad f(x) = \sigma(x) = 1/(1 + e^{-x})\,. \tag{19}$$

In this, $\sigma(x)$ is the sigmoid activation function that takes a real-valued input (x) and returns it to range between 0 and 1, whereas several other activation functions, such as $tanh$ and $ReLU$ (Rectified Linear Unit), can be found in Keras. In Eq. (19), w_i represent the weights and b is the bias, which are adjusted to minimize the discrepancy between the benchmark and prediction that are measured at the output layer via the loss function. Following this, the errors related to each vertex in the output layer will be back propagated to the neurons in the hidden layer and used to calculate the gradient of the loss function, which is also used to update the weights and, thus, minimized the loss function values.

4.2. Deep learning using the long short-term memory (LSTM)

Unlike the feed-forward neural networks (FFNN), which can be considered as mathematical functions, artificial recurrent neural networks (RNNs) can be seen as dynamical systems in the sense that they introduce loops in the

[1] https://towardsdatascience.com/train-validation-and-test-sets-72cb40cba9e7

network allowing for back propagation of past information into the network. Thus, they exhibit temporal dynamic behaviors that might continuously be changing during the training of the network, see, e.g., [55] for a thorough review and challenges. This allows them to be suitable for problems that involve time-dependent and sequence-dependent data, such as in time-dependent material behaviors, engineering by Zhu et al. [56] or Wang and Sun [14] to replace constitutive models by machine learning models.

In this regard, LSTM was introduced by Hochreiter and Schmidhuber [54] as a variation of the RNNs, which has the merit of preserving the error that can be backpropagated through time and layers and presents a solution to the problem of vanishing gradient. The LSTM is implemented in Keras and illustrated graphically in Fig. 4, right. This shows a cell (c_t), which represents a dynamic memory of LSTM that keep the dependencies between the elements in the input sequence. Beside this, the input (i_t) gate regulates which information influx into the cell. The forget (f_t) gate controls which values remain in the cell, whereas the output (o_t) gate controls to which extend that values in the cell are used to compute the output activation function, see, [57] for more details.

For illustration, assuming x_t as the input sequence at time t, which also include the previous outputs from the cell h_{t-1}, and h_t as the corresponding output sequence. In order to determine what information should be forgotten from the cell state, x_t and h_{t-1} are passed through the *forget gate*, where they are subjected to a sigmoid activation function σ and return values between 0 and 1, i.e.,

$$f_t = \sigma(W_f \cdot x_t + U_f \cdot h_{t-1} + b_f)\,, \tag{20}$$

where W_f and U_f are weight matrices and b_f is a bias vector related to the forget gate. Without the forget gate connected to the reset of the cell state, the internal state values may grow indefinitely and eventually cause the network to collapse as was discussed by Gers et al. [58]. Following this is the process of determining the information that could be *stored* in the cell state, i.e. $\tilde{C}_t$, which is a process that includes two activation functions; a sigmoid activation function σ related to the input gate i_t and a hyperbolic tangent function $tanh$ as

$$i_t = \sigma(W_i \cdot x_t + U_i \cdot h_{t-1} + b_i) \quad \text{and} \quad \tilde{C}_t = \tanh(W_C \cdot x_t + U_C \cdot h_{t-1} + b_C)\,. \tag{21}$$

In this, W_i and U_i are weight matrices and b_i is a bias vector related to the input gate, and W_C and U_C are weight matrices and b_C is a bias vector related to $\tilde{C}_t$. Thereafter, the updated 'current' cell state can be calculated based on the old cell state C_{t-1} as

$$C_t \;=\; f_t\, C_{t-1} \;+\; i_t\, \tilde{C}_t\,. \tag{22}$$

Finally, the output signal is calculated after inclusion of the current cell state C_t and the signal through the output gate o_t as

$$h_t \;=\; o_t\, \tanh(C_t) \quad \text{with} \quad o_t = \sigma(W_o \cdot x_t + U_o \cdot h_{t-1} + b_o)\,, \tag{23}$$

where W_o and U_o are related weight matrices and b_o is a bias vector.

5. Results and discussions of test cases via monotonic loading

The purpose of the following numerical experiments is to evaluate the accuracy and compare the different objective functions following the direct or "black box" (BB) graph and the indirect or "physically-informed" ($DG1$ and $DG2$) directed graphs, as was illustrated in Fig. 1. Beside the supervised machine learning model building steps explained in Section 4.1, we split the generated database into three sets, i.e. training, validation and testing, as was also explained in Table 2. In a nutshell, the **training subset** is used for adjusting the weights of the neural networks, the **validation subset** periodically evaluates the quality of training and, thus, contributes to tuning of the neural networks weights, and the **testing subset** represents the unseen data and used for the final evaluation of the trained ML model.

*5.1. Applying **BB**-graph to isotropically-elastic crystal plasticity (***Iso CP***)*

We test in the following the performance of several ML objective functions based on data generated from the crystal plasticity (elasto-plastic) material model. This includes a single f.c.c. crystal with an isotropic, linear elastic material behavior together with a maximum number of 12 activatable slip systems, see Appendix for details. The

Table 3
Parameters of the single f.c.c. crystal with isotropic elasticity.

Parameter	sym.	val.
Young's modulus	E	1500
Poisson's ratio	ν	0.33
Initial yield stress	τ_{Y0}	1.0
Taylor hardening	$\hat{h}$	5.0
1st Euler angle	θ	0°, 30°, 60°, 90°, 120°, 150°, 180°
2nd Euler angle	φ	0°, 15°, 30°, 45°, 60°, 75°, 90°

Table 4
24 loading cases (incremental plane strain) applied to each of the 49 crystal configuration.

	$\Delta\epsilon_{11}$	$\Delta\epsilon_{22}$	$\Delta\epsilon_{12}$		$\Delta\epsilon_{11}$	$\Delta\epsilon_{22}$	$\Delta\epsilon_{12}$
(1)	0.	0.	$4\cdot10^{-4}$	(13)	$2\cdot10^{-4}$	$2\cdot10^{-4}$	0.
(2)	0.	0.	$2\cdot10^{-4}$	(14)	$4\cdot10^{-4}$	$2\cdot10^{-4}$	0.
(3)	0.	0.	$1\cdot10^{-4}$	(15)	$1\cdot10^{-4}$	$2\cdot10^{-4}$	$4\cdot10^{-4}$
(4)	0.	0.	$-1\cdot10^{-4}$	(16)	$1\cdot10^{-4}$	$2\cdot10^{-4}$	$2\cdot10^{-4}$
(5)	0.	0.	$-2\cdot10^{-4}$	(17)	$1\cdot10^{-4}$	$-2\cdot10^{-4}$	$1\cdot10^{-4}$
(6)	0.	0.	$-4\cdot10^{-4}$	(18)	$-1\cdot10^{-4}$	$2\cdot10^{-4}$	$-1\cdot10^{-4}$
(7)	$-4\cdot10^{-4}$	$-2\cdot10^{-4}$	0.	(19)	$-1\cdot10^{-4}$	$-2\cdot10^{-4}$	$-2\cdot10^{-4}$
(8)	$-2\cdot10^{-4}$	$-2\cdot10^{-4}$	0.	(20)	$-1\cdot10^{-4}$	$-2\cdot10^{-4}$	$-4\cdot10^{-4}$
(9)	$2\cdot10^{-4}$	$-2\cdot10^{-4}$	0.	(21)	$2\cdot10^{-4}$	$1\cdot10^{-4}$	$2\cdot10^{-4}$
(10)	$4\cdot10^{-4}$	$-2\cdot10^{-4}$	0.	(22)	$2\cdot10^{-4}$	$-1\cdot10^{-4}$	$1\cdot10^{-4}$
(11)	$-4\cdot10^{-4}$	$2\cdot10^{-4}$	0.	(23)	$-2\cdot10^{-4}$	$1\cdot10^{-4}$	$-1\cdot10^{-4}$
(12)	$-2\cdot10^{-4}$	$2\cdot10^{-4}$	0.	(24)	$-2\cdot10^{-4}$	$-1\cdot10^{-4}$	$-2\cdot10^{-4}$

material properties with consistent units are taken from [40] and summarized in Table 3. The values of the related Euler angles show that the crystal can be found in 49 different configurations (i.e. 49 different rigid body rotations).

As a part of an initial-boundary-value problem that replicates real events, the crystal might be subjected to many different loading scenarios. To restrict the loading cases in our numerical tests, we assume that the crystal is under plane-strain states, i.e. $\overline{\epsilon}_{33} = \overline{\epsilon}_{13} = \overline{\epsilon}_{23} = 0$, which leads, however, to possible 3D rotations of the principal stress directions. In total, 1176 experiments as a result of 24 numerical experiments for each of the 49 crystal configurations have been carried out. These experiments have been then spit into training (792 experiments), validation (192 experiments) and testing (192 experiments) subsets. Furthermore, each of these subsets belongs to different frames of reference, i.e. for the validation subset we have $\theta = 120°$ with $\varphi = 0°$, 15°, 30°, 45°, 60°, 75° beside $\theta = 180°$ with $\varphi = 75°$, 90°. For the unseen testing subset, we consider $\theta = 30°$ with $\varphi = 0°$, 15°, 30°, 45°, 60°, 75°, 90° and $\theta = 120°$ with $\varphi = 90°$. In each numerical experiment, the crystal is subjected to monotonic incremental of one or more strain components, i.e. $\Delta\epsilon_{11}$, $\Delta\epsilon_{22}$ and $\Delta\epsilon_{12}$, where 100 increments have been applied. A summary of applied loading combinations for each crystal configuration is introduced in Table 4.

For the aim of the study, we start by testing the performance of all suggested objective functions considering crystal plasticity with isotropic elastic response and the BB graph. In this, the rotations in the principal stresses are associated merely with the plastic deformations, as isotropic elastic response does not lead to principal stress rotation. For the aim of abstract writing, Table 5 presents a summary of the ML methods applied within the direct "black box" (BB) graph.

Of particular importance in constitutive modeling is the material frame indifference (or material objectivity), which is quantified in the case of ML models by the accuracy of predicting the frame-independent eigenvalues and the related eigenvectors (or their rotation angles, as three eigenvectors form a rotation matrix). In the different approaches in this work, the accuracy is quantified by calculating the discrepancy between the true data (X_{true}) and the predicted data (X_{pred}) using the scaled mean squared error (scaled MSE), which can be expressed for each experiment (i) with number of steps (N = 100) as

$$\text{scaled MSE}_i \;=\; \frac{1}{N}\sum_{j=1}^{N}\Big[(\overline{X}_{\text{true}})_j \;-\; (\overline{X}^{M}_{\text{pred}})_j\Big]^2 \quad \text{where} \quad \overline{X}_{\text{true}} \coloneqq \text{MinMaxScaler}(X_{\text{true}})\,. \tag{24}$$

Table 5
Summary of the ML methods applied within the BB approach, see, Fig. 1, left. The objective function definitions are given in Table 1.

Method	Description	Obj.
BB_{comp}	Considers the components of the true and predicted stress tensor, i.e. Fig. 2, left	$\mathcal{D}_0^{ML}$
BB_{Lie}	Considers the eigenvalues and the components of the skew-symmetric rotation tensor (Lie Algebra) of the true and predicted stress tensor, i.e. Fig. 3, right	$\mathcal{D}_{\sigma Lie}^{ML}$
BB_{Eul}	Considers the eigenvalues and Euler angles of the true and predicted stress tensor, i.e. Fig. 3, left	$\mathcal{D}_{\sigma Eu}^{ML}$
BB_{SN}	Considers the eigenvalues and eigenvectors of the true and predicted stress tensor, i.e. Fig. 2, right	$\mathcal{D}_{SN}^{ML}$
BB_{DI}	Considers the eigenvalues and the distance from the identity matrix of the predicted and true rotations, i.e. Fig. 2, right	$\mathcal{D}_{\sigma DI}^{ML}$

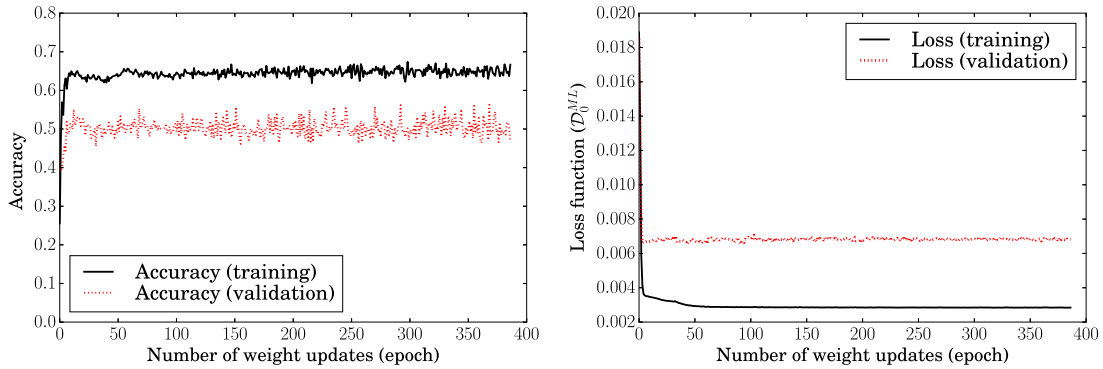

Fig. 5. ML training history related to the training and validation datasets within **Iso CP**; Accuracy over number of weight updates (left) and loss function value over number of weight updates (right) following BB_{comp} method.

In particular, the MinMaxScaler is constructed based only on X_{true} data, and then this scalar is used to scale the predicted values X_{pred} yielding $\overline{X}^M_{\text{pred}}$. Thus, $\overline{X}_{\text{true}}$ is found in the range [0, 1], whereas $\overline{X}^M_{\text{pred}}$ is not necessarily in the range [0, 1].

As a starting point, the BB_{comp} approach is applied for generation of ML material model related to isotropic elastic crystal-plasticity (Iso CP) model. Fig. 5 shows the training history of the training and validation datasets following the BB_{comp} method within direct graph approach, where increasing the number of weight updates further than ca. 380 does not lead to improvement of the accuracy or reduction of values of the loss function $\mathcal{D}_0^{ML}$.

For the BB_{Comp} method, the frame-independent quantities (the eigenvalues and eigenvectors) do not result directly from the ML model. Thus, as a part of the post-processing, the spectral decomposition is applied to extract the three eigenvalues and eigenvectors and, thus, the related three Euler angles of the true and predicted stresses, which allows to calculate the scaled MSE using Eq. (24). Fig. 6 illustrates the scaled MSE for each experiment of the training, validation and testing datasets considering the BB_{Comp} ML approach, where Fig. 6, left, is related to the overall MSE (the principal stresses and Euler angles together). The scaled MSE values in Fig. 6, middle, are related to the pure principal stresses, whereas Fig. 6, right, are connected the Euler angles. The three cases in Fig. 6 show that the training using the components of the stress and strain tensors with the loss function $\mathcal{D}_0^{ML}$ results in very accurate predictions with respect to the principal stress values (PV), whereas the main source of errors is the prediction of the rotation-related Euler angles (Rot).

For evaluating the performance of the ML model, we qualitatively compare between the scaled MSE of the testing and that of the training datasets. This is based on calculating the non-parametric, empirical cumulative distribution functions (eCDFs), see, [59,60]. In particular, eCDFs are calculated for the MSE_i of testing (N_{testdata}) and training ($N_{\text{traindata}}$) datasets considering all the components (all), i.e. $i \in [1, N^{\text{all}}_{\text{testdata}}]$, $i \in [1, N^{\text{all}}_{\text{traindata}}]$, only the principal stress values (PV), i.e. $i \in [1, N^{\text{PV}}_{\text{testdata}}]$, $i \in [1, N^{\text{PV}}_{\text{traindata}}]$, and only the rotation-related quantities (Rot), i.e. $i \in [1, N^{\text{Rot}}_{\text{testdata}}]$, $i \in [1, N^{\text{Rot}}_{\text{traindata}}]$. Thus, for a dataset N with MSE_i sorted in ascending order, the eCDF

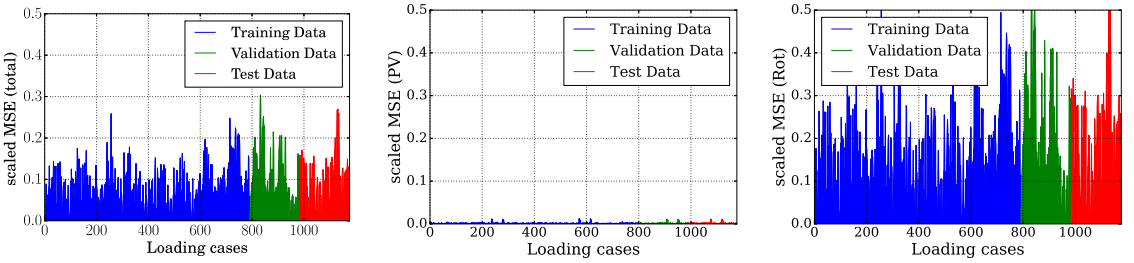

Fig. 6. Scaled MSE of the training, validation and testing datasets within **Iso CP** using BB_{Comp} method. Overall scaled MSE of the predicted principal stresses and the corresponding Euler angles (left), scaled MSE related only to the principal stresses (middle) and scaled MSE related only to the Euler angles (right).

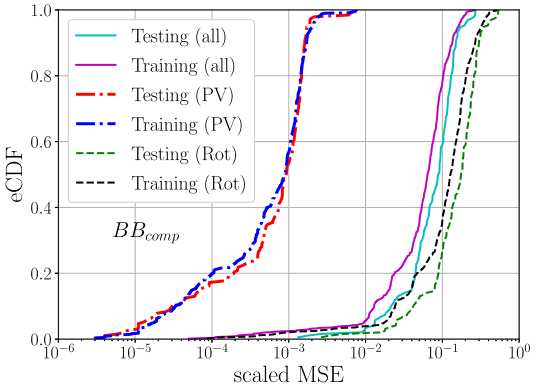

Fig. 7. eCDF vs. scaled MSE of the training and testing dataset of **Iso CP** using the BB_{Comp} method.

can be computed as follows:

$$F_N(\text{MSE}) = \begin{cases} 0, & \text{MSE} < \text{MSE}_1, \\ \dfrac{r}{N}, & \text{MSE}_r \le \text{MSE} < \text{MSE}_{r+1},\ r = 1, \ldots, N-1, \\ 1, & \text{MSE}_N \le \text{MSE}. \end{cases} \tag{25}$$

Following this, Fig. 7 shows that the ML model with BB_{Comp} does not suffer from a significant over-fitting in predicting the principal stresses (PV) as the curves of the training and testing are very close to each other. However, some over-fitting in the prediction of the rotation-related Euler angles (Rot) can be noticed. Moreover, it shows that most of the errors stem from the prediction of the rotations.

Alternatively, we apply and compare in the following approaches (BB_{Lie}, BB_{Eul}, BB_{SN}, and BB_{DI}), which were described in Table 5. Fig. 8 presents the eCDF vs. scales MSE curves related to the four methods. A common observation related to the four methods is that most of the errors stem from prediction of the rotations, whereas the principal stress values prediction is far more accurate. Moreover, unavoidable slight over-fitting can be observed in connection with the rotation prediction (Rot) in BB_{Eul} and BB_{Lie}. Besides, the errors related to the training and testing datasets are very close in BB_{SN} and BB_{DI}.

For a more comprehensive comparison, Fig. 9 presents the plots eCDF vs. scales MSE for the five methods in Table 5. In particular, Fig. 9, right, shows that BB_{Eul} yields the least errors in prediction of the rotations, which is followed by BB_{Lie}, whereas BB_{SN} and BB_{DI} yield the most errors in rotation prediction. On the other hand, Fig. 9, left, shows that BB_{Eul}, BB_{Lie}, and BB_{SN} yield similar accuracy in predicting the principal stress values, whereas BB_{DI} presents the least accurate scheme. BB_{Comp} shows inconsistency in PV prediction accuracy, as it results in high accuracy in almost 30% of experiments and low accuracy in the rest experiments. Moreover, both BB_{SN} and BB_{DI} approaches are more expensive in the training as the eigenvalues of the stresses include 3 entities and the three eigenvectors include 9 entities (in total 12 entities), where as the BB_{Comp}, BB_{Lie} and BB_{Eul} approaches

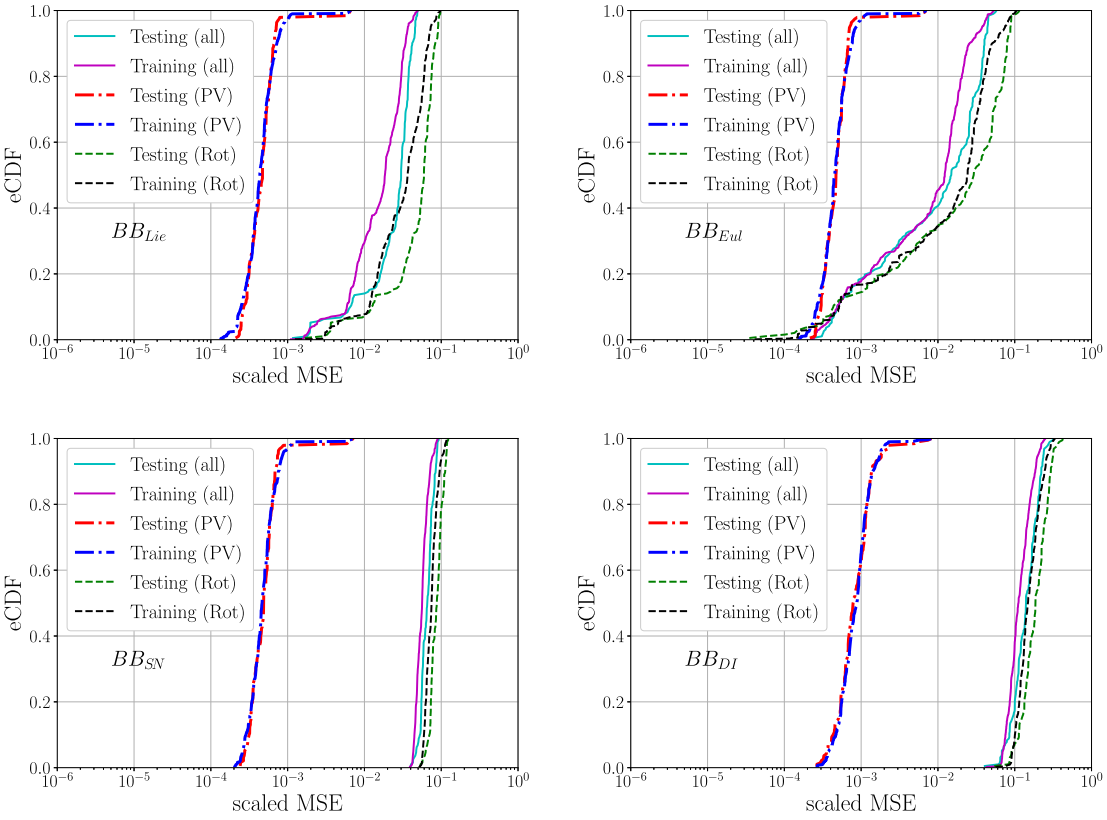

Fig. 8. eCDF vs. scales MSE of the training and testing datasets within **Iso CP** using BB_{Lie}, BB_{Eul}, BB_{SN}, and BB_{DI} methods.

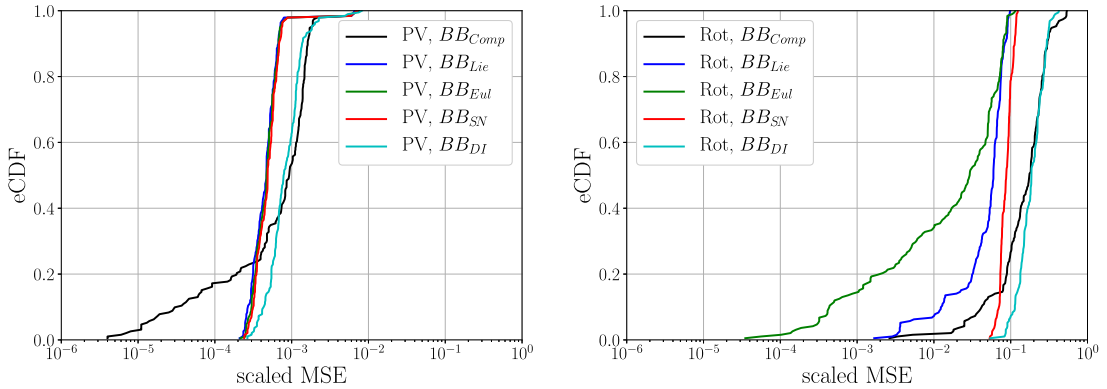

Fig. 9. eCDF vs. scaled MSE within **Iso CP** of the testing dataset related to the principal stress values (PV) and the rotation-related quantities (Rot) using BB_{Comp} BB_{Lie}, BB_{Eul}, BB_{SN}, and BB_{DI} methods.

include, in total, 6 entities, thus, much cheaper with regard of computational costs. Based on this, BB_{SN} and BB_{DI} approaches will be excluded from the further discussions.

*5.2. Applying **BB**-graph to transversely-isotropic-elastic crystal plasticity (***TranIso CP***)*

The following case studies aim to test the performance of the three ML objective functions applied in BB_{Comp}, BB_{Lie} and BB_{Eul} methods, and based on database generated from an anisotropic crystal plasticity material model

Table 6
Parameters of the single f.c.c. crystal with transversely-isotropic elasticity (**TranIso CP**).

Parameter	sym.	val.
Lamé's first parameter	λ_T	1095.0
1st shear modulus	μ_T	508.0
2nd shear modulus	μ_L	620.0
material parameter	α	10.0
material parameter	β	10.0
initial yield stress	τ_{Y0}	1.0
Taylor hardening	$\hat{h}$	5.0
1st Euler angle	θ	0°, 30°, 60°, 90°, 120°, 150°, 180°
2nd Euler angle	φ	0°, 15°, 30°, 45°, 60°, 75°, 90°

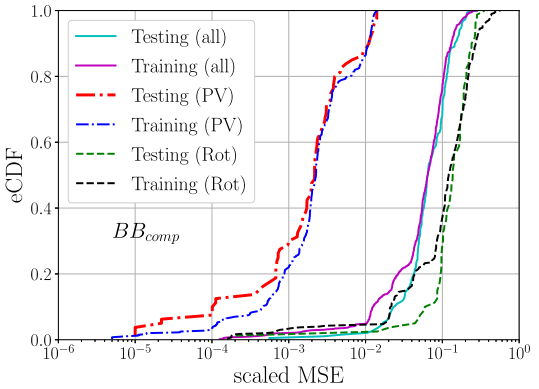

Fig. 10. eCDF vs. scaled MSE within **TranIso CP** of the training and testing datasets using the BB_{Comp} method.

(TranIso CP). This is related to a single f.c.c. crystal with a transversely-isotropic elastic behavior together with a maximum number of 12 activatable slip systems, see Appendix for details. Unlike the previous case of isotropic elasticity, having anisotropic elasticity leads to rotations in the principal directions in both elastic and plastic regimes, thus, making it even more challenging in the machine learning with respect to capturing the rotation patterns in the model training.

The material properties with consistent units are based on [40,41] and summarized in Table 6, where the related Euler angles show that the crystal can be found in 49 different configurations (different rigid body rotations). The applied loading combinations for each crystal configuration are given is Table 4, which result in a dataset of 1176 experiments. For the ML procedure, these experiments are spit into training (792 experiments), validation (192 experiments) and testing (192 experiments) subsets.

We start by applying the BB_{comp} approach within transversely-isotropic crystal plasticity (TranIso CP), which is followed by application of spectral decomposition to extract the eigenvalues and Euler angles of the predicted stresses. Fig. 10 shows again that most of the errors in connection with the testing and training datasets (all) are related to the prediction of the Euler angles (Rot), whereas good predictions of the principal stress values (PV) can be obtained. Moreover, a slight over-fitting in the training can be observed (scaled MSE of the testing dataset are slightly greater than that of the training dataset).

In an attempt to improve the prediction of the rotations in ML material model, we alternatively apply BB_{Lie} and BB_{Eul}, which consider in the training the three components of the skew-symmetric rotation matrix and the Euler angles, respectively. Fig. 11 shows the eCDF vs. scales MSE curves related to the BB_{Lie} approach (left) and that related to BB_{Eul} (right). A similar conclusion as before can be drawn here; In most of the experiments in training and testing datasets, the errors from the rotation prediction (Rot) dominate over that from prediction of the principal stress values (PV).

For a quantitative comparison between the aforementioned BB approaches, Fig. 12, left, shows that, except for almost 30% of the experiments, BB_{Lie} and BB_{Eul} are more accurate than BB_{Comp} in predicting the principal stress

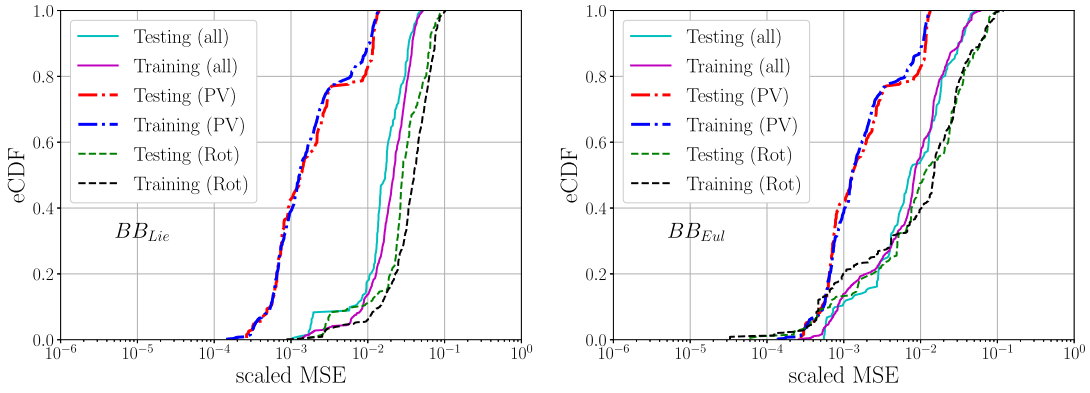

Fig. 11. eCDF vs. scales MSE of the training and testing datasets within **TranIso CP** using BB_{Lie} and BB_{Eul} approaches.

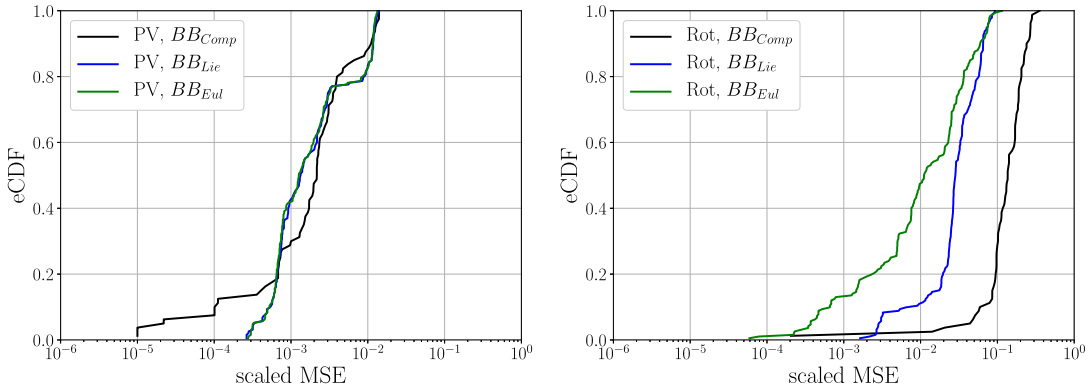

Fig. 12. eCDF vs. scaled MSE within **TranIso CP** of the training and testing datasets related to the principal values (PV) and the rotations (Rot). The applied approaches are BB_{Comp} BB_{Lie} and BB_{Eul}.

Table 7
Summary of the ML methods applied within the $DG1$ approach, see, Section 2.

Method	Description	Obj.
$DG1_{comp}$	Considers the components of tensors in the training together with the directed graph ($DG1$) in Fig. 1	$\mathcal{D}_0^{ML}$
$DG1_{Lie}$	Considers the eigenvalues and the components of the skew-symmetric rotation tensor (Lie Algebra) with the directed graph ($DG1$) in Fig. 1	$\mathcal{D}_{\sigma Lie}^{ML}$
$DG1_{Eul}$	Considers the eigenvalues and Euler angles of tensors with the directed graph ($DG1$) in Fig. 1	$\mathcal{D}_{\sigma Eu}^{ML}$

values. However, Fig. 12, right, shows that BB_{Eul} is more accurate than BB_{Lie} in predicting the rotation attributes, whereas BB_{Comp} has the least accuracy in rotation prediction.

*5.3. Applying **DG1**-graph to transversely-isotropic-elastic crystal plasticity (**TranIso CP**)*

With the aim to improving the prediction capacity of the BB approaches, especially that related to the rotation prediction, a simple modification can applied through inclusion of the history of the output stresses. This yields an incremental prediction scheme ($DG1$-approach), which makes uses of the predicted outputs at previous steps to improve the predictions at next steps as explained in Section 2 . For an abstract writing, Table 7 presents a summary of the ML methods applied within the $DG1$-approach, where the three objective functions $\mathcal{D}_0^{ML}$, $\mathcal{D}_{\sigma Eu}^{ML}$ and $\mathcal{D}_{\sigma Lie}^{ML}$ are implemented.

Starting with $DG1_{comp}$ method as explained in Table 7, Fig. 13, right, shows the ML accuracy and loss function values over number of weight updates (epoch). It is obvious that the training and validation datasets yield coincide

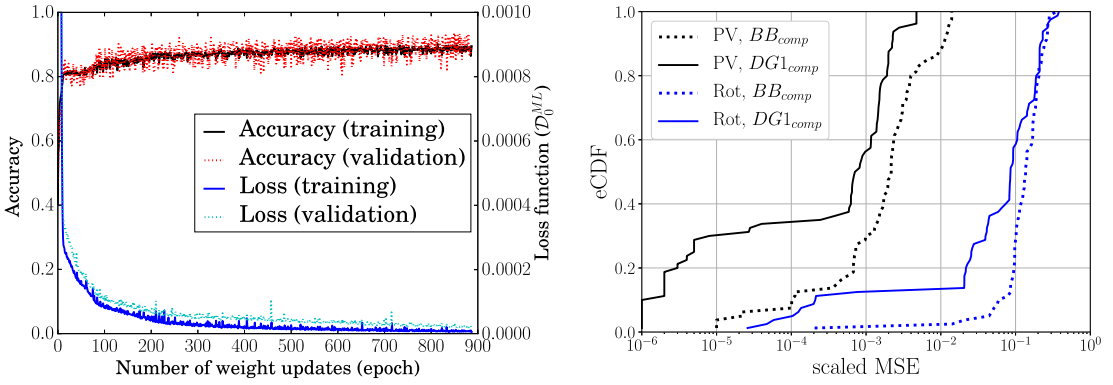

Fig. 13. ML accuracy and loss function values over number of weight updates for **TranIso CP** case using $DG1_{comp}$ settings (left), and a comparison between $DG1_{comp}$ and BB_{comp} with regard to the prediction accuracy of the eigenvalues and Euler angles of the testing dataset (right).

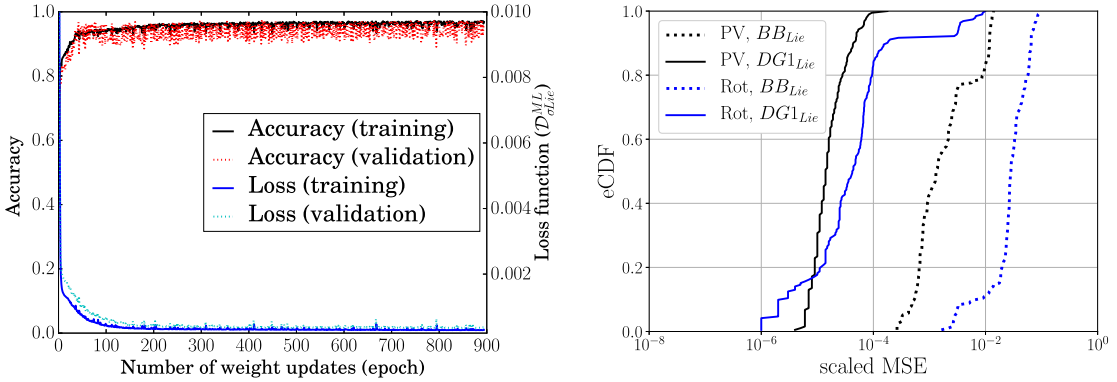

Fig. 14. ML accuracy and loss function values over number of weight updates for **TranIso CP** case using $DG1_{Lie}$ settings (left), and a comparison between $DG1_{Lie}$ and BB_{Lie} with regard to the prediction accuracy of the eigenvalues and the skew-symmetric rotation tensor components of the testing dataset (right).

curves, i.e. no over-fitting occurs. Applying the spectral decomposition to generate the frame-independent quantities of the predicted and true stresses, i.e. the eigenvalues and the related Euler angles, allows to compare between $DG1_{comp}$ and BB_{comp} with regard to the prediction accuracy. Fig. 13, left, shows that applying $DG1_{comp}$ leads to a significant improvement in the accuracy in comparison with BB_{comp}, especially in the errors related to the principal stress values (PV). However, in spite of the accuracy improvement, the prediction of Euler angles (Rot) still the main source of errors in the $DG1_{comp}$ approach, where only a portion of the testing experiments (eCDF ¡ 0.2) show good accuracy in rotation prediction.

In an analogous treatment using $DG1_{Lie}$ approach (see, Table 7), Fig. 14, left, shows an alignment between the training and the testing datasets with regards to the training accuracy and the loss function values. In comparison with BB_{Lie}, Fig. 14, right shows the great improvement in the accuracy related to the prediction of both the principal stress values and the rotations.

Following this, we look at the ML model performance in connection with the approach $DG1_{Eul}$ described in Table 7. Unlike the case with $DG1_{Lie}$, Fig. 15, left, shows that the loss function values start to increase after around 800 epochs, however, without improving the training accuracy. Therefore, the adopted trained ML model corresponds to 750 epochs. In comparison with BB_{Eul}, Fig. 15, right, shows that considering $DG1_{Eul}$ leads to great improvement in the accuracy of principal stress values prediction. However, no improvement in the predicted rotation accuracy can be reached.

To this end, Fig. 16 introduces a comparison between $DG1_{Comp}$ $DG1_{Lie}$ and $DG1_{Eul}$ with regards to the errors in predicting the stress eigenvalues and the rotation-related quantities. In both cases, $DG1_{Lie}$ shows a better

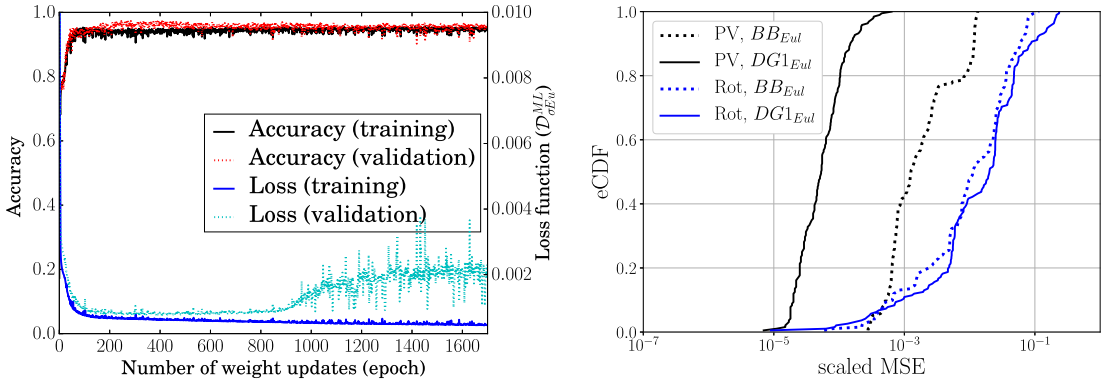

Fig. 15. ML accuracy and loss function values over number of weight updates for **TranIso CP** case using $DG1_{Eul}$ settings (left), and a comparison between $DG1_{Eul}$ and BB_{Eul} with regard to the prediction accuracy of the principal stress values and the Euler angles of the testing dataset (right).

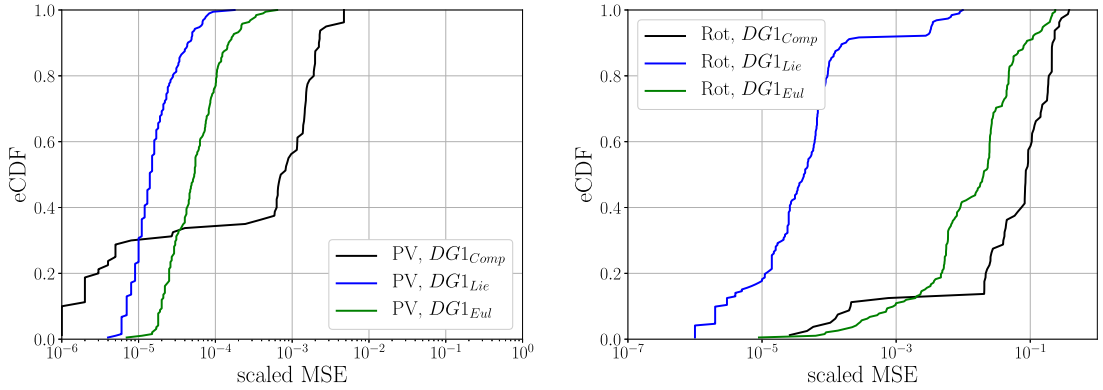

Fig. 16. eCDF vs. scaled MSE within **TranIso CP** of the testing datasets related to the principal stress values (left) and the rotations (right). The applied methods are $DG1_{Comp}$ $DG1_{Lie}$ and $DG1_{Eul}$.

Table 8
Summary of the ML methods applied within $DG2$ approach, see, Section 2.

Method	Description	Obj.
$DG2_{Lie}$	Considers the eigenvalues and the components of the skew-symmetric rotation tensor (Lie Algebra) with the directed graph ($DG2$) in Fig. 1	$\mathcal{D}^{ML}_{\sigma Lie}$
$DG2_{Eul}$	Considers the eigenvalues and Euler angles of tensors with the directed graph ($DG2$) in Fig. 1	$\mathcal{D}^{ML}_{\sigma Eu}$

performance, especially in predicting the rotations, where it is far more accurate than the other two schemes. This is followed by $DG1_{Eul}$, whereas for some of the experiments, $DG1_{Comp}$ leads to good accuracy in Rot and PV predictions.

*5.4. Applying **DG2**-graph to transversely-isotropic-elastic crystal plasticity (**TranIso CP**)*

The following discussion focuses on the application of $DG2$-approach, illustrated in Fig. 1, with the approaches $DG2_{Lie}$ and $DG2_{Eul}$ explained in Table 8 as well as in Section 2.

$DG2$-approach compasses two ML models: The FFNN model applied to the history-independent *elastic step* and the LSTM as a RNN applied to the *plastic step*. This split allows to distinguish between the elasticity-related rotations and that related to activation of the slip systems within crystal plasticity. Fig. 17, left, depicts the eCDF vs. scaled MSE curves of the *elastic step* in connection with the testing dataset. This shows that $DG2_{Lie}$ yields slightly

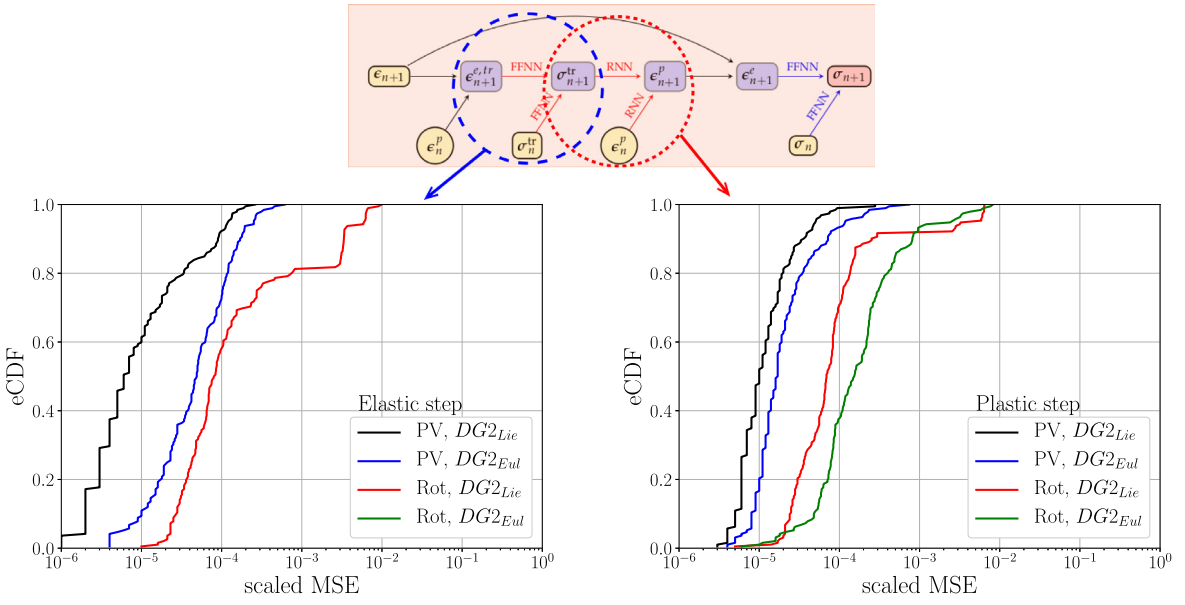

Fig. 17. eCDF vs. scaled MSE within **TranIso CP** of the testing datasets within the directed-graph $DG2$. The *elastic step* (left) shows the accuracy in predicting the principal stress values (PV) and rotation-related quantities (Rot) of $DG2_{Lie}$ and $DG2_{Eul}$ (the curve "Testing Rot (Eu)" is too close to zero). The *plastic step* (right) compares the performance of $DG2_{Lie}$ and $DG2_{Eul}$.

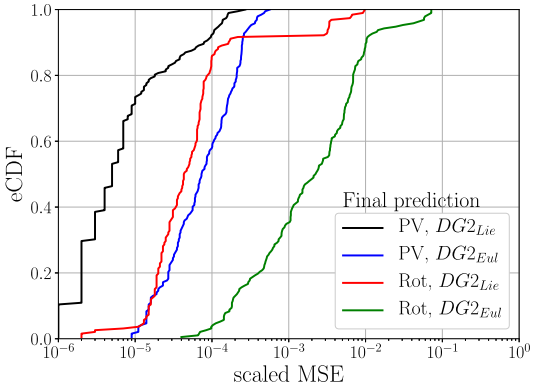

Fig. 18. eCDF vs. scaled MSE within **TranIso CP** of the testing datasets within the informed, directed graph $DG2$. A comparison between the performance of $DG2_{Lie}$ and $DG2_{Eul}$ in prediction of the final principal stress values (PV) and the rotation-related quantities (Rot).

more accurate approximation of the principal stress values (PV) than $DG2_{Eul}$. However, the rotation prediction (Rot) of $DG2_{Eul}$ is far more accurate and the errors are close to zero. On the other hand, Fig. 17, right, shows the eCDF vs. scaled MSE curves of the *plastic step*, where it is obvious that $DG2_{Lie}$ is more accurate than $DG2_{Eul}$ with regard to the accuracy of the predicted eigenvalues and rotations.

Following this, we compare between $DG2_{Lie}$ and $DG2_{Eul}$ with regard to the accuracy in predicting the eigenvalues and rotation entities of the total stress $\boldsymbol{\sigma}_{n+1}$ as a final step in the $DG2$-approach. This is considered a *second elastic step* (history-independent), thus, the FFNN can be applied. The *second elastic step* has the same graph structure as the first elastic step, used to predict the trial stress $\boldsymbol{\sigma}^{tr}_{n+1}$, i.e. both have an elastic strain and stress history as an input and an updated stress as an output. Therefore, the same trained FFNN can be used in the *second elastic step*, i.e. no new training is needed in this step. As a final stress prediction within the $DG2$-approach, Fig. 18 shows that $DG2_{Lie}$ yields more accurate predictions than $DG2_{Eul}$ of both the eigenvalues of the stress (PV) and the related rotation components (Rot).

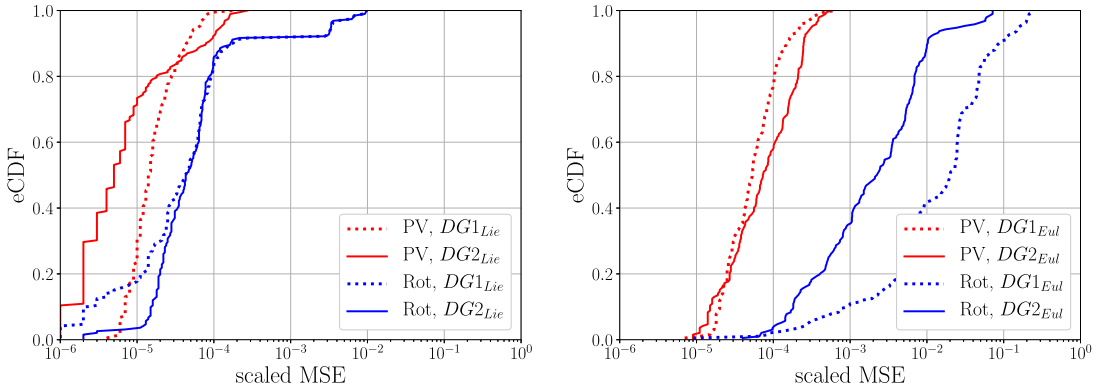

Fig. 19. A comparison between $DG1_{Lie}$ and $DG2_{Lie}$ (left) and between $DG1_{Eul}$ and $DG2_{Eul}$ (right) with regard to the prediction accuracy of the principal stress values (PV) and the rotation-related components of the testing datasets within **TranIso CP**.

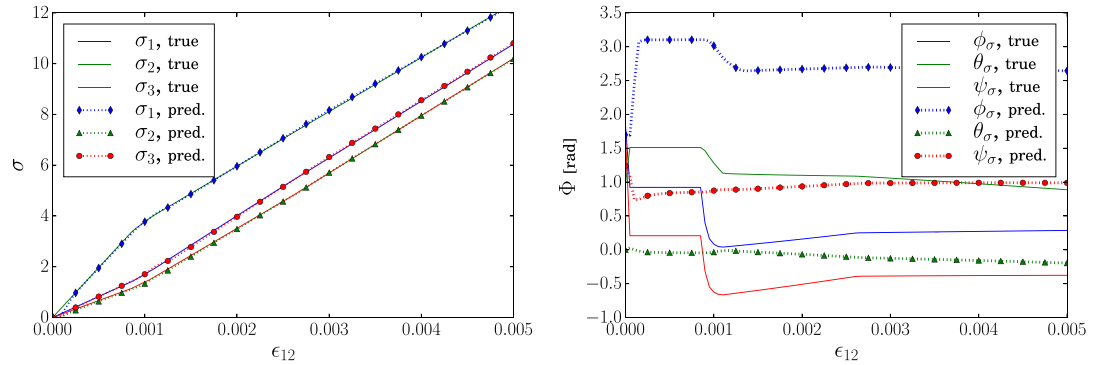

Fig. 20. A comparison between the true and the predicted stress eigenvalues via the stress–strain trajectory (left) and a comparison between the true and predicted Euler angles via the angle-strain trajectory (right) for an experiment from the testing dataset considering $DG1_{Comp}$ within **TranIso CP**.

For sake of completeness and comparison between the simple informed, directed graph ($DG1$) and the more detailed elasto-plasticity graph ($DG2$), we compare between $DG1_{Lie}$ and $DG2_{Lie}$ as well as between $DG1_{Eul}$ and $DG2_{Eul}$ with regard to the accuracy in predicting the rotations and the principal stress values. Fig. 19, left, shows that $DG1_{Lie}$ and $DG2_{Lie}$ yield close accuracy in predicting the entities of the skew-symmetric rotation matrix (Rot), where $DG1_{Lie}$ tends to be more accurate in cases for eCDF<0.4. For the stress eigenvalues (PV) prediction, $DG2_{Lie}$ shows more accuracy, except for cases of eCDF>0.4. However, Fig. 19, right, shows significant improvement in the Euler angle (Rot) prediction when using $DG2_{Eul}$ instead of $DG1_{Eul}$, whereas $DG2_{Eul}$ is slightly less accurate than $DG1_{Eul}$ in predicting the eigenvalues of the stress (PV).

5.5. Discussion of the results

The discussion and comparisons in Sections 5.1–5.4 focused on studying the effects of data form (e.g. tensor components or spectral representation), the ML objective "loss" functions, and the graph structure (BB, $DG1$ and $DG2$) on the ML model performance (accuracy in predicting the eigenvalues and the rotations). For completeness, we apply the final trained ML-models, i.e. $DG1_{Comp}$, $DG2_{Lie}$ and $DG2_{Eul}$ to experiments from the testing dataset, where the chosen experiments correspond to a mean value of the scaled MSE of each case. The stress–strain trajectories in Fig. 20, left, reflect the very good accuracy of $DG1_{Comp}$ in predicting the stress components and, thus, the corresponding eigenvalues. However, Fig. 20, right, shows the major deficiency of $DG1_{Comp}$, as the predicted rotations are completely inaccurate. Thus, the resulting ML material model based on $DG1_{Comp}$ violates the objectivity condition of material models.

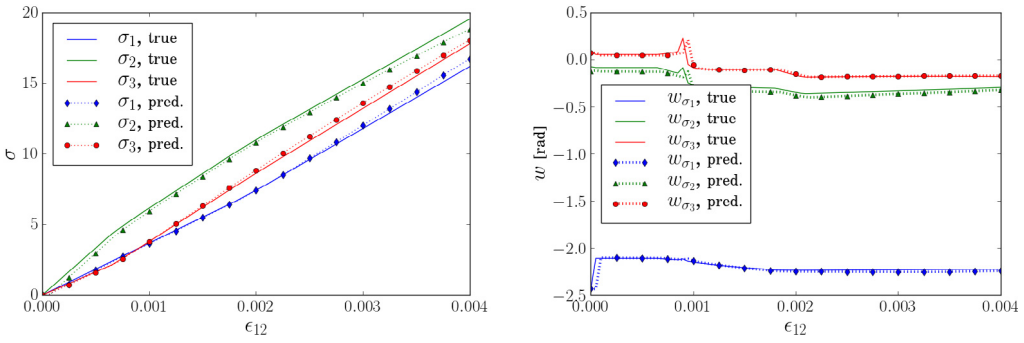

Fig. 21. A comparison between the true and the predicted stress eigenvalues via the stress–strain trajectory (left) and a comparison between the true and predicted skew-symmetric rotation matrix components via the angle-strain trajectory (right) for an experiment from the testing dataset considering $DG2_{Lie}$ within **TranIso CP**.

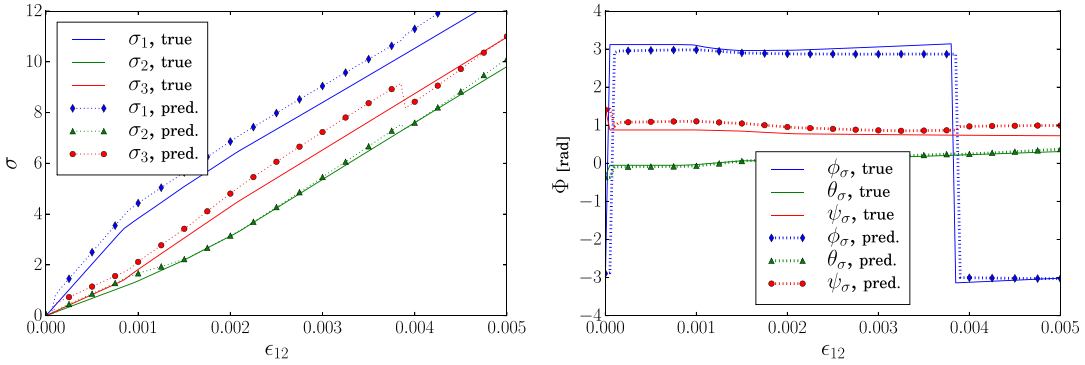

Fig. 22. A comparison between the true and the predicted stress eigenvalues via the stress–strain trajectory (left) and a comparison between the true and predicted Euler angles via the angle-strain trajectory (right) for an experiment from the testing dataset considering $DG2_{Eul}$ within **TranIso CP**.

Changing the data representation to spectral form via eigenvalues and skew-symmetric rotation matrix in Lie algebra, and applying the geodesic on the unit sphere in Lie algebra as a loss function in ML yields a far more accurate prediction of the rotations, while the eigenvalue prediction maintains good accuracy. This can be seen in Fig. 21 that shows the stress–strain trajectories (left) and the angle-strain trajectory (right), $DG2_{Lie}$ approach is applied.

Using the data representation via eigenvalues and Euler angles within $DG2_{Eul}$ approach, Fig. 22 shows the prediction results of eigenvalues (left) and Euler angles (right), where, obviously, the accuracy in rotation prediction is far better than that of $DG1_{Comp}$. It is remarkable in Fig. 22, right, that Euler angles during rotation of the principal stress directions might change suddenly from π and $-\pi$, which adds additional challenges to the training of the model.

Converting the data from spectral form to component form, Fig. 23 shows the stress–strain trajectories in component form. The capability of $DG2_{Lie}$ and $DG2_{Eul}$ in capturing the stress–strain behavior is evident. However, $DG2_{Lie}$ leads to more accurate results.

6. Application to datasets with cyclic strains (loading/unloading)

The test cases and results in Section 5, especially the comparisons in Fig. 19, left, showed that $DG1_{Lie}$ provides a very good compromise between accuracy and computational costs (one-step graph). Thus, for the sake of completeness, we test the capability of $DG1_{Lie}$ approach by considering a dataset that includes loading/unloading cases. In particular, an anisotropic crystal plasticity material model (TranIso CP) with parameters in Table 6 is considered, and the applied loading combinations for each crystal configuration are based on Table 4 . In this, the

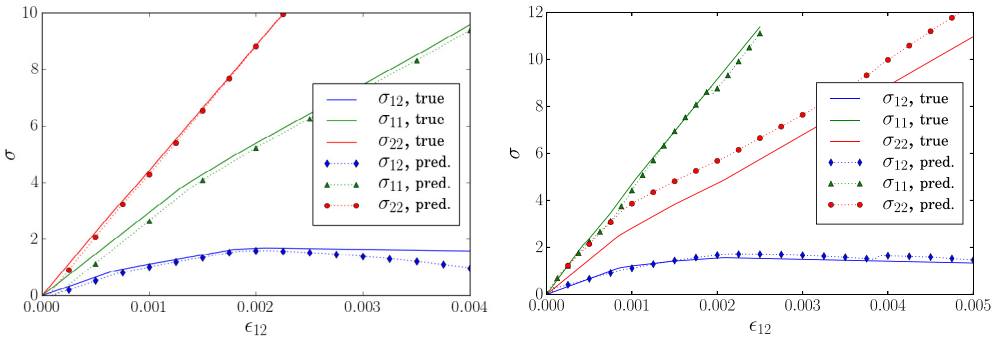

Fig. 23. A comparison between the true and the predicted stress components via the stress–strain trajectory with $DG2_{Lie}$ (left) $DG2_{Eul}$ (right) within **TranIso CP**.

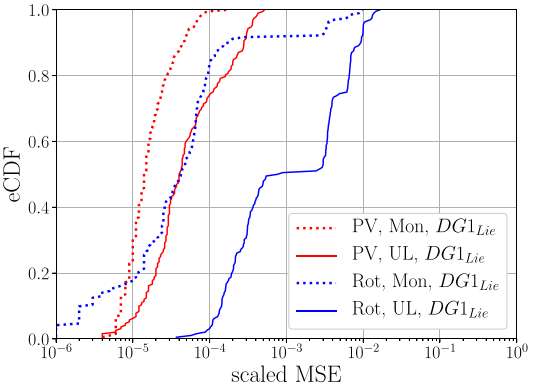

Fig. 24. Application of $DG1_{Lie}$ to datasets that consider cyclic (loading/unloading) vs monolithic loading, where the comparison is based on the prediction accuracy of the principal stress values (PV) and the rotation-related components of the testing dataset within **TranIso CP**.

crystal is subjected to incremental loading (50 increments) of one or more strain components, i.e. $2\,\Delta\epsilon_{11}$, $2\,\Delta\epsilon_{22}$ and $2\,\Delta\epsilon_{12}$, followed by a gradual unloading (50 increments). Beside the 24 loading combinations, 49 different rigid body rotations of the crystal are considered. Thus, the database includes 1176 experiments, which then split into training (792 experiments), validation (192 experiments) and testing (192 experiments) subsets.

Comparing the performance of $DG1_{Lie}$ for monotonic vs cyclic loading (testing dataset), Fig. 24 shows deterioration in prediction accuracy of both the principal stress values (PV) and the skew-symmetric rotation tensor components (Rot). However, the errors (scales-MSE) for PV in all cases still $< 5 \times 10^{-4}$ and for Rot in all cases $< 10^{-2}$.

For illustration, we apply the final trained ML-models, i.e. $DG1_{Lie}$, to experiments from the loading/unloading testing dataset, where the chosen experiments correspond to a mean value of the scaled MSE. The stress–strain trajectories in Fig. 25, left, are related to simultaneous loading/unloading of the aforementioned three strain components, whereas the trajectory in Fig. 25, right, is related to an experiment of which $\Delta\epsilon_{12}$ is the only strain component that is changing. All the mentioned trajectories reflect the good accuracy of $DG1_{Lie}$ in predicting the stress components, which agrees with the findings in the case of monotonic loading.

7. Conclusion

With the aim to generate a frame-invariant machine-learning constitutive model of anisotropic, inelastic material response, the underlying research work presented comprehensive comparisons between several approaches that include changing the data form, using different objective functions in the training and using the informed, directed graph scheme that hybridizes ML models with the classical material laws.

For generation of datasets for training and validation of the ML model, the focus was laid on crystal plasticity behavior, which includes inherent anisotropy in the plastic regime due to activation of the differently-oriented slip

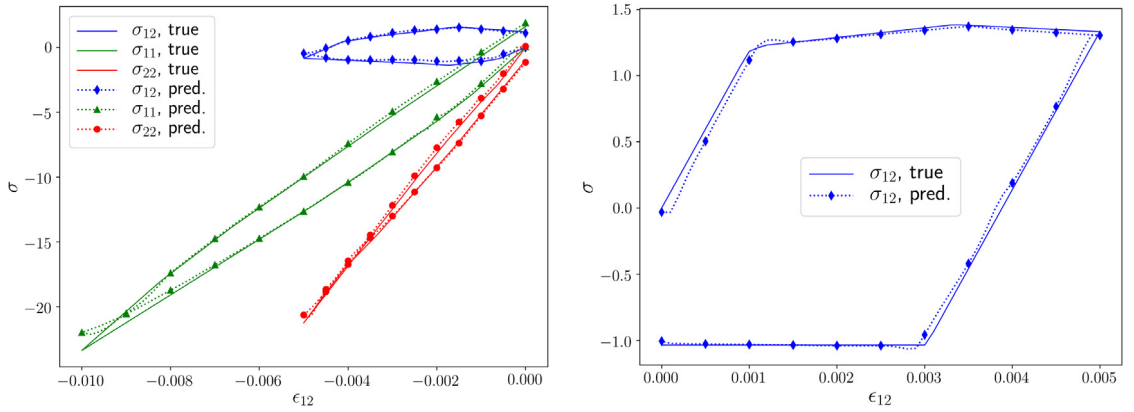

Fig. 25. A comparison between the true and the predicted stress components via the stress–strain trajectory with $DG1_{Lie}$ of two different experiments (left and right) from the testing dataset within **TranIso CP**.

systems, whereas considering anisotropic elasticity contributed to enhancement of the principal stress direction rotations. Different loading combinations beside different rotations were applied to a single f.c.c. crystal, resulted in 1176 experiments with incremental strain-controlled loading (monotonic and cyclic). Consequently, the generated datasets contained the total strain and the total stress components at each time step of each experiment.

Alternative to the tensor representation via component form, spectral decomposition was applied to the different tensors allowing to generate their eigenvalues and eigenvectors. In this, concatenating the orthogonal, normalized eigenvectors yielded the related 3D rotation tensors in $SO(3)$. These rotation matrices with 9 entities could further be parameterized to extract the three Euler angles. As an alternative convenient possibility, we applied logarithmic mapping to represent the rotation matrix via the skew-symmetric counterpart in Lie Algebra $so(3)$, which includes also three different entities. These changes in the data form allowed to consider different metrics for measuring the distance between two 3D rotations in order to design new objective "loss" functions for supervised machine learning. In particular, the proposed loss functions were based on the distance from the identity matrix, the Euclidean distance between the Euler Angles, and geodesic on the unit sphere in Lie algebra.

With regard to the information flow in the machine learning-based, path-dependent crystal plasticity material model, three variations of the graph with different details were tested: (1) The direct relation (BB) graph that used the recurrent neural networks (RNN) to predict the stresses based on the strains as the only input. (2) The "informed" direct relation ($DG1$) with RNN, which used additionally the predicted output history (stress history) to improve the accuracy of the predicted stresses. (3) The elasto-plasticity-based informed, directed graph ($DG2$), relied on splitting the prediction into an elastic and a plastic step, in analogy to the return-mapping plasticity algorithm. The feed-forward neural networks (FFNN) were used in the history-independent elastic steps, whereas RNN was applied in the memory-dependent plastic steps.

In the numerical test cases, we examined the performance of the different loss functions with different data representation forms in the aforementioned three graph designs, in order to figure out which combination yields the most accurate eigenvalues and rotation-related quantities prediction.

Based on this observations, we concludes that the loss function based on geodesic on the unit sphere in Lie algebra together with $DG1$-graph yields the best predictions of rotation, whereas applying $DG2$-graph with this loss function does not improve the rotation prediction, but slightly improves the blind predictions of principal stress values. On the other hand, significant improvement was observed in the rotation prediction accuracy when applying the loss function based on the Euclidean distance between the Euler angles in $DG2$-graph instead of BB or $DG1$-graph.

Our numerical experiments, which consist of both monotonic and cyclic loading cases, revealed that using the components of the tensors as input for supervised machine learning (instead of the spectral decomposition forms) may yield erroneous prediction of the principal directions of tensors, which is crucial in anisotropic material modeling.

To further generalize the applications of this research, addition research activities are planned to incorporate predictions of anisotropic responses of elasto-plastic materials in the geometric nonlinear regime and to predict the elasto-viscoplastic of high-strain rate materials.

Acknowledgments

This research is supported by the NSF CAREER grant from Mechanics of Materials and Structures program at National Science Foundation under grant contract CMMI-1846875, the Dynamic Materials and Interactions Program from the Air Force Office of Scientific Research under grant contracts FA9550-17-1-0169 and FA9550-19-1-0318, and the Nuclear Energy University Program from the Department of Energy under grant contract DE-NE0008534. These supports are gratefully acknowledged. The views and conclusions contained in this document are those of the authors, and should not be interpreted as representing the official policies, either expressed or implied, of the sponsors, including the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

Appendix. Elasto-plastic crystal plasticity algorithm for database generation

Within crystal plasticity, we make use of the solution of f.c.c. single-crystal plasticity within small-strain framework to generate the needed database for the machine learning in this paper. In this, the procedure of the "ultimate algorithm", introduced in [16,40], is applied to identify the set of active systems and delivers an exact solution for crystal plasticity, in which the crystals are subjected to tension, shear or mixed tension–shear deformations varying as a ramp function. In the derivation, the total homogeneous strain rate $\dot{\boldsymbol{\epsilon}}$ in a crystal is additively decomposed into an elastic ($\dot{\boldsymbol{\epsilon}}^e$) and a plastic ($\dot{\boldsymbol{\epsilon}}^p$) components as

$$\dot{\boldsymbol{\epsilon}} = \dot{\boldsymbol{\epsilon}}^e + \dot{\boldsymbol{\epsilon}}^p \quad \text{with} \quad \dot{\boldsymbol{\epsilon}}^p = \sum_{\beta} \dot{\gamma}^{(\beta)} \boldsymbol{S}^{(\beta)} \quad \text{and} \quad \boldsymbol{S}^{(\beta)} := \tfrac{1}{2}\left(\boldsymbol{n}^{(\beta)} \otimes \boldsymbol{m}^{(\beta)} + \boldsymbol{m}^{(\beta)} \otimes \boldsymbol{n}^{(\beta)}\right). \tag{26}$$

In Eq. (26), $\dot{\gamma}^{(\beta)}$ denotes the plastic slip rate and $\boldsymbol{S}^{(\beta)}$ is the symmetric part of the Schmid tensor on β-slip systems, whereas $\boldsymbol{n}^{(\beta)}$ represents the unit-normal to the slip plane and $\boldsymbol{m}^{(\beta)}$ is the direction of plastic slip. In this study, the focus is laid on f.c.c. crystals, which possess eight slip planes and three slip directions for each, thus 24 possible forward and backward slip systems. However, the maximum number of possible simultaneous active slip systems is $N = 12$, as when the forward slip direction is active, then the corresponding backward must be latent. Distinguishing between the fixed Cartesian frame (x, y, z) and the crystal reference frame (x_c, y_c, z_c), the crystal orientation can be described by the Euler angles, consisting of a positive rotation θ around y-axis, followed by a positive rotation φ around z-axis.

Considering $\tau^{(\beta)}$ as the Schmid's resolved shear stress for each slip system., the onset of plasticity is associated with the critical stress $\tau_Y^{(\beta)}$ for each slip system. In particular, the yield surfaces $f^{(\beta)}$ for both forward and backward slips can be expressed as

$$f^{(\beta)} = |\tau^{(\beta)}| - \tau_Y^{(\beta)} \quad \text{with} \quad \tau^{(\beta)} = \pm\hat{\boldsymbol{\sigma}} : \boldsymbol{S}^{(\beta)} \quad \text{and} \quad \beta = 1, 2, \ldots, N\,. \tag{27}$$

Following this, the plastic flow relation that considers all activated slip system reads

$$\dot{\boldsymbol{\epsilon}}^p = \sum_{\beta=1}^{2N} \dot{\gamma}^{(\beta)} \frac{\partial f^{(\beta)}}{\partial \hat{\boldsymbol{\sigma}}} = \sum_{\beta=1}^{2N} \dot{\gamma}^{(\beta)} \boldsymbol{S}^{(\beta)}, \tag{28}$$

where the condition for the onset of plastic flow can be expressed via the Karush–Kuhn–Tucker (KKT) conditions as

$$\dot{\gamma}^{(\beta)} \geq 0\,, \quad f^{(\beta)} \geq 0\,, \quad \dot{\gamma}^{(\beta)}\, f^{(\beta)} = 0. \tag{29}$$

Thus, a slip system (β) is considered active if $\dot{\gamma}^{(\beta)} \geq 0$ and $f^{(\beta)} = 0$, whereas $\dot{\gamma}^{(\beta)} = 0$ and $f^{(\beta)} < 0$ corresponds to an inactive system. The set of all active slip systems will be referred to as $\mathcal{I}_{\text{act}}$. Among several hardening laws for elastoplastic crystal plasticity, a convenient linear law can be expressed as

$$\dot{\tau}_Y^{(\beta)} = \hat{h} \sum_{\xi=1}^{2N} \dot{\gamma}^{(\xi)}\,, \tag{30}$$

with $\hat{h}$ being the plastic hardening moduli, assumed in this study as a constant for all slip systems. Eq. (30) shows that the hardening is applied to both the active and the latent slips. Following this, the elastic stress rate can be expressed as

$$\dot{\hat{\boldsymbol{\sigma}}} = \boldsymbol{C}^e \,:\, (\dot{\boldsymbol{\epsilon}} - \dot{\boldsymbol{\epsilon}}^p) \tag{31}$$

with $\boldsymbol{C}^e$ as the elasticity tensor of the f.c.c. lattice. In the numerical solution, we apply the steps of the "ultimate algorithm", in which step-wise incremental strains $\Delta\boldsymbol{\epsilon}$ are applied to the crystal within each time step Δt, i.e. $\Delta\boldsymbol{\epsilon}(\tau) = \kappa\,\Delta\boldsymbol{\epsilon}$ with $0 \leq \kappa = \tau/\Delta t \leq 1$. In this, κ should be tested at each slip system. This is followed by tracking the active slip systems $\mathcal{I}_{\text{act}}$. The procedure can be split into two sub-algorithms, i.e. an elastic predictor algorithm to determine the onset of plastic strain, and a plastic integrator algorithm to determine the linearly-independent active slip systems. For the aim of the underlying work, we present an abstract representation of the elastic predictor algorithm, where more details and alternative approaches can be found in, e.g., [16,33,40,61].

Algorithm (I): Elastic predictor for the small-strain rate-independent crystal plasticity

Initial: Given at time t_k: $\hat{\sigma}_k,\ \boldsymbol{\epsilon}_k = \boldsymbol{\epsilon}^e_k,\ \mathcal{I}_{\text{act, k}}$

Step 1: Apply strain increments: $\Delta\boldsymbol{\epsilon}(\tau) = \boldsymbol{\epsilon}_{k+1} - \boldsymbol{\epsilon}_k = \kappa\,\Delta\boldsymbol{\epsilon}$ at $t_{k+1} = t_k + \Delta t$

Step 2: Compute the trial stress: $\hat{\sigma}^{\text{tr}}_{k+1} = \hat{\sigma}_k + \boldsymbol{C}^e : \Delta\boldsymbol{\epsilon}(\tau)$

Step 3: Assemble trial active slip set $\mathcal{I}^{\text{tr}}_{\text{act}} := \left\{\beta \in \{1, \cdots 2N\} \;\middle|\; f^{(\beta)} = |\hat{\boldsymbol{\sigma}}^{\text{tr}}_{k+1} : \boldsymbol{S}^{(\beta)}| - \tau^{(\beta)}_{Y,k} > 0\right\}$

Step 4: If $\mathcal{I}^{\text{tr}}_{\text{act}} = \emptyset \longrightarrow$ elastic step

$\hat{\sigma}_{k+1} = \hat{\sigma}^{\text{tr}}_{k+1}\ ,\ \tau^{(\beta)}_{Y,k+1} = \tau^{(\beta)}_{Y,k}$

Step 4: If $\mathcal{I}^{\text{tr}}_{\text{act}} \neq \emptyset \longrightarrow$ elasto-plastic step

Step-wise identification of the active slip systems $\mathcal{I}_{\text{act}}$

test $\kappa^{(\beta)}$ for all slip system with $\beta \in \mathcal{I}_{\text{act}}$

compute $\Delta\gamma^{(\beta)}$ (incremental plasti slip) for $\beta \in \mathcal{I}_{\text{act}}$

compute $\boldsymbol{C}^{ep}$ (elasto-plastic tangent moduli)

update $\hat{\sigma}_{k+1} = \hat{\sigma}_k\ ,\ \tau^{(\beta)}_{Y,k+1} = \tau^{(\beta)}_{Y,k}\ ,\ \mathcal{I}_{\text{act, k+1}}$

References

[1] Trenton Kirchdoerfer, Michael Ortiz, Data-driven computational mechanics, Comput. Methods Appl. Mech. Engrg. 304 (2016) 81–101.

[2] Robert Eggersmann, Trenton Kirchdoerfer, Stefanie Reese, Laurent Stainier, Michael Ortiz, Model-free data-driven inelasticity, Comput. Methods Appl. Mech. Engrg. 350 (2019) 81–99.

[3] Kun Wang, WaiChing Sun, Qiang Du, A cooperative game for automated learning of elasto-plasticity knowledge graphs and models with AI-guided experimentation, Comput. Mech. (2019) 1–33.

[4] Zeliang Liu, C.T. Wu, Exploring the 3D architectures of deep material network in data-driven multiscale mechanics, J. Mech. Phys. Solids 127 (2019) 20–46.

[5] Hang Yang, Xu Guo, Shan Tang, Wing Kam Liu, Derivation of heterogeneous material laws via data-driven principal component expansions, Comput. Mech. (2019) 1–15.

[6] Daniel Charles Drucker, Some implications of work hardening and ideal plasticity, Quart. Appl. Math. 7 (4) (1950) 411–418.

[7] R.J. Green, A plasticity theory for porous solids, Int. J. Mech. Sci. 14 (4) (1972) 215–224.

[8] Andrew Schofield, Peter Wroth, Critical State Soil Mechanics, volume 310, McGraw-Hill London, 1968.

[9] WaiChing Sun, A unified method to predict diffuse and localized instabilities in sands, Geomech. Geoengin. 8 (2) (2013) 65–75.

[10] SeonHong Na, WaiChing Sun, Computational thermo-hydro-mechanics for multiphase freezing and thawing porous media in the finite deformation range, Comput. Methods Appl. Mech. Engrg. 318 (2017) 667–700.

[11] Eric C. Bryant, WaiChing Sun, A micromorphically regularized cam-clay model for capturing size-dependent anisotropy of geomaterials, Comput. Methods Appl. Mech. Engrg. 354 (2019) 56–95.

[12] J. Ghaboussi, J.H. Garrett Jr, Xiping Wu, Knowledge-based modeling of material behavior with neural networks, J. Eng. Mech. 117 (1) (1991) 132–153.
[13] M. Lefik, B.A. Schrefler, Artificial neural network as an incremental non-linear constitutive model for a finite element code, Comput. Methods Appl. Mech. Engrg. 192 (28–30) (2003) 3265–3283.
[14] Kun Wang, WaiChing Sun, A multiscale multi-permeability poroplasticity model linked by recursive homogenizations and deep learning, Comput. Methods Appl. Mech. Engrg. 334 (2018) 337–380.
[15] Marcus Stoffel, Franz Bamer, Bernd Markert, Neural network based constitutive modeling of nonlinear viscoplastic structural response, Mech. Res. Commun. 95 (2019) 85–88.
[16] Ronaldo I. Borja, Helia Rahmani, Discrete micromechanics of elastoplastic crystals in the finite deformation range, Comput. Methods Appl. Mech. Engrg. 275 (2014) 234–263.
[17] Matthew R. Kuhn, WaiChing Sun, Qi Wang, Stress-induced anisotropy in granular materials: fabric, stiffness, and permeability, Acta Geotech. 10 (4) (2015) 399–419.
[18] Kun Wang, WaiChing Sun, A semi-implicit discrete-continuum coupling method for porous media based on the effective stress principle at finite strain, Comput. Methods Appl. Mech. Engrg. 304 (2016) 546–583.
[19] Kun Wang, WaiChing Sun, An updated lagrangian LBM–DEM–FEM coupling model for dual-permeability fissured porous media with embedded discontinuities, Comput. Methods Appl. Mech. Engrg. 344 (2019) 276–305.
[20] Kun Wang, WaiChing Sun, Meta-modeling game for deriving theory-consistent, microstructure-based traction–separation laws via deep reinforcement learning, Comput. Methods Appl. Mech. Engrg. 346 (2019) 216–241.
[21] Zeliang Liu, C.T. Wu, M. Koishi, A deep material network for multiscale topology learning and accelerated nonlinear modeling of heterogeneous materials, Comput. Methods Appl. Mech. Engrg. 345 (2019) 1138–1168.
[22] Raymond H. Myers, Douglas C. Montgomery, Christine M. Anderson-Cook, Response Surface Methodology: Process and Product Optimization using Designed Experiments, John Wiley & Sons, 2016.
[23] Chih-Chung Chang, Chih-Jen Lin, LIBSVM: A library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (3) (2011) 27.
[24] Christian Soize, Roger Ghanem, Cosmin Safta, Xun Huan, Zachary P. Vane, Joseph C. Oefelein, Guilhem Lacaze, Habib N. Najm, Enhancing model predictability for a scramjet using probabilistic learning on manifolds, AIAA J. 57 (1) (2018) 365–378.
[25] Tomonari Furukawa, Genki Yagawa, Implicit constitutive modelling for viscoplasticity using neural networks, Internat. J. Numer. Methods Engrg. 43 (2) (1998) 195–219.
[26] Jamshid Ghaboussi, David A. Pecknold, Mingfu Zhang, Rami M. Haj-Ali, Autoprogressive training of neural network constitutive models, Internat. J. Numer. Methods Engrg. 42 (1) (1998) 105–126.
[27] D.Z. Huang, K. Xu, C. Farhat, E. Darve, Predictive modeling with learned constitutive laws from indirect observations, 2019, arXiv preprint arXiv:1905.12530.
[28] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al., Relational inductive biases, deep learning, and graph networks, 2018, arXiv preprint arXiv:1806.01261.
[29] R. Banerjee, K. Sagiyama, G.H. Teichert, K. Garikipati, A graph theoretic framework for representation, exploration and analysis on computed states of physical systems, Comput. Methods Appl. Mech. Engrg. 351 (2019) 501–530.
[30] Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Maosong Sun, Graph neural networks: A review of methods and applications, CoRR abs/1812.08434 (2018) URL http://arxiv.org/abs/1812.08434.
[31] Xiang Song Li, Yannis F. Dafalias, Anisotropic critical state theory: role of fabric, J. Eng. Mech. 138 (3) (2011) 263–275.
[32] Robert J. Asaro, Crystal plasticity, J. Appl. Mech. 50 (4b) (1983) 921–934.
[33] SeonHong Na, WaiChing Sun, Computational thermomechanics of crystalline rock, Part I: A combined multi-phase-field/crystal plasticity approach for single crystal simulations, Comput. Methods Appl. Mech. Engrg. 338 (2018) 657–691.
[34] Arnd Koeppe, Franz Bamer, Bernd Markert, An efficient Monte Carlo strategy for elasto-plastic structures based on recurrent neural networks, Acta Mech. 230 (9) (2019) 3279–3293.
[35] Ari L. Frankel, Reese E. Jones, Coleman Alleman, Jeremy A. Templeton, Predicting the mechanical response of oligocrystals with deep learning, 2019, arXiv preprint arXiv:1901.10669.
[36] Yibo Yang, Paris Perdikaris, Adversarial uncertainty quantification in physics-informed neural networks, J. Comput. Phys. 394 (2019) 136–152.
[37] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics informed deep learning (Part I): Data-driven solutions of nonlinear partial differential equations, 2017, arXiv:1711.10561.
[38] Robert Dabrowski, Krzysztof Stencel, Grzegorz Timoszuk, Software is a directed multigraph, in: European Conference on Software Architecture, Springer, 2011, pp. 360–369.
[39] J.C. Simo, T.J.R. Hughes, Computational Inelasticity, Springer-Verlag New York, 1998.
[40] R.I. Borja, Plasticity: Modeling & Computation, Springer Science & Business Media, 2013.
[41] J. Schröder, Theoretische und algorithmische Konzepte zur phänomenologischen Beschreibung anisotropen materialverhaltens, Report Nr.: I-1, Institut of Mechanics (CE), Chair I, University of Stuttgart, 1996.
[42] J. Schröder, P. Neff, Invariant formulation of hyperelastic transverse isotropy based on polyconvex free energy functions, Int. J. Solids Struct. 40 (2) (2003) 401–445.
[43] Jean-Paul Boehler, A simple derivation of representations for non-polynomial constitutive equations in some cases of anisotropy, ZAMM Z. Angew. Math. Mech. 59 (4) (1979) 157–167.
[44] F.C. Park, B. Ravani, Smooth invariant interpolation of rotations, ACM Trans. Graph. 16 (3) (1997) 277–295.

[45] Du Q. Huynh, Metrics for 3D rotations: Comparison and analysis, J. Math. Imaging Vision 35 (2) (2009) 155–164.
[46] Pierre M. Larochelle, Andrew P. Murray, Jorge Angeles, A distance metric for finite sets of rigid-body displacements via the polar decomposition, J. Mech. Des. 129 (8) (2007) 883–886.
[47] Alejandro Mota, WaiChing Sun, Jakob T. Ostien, James W. Foulk Iii, Kevin N. Long, Lie-group interpolation and variational recovery for internal variables, Comput. Mech. 52 (6) (2013) 1281–1299.
[48] M. Ortiz, R.A. Radovitzky, E.A. Repetto, The computation of the exponential and logarithmic mappings and their first and second linearizations, Internat. J. Numer. Methods Engrg. 52 (12) (2001) 1431–1441.
[49] W. Noll, n the continuity of the fluid and solid states, J. Ration. Mech. Anal. 4 (1955) 3–81.
[50] François Chollet, et al., Keras, 2015, https://github.com/fchollet/keras.
[51] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al., Tensorflow: Large-scale machine learning on heterogeneous distributed systems, 2016, arXiv preprint arXiv:1603.04467.
[52] Wes McKinney, et al., Data structures for statistical computing in python, in: PROC. of the 9th PYTHON in SCIENCE CONF., SCIPY 2010, 2010.
[53] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al., Scikit-learn: Machine learning in Python, J. Mach. Learn. Res. 12 (Oct) (2011) 2825–2830.
[54] Sepp Hochreiter, Jürgen Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[55] Mantas Lukoševičius, Herbert Jaeger, Reservoir computing approaches to recurrent neural network training, Comp. Sci. Rev. 3 (3) (2009) 127–149.
[56] Jian-Hua Zhu, Musharraf M. Zaman, Scott A. Anderson, Modeling of soil behavior with a recurrent neural network, Can. Geotech. J. 35 (5) (1998) 858–872.
[57] K. Greff, R.K. Srivastava, J. Koutník, B.R. Steunebrink, J. Schmidhuber, LSTM: A search space odyssey, IEEE Trans. Neural Netw. Learn. Syst. 28 (10) (2017) 2222–2232.
[58] F.A. Gers, J. Schmidhuber, F. Cummins, Learning to forget: continual prediction with LSTM, in: 1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470), vol. 2, 1999, pp. 850–855.
[59] Maurice George Kendall, et al., The Advanced Theory of Statistics, second ed., Charles Griffin and Co., Ltd., London, 1946.
[60] J.E. Gentle, Computational Statistics, Springer, ISBN: 978-0-387-98145-1, 2009.
[61] J. Schröder, C. Miehe, Aspects of computational rate-independent crystal plasticity, Comput. Mater. Sci. 9 (1) (1997) 168–176.