Available online at www.sciencedirect.com
ScienceDirect

Comput. Methods Appl. Mech. Engrg. 384 (2021) 113952

Computer methods in applied mechanics and engineering

www.elsevier.com/locate/cma

An FE–DMN method for the multiscale analysis of short fiber reinforced plastic components

Sebastian Gajek, Matti Schneider, Thomas Böhlke[*]

Karlsruhe Institute of Technology (KIT), Institute of Engineering Mechanics, Germany

Received 8 February 2021; received in revised form 4 May 2021; accepted 18 May 2021
Available online xxxx

Abstract

In this work, we propose a fully coupled multiscale strategy for components made from short fiber reinforced composites, where each Gauss point of the macroscopic finite element model is equipped with a deep material network (DMN) which covers the different fiber orientation states varying within the component. These DMNs need to be identified by linear elastic precomputations on representative volume elements, and serve as high-fidelity surrogates for full-field simulations on microstructures with inelastic constituents.

We discuss how to extend direct DMNs to account for varying fiber orientation, and propose a simplified sampling strategy which significantly speeds up the training process. To enable concurrent multiscale simulations, evaluating the DMNs efficiently is crucial. We discuss dedicated techniques for exploiting sparsity and high-performance linear algebra modules, and demonstrate the power of the proposed approach on an injection molded quadcopter frame as a benchmark component. Indeed, the DMN is capable of accelerating two-scale simulations significantly, providing possible speed-ups of several magnitudes.
© 2021 Published by Elsevier B.V.

Keywords: Micromechanics; Computational homogenization; Multiscale simulation; Deep material networks; Laminates; Short fiber reinforced composites

1. Introduction

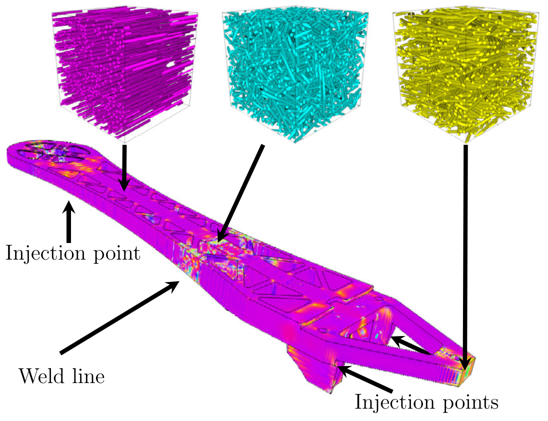
1.1. Problem setting

Injection molded short fiber reinforced components are frequently used for industrial applications as they combine favorable mechanical properties, free formability and short cycle times. As a result of the injection molding process, the fiber orientation and the fiber volume fraction may vary continuously within the component. Characterizing all possible orientation states for such materials is an arduous and expensive task, both experimentally and by simulative means.

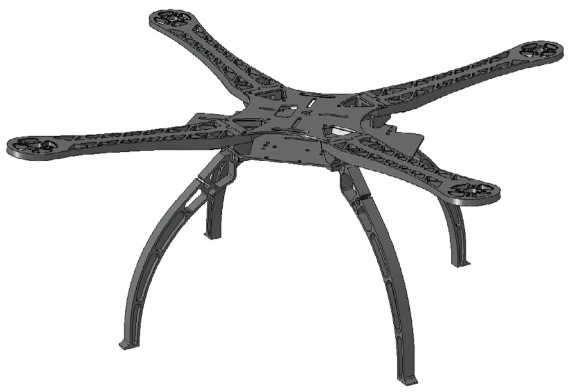
To illustrate the complexity to be handled routinely, Fig. 1(a) shows a quadcopter frame arm made from short fiber reinforced polyamide with a local fiber orientation determined by an injection molding simulation, see Section 5 for details. The color scale encodes the local fiber orientation tensor. Magenta corresponds to a unidirectional, cyan to an

* Corresponding author.
E-mail address: thomas.boehlke@kit.edu (T. Böhlke).

https://doi.org/10.1016/j.cma.2021.113952
0045-7825/© 2021 Published by Elsevier B.V.



(a) Injection molded quadcopter arm



(b) Quadcopter frame geometry [1]

Fig. 1. Injection molded quadcopter arm (a) with local fiber orientation: magenta, cyan and yellow correspond to unidirectional, isotropic and planar isotropic fiber orientation states. Quadcopter frame used as benchmark geometry (b) for the component scale simulation.

isotropic and yellow to a planar isotropic fiber orientation state. We observe that, for the majority of the quadcopter arm, the fibers are almost aligned. Moreover, we encounter isotropic and planar isotropic fiber orientations in areas where weld lines have formed. As weld lines correspond to weak spots in the structure, it is critical to account for such regions accurately in mechanical simulations.

1.2. State of the art

Fig. 1(b) represents an example for a component with a spatially varying complex microstructure. A monolithic finite element (FE) simulation of such a structure which *resolves* the heterogeneities is not feasible with the current computational power. If the microscopic heterogeneities fluctuate on a scale that is much smaller than the size of the component, homogenization methods may be used for obtaining so-called effective material models which account for the physical mechanisms of complex and highly heterogeneous microstructures. Effective models emerge naturally by solving a partial differential equation, the cell problem, on a suitable microstructure. For linear material models, the effective properties may be precomputed and cached. In this way, it is possible to compute the mechanical response of microstructured components.

Treating inelastic materials is more difficult, as the internal variables live naturally on the microstructure, and cannot be "homogenized" to the macroscopic scale, in general. FE^2 methods, introduced by Renard–Marmonier [2] and subsequently refined [3–6], offer a solution by furnishing each Gauss point of the macroscopic finite element model with a microstructure on which the cell problem is solved, accounting for the evolution of the internal variables on the microscopic scale. To speed up the solution process on the microscale, FFT-based approaches [7–9] may be used, giving rise to the FE–FFT [10–12] method in the concurrent multiscale context. Despite recent progress in computational efficiency and parallel computing strategies, concurrent multiscale methods with full-field models on the microscopic scale are typically too computationally demanding for industrial use.

To mitigate the computational burden of concurrent multiscale methods, the microscale problem is considered as a parametric partial differential equation which needs to be solved repeatedly (for slightly different input parameters), and model order reduction techniques may be utilized. Motivated by classical mean-field methods [13,14], Dvorak and co-workers [15–17] introduced the transformation field analysis (TFA). The TFA applies to small strain (visco-)plastic material models and assumes the inelastic strains to be piece-wise uniform on specific subdomains, accounting for the resulting elastic deformations via strain localization tensors. In this way, effective models with a finite number of internal variables arise, see also Chaboche et al. [18]. Furthermore, Liu and co-workers [19–21] introduced the self-consistent clustering analysis (SCA), which is similar in spirit to the TFA, but exploits the Hashin–Shtrikman variational principle [22–24] and is not per se limited to small strain (visco-)plasticity. Nevertheless, when considered as discretization methods, both TFA and SCA show a slow convergence rate in terms of the number of clusters [25] which is rooted in the weak approximation capabilities of piecewise uniform functions [26].

Inspired by recent variational estimates for nonlinear materials [27], the non-uniform transformation field analysis [28] (NTFA) relies upon non-uniform inelastic basis functions, permitting the approximation errors of the fields to be made as small as desired. However, the difficulty is transferred to prescribing suitable evolution equations [29,30] for the reduced inelastic strains, which should be independent of the basis, cheap to evaluate and consistent upon refinement. Possible remedies are Taylor series expansions of the force potential [31–33], mixed variational principles [34,35] or dedicated "reducible" models [36], giving rise to the $\text{FE}^{2\text{R}}$ (R as reduced) method [37].

As an alternative to methods which approximate the solution of micromechanical problems on unit cells, it is possible to approximate the effective properties directly. Typically, this comes at the cost of operating on a high-dimensional domain of interest. Data driven methods, especially artificial neural networks (ANNs), are predestined for such approximation tasks. There is a number of works considering training ANNs to approximate the effective elastic energy of a medium, see Yvonnet and coworkers [38–40]. Furthermore, the regularity of the effective stress facilitates the direct approximation of the stress–strain relationship of inelastic problems, see the works of Jadid [41], Penumadu–Zhao [42] or Srinivasu et al. [43] for different approaches. Motivated by natural language processing, recurrent neural networks (RNN) may provide a framework for incorporating history dependence into the approximation of the stress–strain relationship, see Mozaffar et al. [44] and Koeppe et al. [45]. Gorji and coworkers [46] demonstrated that RNN are able to capture effects such as the Bauschinger effect, permanent softening or latent hardening in the context of elastoplasticity. RNN may also be employed in a multiscale setting, see Ghavamian–Simone [47], Xu et al. [48] and Wu et al. [49]. Using artificial neural networks accompanied by an on-the-fly switching to a reduced order model in a two-scale simulation was investigated by Fritzen et al. [50]. A shortcoming of such data driven methods appears to be that their predictive quality appears low far away from the training domain, and accounting for the inherent physics, i.e., monotonicity and thermodynamic consistency, may be difficult.

Liu et al. [51,52] proposed a data driven modeling approach based on an explicit microstructure model consisting of hierarchical laminates. More precisely, for a K-phase microstructure, Liu et al. consider a K-ary tree structure of laminates with fixed direction of lamination and intermittent rotations. In analogy to deep artificial neural networks, they called such an identified surrogate model a deep material network (DMN). Instead of approximating the effective energy or the stress–strain relationship, DMNs are sought to approximate the effective stiffness of a fixed microstructure considered as a function of the input stiffness tensors of the constituents. For the parameter identification, they rely upon automatic differentiation and stochastic gradient descent, as typical for training artificial neural networks. After training, the hierarchical laminate can be applied to inelastic problems, even at finite strains, and the approximation accuracy is rather impressive.

Discarding the intermittent rotations, Gajek et al. [53] introduced direct DMNs, which are based on laminates with variable direction of lamination. Direct DMNs enable a faster and more robust identification process compared to the indirect DMNs of Liu et al. [51,52], and also compare favorably for inelastic constituents. Furthermore, Gajek et al. [53] motivated the training on linear elastic data and generalization to the nonlinear regime by showing that, to first order in the strain rate, the effective inelastic behavior of composite materials is determined by linear elastic localization. As a byproduct, Gajek et al. [53] established that deep material networks inherit thermodynamic consistency and stress–strain monotonicity from their phases. This property is crucial, as it ensures the effective models to inherit stabilizing numerical properties, like strong convexity, from the phases of the composite. Deep material networks were augmented by cohesive zone models by Liu [54].

Recently, Liu et al. [55] proposed a transfer learning approach to treat fiber reinforced composites by DMNs. More precisely, they proposed to interpolate the parameters of already identified DMNs, corresponding to different fiber orientation states. In this work, we go beyond this a posteriori approach, and seek DMNs covering the entire spectrum of fiber orientations arising in such a fiber reinforced component.

1.3. Contribution and outline

In this article, we investigate a multiscale methodology for direct deep material networks, which covers all possible variations of the second-order fiber orientation tensors and permits concurrent multiscale simulations with

DMNs at the Gauss point level. Similar to the FE^2, FE-FFT and the FE^{2R} methods, we call the ensuing method the FE-DMN method.

As point of departure, we consider direct deep material networks [53] and restrict to microstructures without micro-orientations, i.e., microstructures comprising isotropic phases. To account for a spatially varying fiber orientation, see Section 2, we augment direct DMNs by the fiber orientation interpolation concept introduced by Köbler et al. [56]. To this end, we assume that the local fiber volume fractions of the individual laminates in the hierarchy are independent of the local fiber orientation. Therefore, it suffices to fix the fiber volume fractions and to interpolate the lamination directions only. In contrast to Liu et al. [55], we do not consider a *transfer learning* strategy, i.e., to interpolate already identified models, but propose an a priori interpolation strategy. More precisely, we investigate deep material networks which explicitly depend on the fiber orientation, and identify the optimal model parameters *jointly*.

For this purpose, we utilize high-fidelity microstructures of fiber reinforced composites [57] which permit us to routinely cover the possible fiber orientations at industrial filler fraction and fiber aspect ratio. For the sake of simplicity, we assume that both fiber volume fraction and fiber length are fixed for all generated microstructures. We sample the linear elastic training data from up to 31 microstructure realizations. We also improve upon the previous sampling strategy [51,52] for the constituents' stiffness tensors used in the offline training. We show by example that it is sufficient to cover those stiffnesses which arise as possible material tangents on the microscopic scale.

For component scale simulations of industrial problems, it is necessary to optimize the user-defined material models based upon the identified DMNs, see Section 3.

We take special care to ensure that the interpolated DMN generalizes accurately to the inelastic regime. To this end, 78 additional microstructure realizations were generated, exclusively for the inelastic validations. We compute the stress response of each of the 109 generated microstructure representations by an FFT-based computational homogenization [7,8] code and compare the former to the predicted stress response of the interpolated direct DMN. The validation results show that with a maximum error of 5.5%, the DMN is capable of predicting the effective stress of all investigated 109 discrete fiber orientation states sufficiently. We refer to Section 4 for details.

To show that our approach is applicable to state of the art engineering computations, we consider the entire process chain of a quadcopter frame, see Fig. 1(b) and Section 5. We conduct an injection molding simulation where we assume the fiber volume fraction to be homogeneous throughout the part, map the spatially varying fiber orientations upon a finite element mesh and conduct a two-scale simulation using the identified DMN surrogate model. We implement the DMN as an implicit user-material (UMAT) subroutine in ABAQUS only relying upon the provided software interfaces. In contrast, Liu et al. [55] employed DMNs in an explicit finite element analysis.

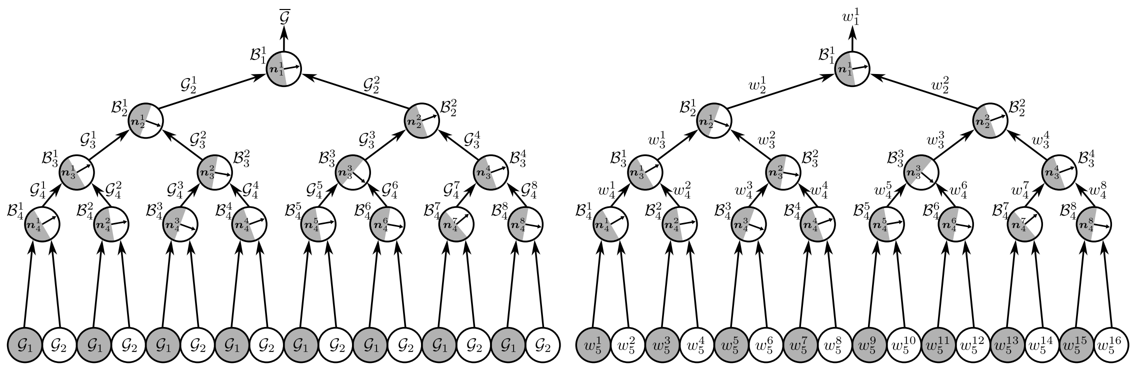
The quadcopter consists of four arms made from injection molded short fiber reinforced polyamide, two base plates which are made from aluminum and four legs which are made from pure polyamide. The simulation model of the quadcopter consist of more than two million elements resulting in almost ten million degrees of freedom. In more than 1.9 million elements, a deep material network is integrated implicitly at every Gauss points, accounting for the local microstructure information in the simulation.

Last but not least, we discuss the computational costs accompanied by our approach in Section 6, and demonstrate that the educated guess of the potential computational power of DMNs made in the conclusion of Liu et al. [55] was too pessimistic.

2. Direct deep material networks for variable fiber orientation

In this section, we extend direct deep material networks to short fiber reinforced composite microstructures parameterized by the second order fiber orientation tensor. The main technical tool is the fiber orientation interpolation technique, introduced by Köbler et al. [56], which we apply on the node level of the deep material network.

First, we recall the basics of direct deep material networks and the fiber orientation triangle. Subsequently, we combine both concepts.



(a) Material propagation (b) Weight propagation

Fig. 2. Schematic illustration of the material (a) and weight (b) propagation in a two-phase direct DMN of depth four (the input level is not counted).

2.1. Direct deep material networks

Let $\mathcal{GSM}$ denote the set of all generalized standard materials (GSM) [58]. Then, any two-phase periodic microstructure $Y \subseteq \mathbb{R}^d$ in d spatial dimensions gives rise to the (nonlinear) homogenization function

$$\mathcal{M}_Y : \mathcal{GSM} \times \mathcal{GSM} \to \mathcal{GSM}, \quad (\mathcal{G}_1, \mathcal{G}_2) \mapsto \bar{\mathcal{G}}, \tag{2.1}$$

which maps two input GSMs to the effective GSM that emerges by solving the cell problem of first order homogenization, see Gajek et al. [53] for details. Homogenization functions may be regarded as the basic objects of studying micromechanics at small strains.

In general, evaluating homogenization functions requires significant computational resources. Only for special microstructures, the evaluation can be performed with minimal effort. An example for such a microstructure is given by a two-phase laminate, uniquely characterized by a direction $\boldsymbol{n}$ of lamination and the volume fractions c_1 and c_2 of the two phases.

A direct deep material network is defined as a hierarchy of such two-phase laminates, see Fig. 2. The concept was originally introduced by Liu and co-workers [51,52] for laminates with *fixed* direction of lamination, but additional rotation layers, and simplified by Gajek et al. [53]. By combining laminates in a hierarchical manner, the resulting homogenization function

$$\mathcal{DMN}_Y : \mathcal{GSM} \times \mathcal{GSM} \to \mathcal{GSM} \tag{2.2}$$

may be rather complex and, by a judicious choice of the involved laminates, may be used as an approximation of the homogenization function (2.1) corresponding to the original microstructure Y, which is significantly less demanding to evaluate.

On a more formal level, a direct DMN is a perfect, ordered, rooted binary tree of depth K, where a two-phase laminate $\mathcal{B}_k^i$ is assigned to each node of the tree. We reserve the letter k for labeling the depth of a node, whereas the horizontal index is consistently indexed by the letter i. Our layer count only comprises the laminate layers, and the input is counted separately. Thus, the DMN comprises $2^K - 1$ laminate nodes. For a two-phase DMN of depth K, the homogenization function $\mathcal{DMN}_Y$

$$\bar{\mathcal{G}} = \mathcal{DMN}_Y(\mathcal{G}_1, \mathcal{G}_2) \tag{2.3}$$

is defined recursively by traversing the binary tree from the leaves, at level K, to the root

$$\bar{\mathcal{G}} = \mathcal{G}_1^1 \quad \text{with} \quad \mathcal{G}_k^i = \mathcal{B}_k^i(\mathcal{G}_{k+1}^{2i-1}, \mathcal{G}_{k+1}^{2i}), \quad k = 1 \ldots K, \; i = 1 \ldots 2^{k-1}. \tag{2.4}$$

Input materials are assigned in an alternating fashion, i.e.,

$$\mathcal{G}^i_{K+1} = \begin{cases} \mathcal{G}_1, & i \text{ odd}, \\ \mathcal{G}_2, & i \text{ even}, \end{cases} \tag{2.5}$$

holds. We refer to Fig. 2(a) for a schematic.

Liu et al. [51,52] noticed that parameterizing the involved laminates by the volume fractions c_1 and c_2 is not optimal. Indeed, if one of the volume fractions is zero, the entire corresponding sub-tree will have no further influence. It is more convenient to parameterize the laminates' volume fractions by assigning pairs of weights w^{2i-1}_{K+1} and w^{2i}_{K+1} to each laminate on the input level K. These weights should be non-negative and sum to unity. Then, by traversing the binary tree from the leaves to the root, the weights on the kth level are computed by the sum of weights of the respective laminates on the previous level, i.e.,

$$w^i_k = w^{2i-1}_{k+1} + w^{2i}_{k+1} \tag{2.6}$$

holds, see Fig. 2(b). The volume fractions c_1 and c_2 of each laminate $\mathcal{B}^i_k$ are then computed by normalization

$$c_1 = \frac{w^{2i-1}_{k+1}}{w^{2i-1}_{k+1} + w^{2i}_{k+1}} \quad \text{and} \quad c_2 = 1 - c_1. \tag{2.7}$$

For fixed tree topology, a direct DMN is uniquely determined by the directions of lamination, one for each laminate, and the weights of the input layer. These free parameters are identified based on linear elastic precomputations, the so-called **offline training**, see Section 3.1. Once the free parameters are identified, the DMN can be applied to nonlinear and inelastic materials. This **online evaluation** is described in Section 3.2.

2.2. *The fiber orientation triangle*

Suppose a fiber orientation state is given in terms of a fiber orientation distribution (FOD) function ρ, which specifies the probability to find fibers in direction $\boldsymbol{p}$. Advani–Tucker [59] introduced the second order fiber orientation tensor [59]

$$\boldsymbol{A}_2 = \int \boldsymbol{p} \otimes \boldsymbol{p}\, \rho(\boldsymbol{p}) \mathrm{d}A(\boldsymbol{p}) \tag{2.8}$$

as a compact measure for the current fiber orientation state. Despite its limited information content, its compact form makes it the typical quantity of interest for commercial injection molding simulations [60]. Higher-order moments of the FOD are then estimated by closure approximations, see Montgomery–Smith et al. [61].

The tensor $\boldsymbol{A}_2$ is symmetric and positive definite with unit trace. Consequently, only five independent parameters are involved. In terms of an eigenvalue decomposition

$$\boldsymbol{A}_2 = \boldsymbol{Q}\, \mathrm{diag}\,(\lambda_1, \lambda_2, \lambda_3)\, \boldsymbol{Q}^\mathsf{T}, \tag{2.9}$$

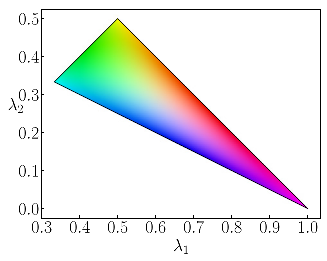
where the matrix $\boldsymbol{Q} \in SO(3)$ is orthogonal and the eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3$ are sorted in a descending order, the fiber orientation tensor $\boldsymbol{A}_2$ may be described by two parameters λ_1 and λ_2 which satisfy the inequalities

$$\frac{1}{3} \leq \lambda_1 \leq 1 \quad \text{and} \quad \frac{1-\lambda_1}{2} \leq \lambda_2 \leq \min\,(\lambda_1, 1 - \lambda_1). \tag{2.10}$$

Thus, up to an orthogonal transformation, every tensor $\boldsymbol{A}_2$ corresponds to a point in the triangle described by the inequalities (2.10), see Fig. 3. In this article, we follow Köbler et al. [56] and use the CMYK coloring scheme for encoding different fiber orientations as shown in Fig. 3.

The manufacturing process induces local variations of the fiber orientation of short fiber reinforced thermoplastic components. Thus, for component-scale simulations, these variations need to be accounted for by the material models. If the fiber orientation state is described in terms of the second-order fiber orientation tensor $\boldsymbol{A}_2$, a family of effective material models, one for each such tensor $\boldsymbol{A}_2$, needs to be supplemented.

By general covariance considerations, two fiber orientation states which differ only by an orthogonal transformation should give rise to effective material responses which differ only by this orthogonal transformation. Consequently, if the considered fiber orientation states are parameterized by the second order fiber orientation tensor, the essentially different fiber orientation states will be parameterized by the points inside the fiber orientation triangle (2.10).



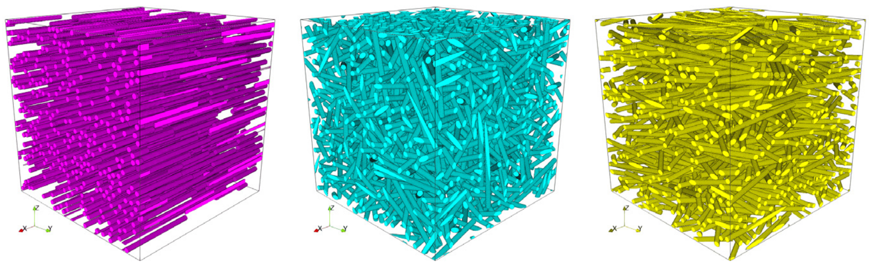

(a) $\lambda_1 = 1.0,\ \lambda_2 = 0.0$
(b) $\lambda_1 = 0.\overline{33},\ \lambda_2 = 0.\overline{33}$
(c) $\lambda_1 = 0.495,\ \lambda_2 = 0.495$

Fig. 3. Fiber orientation reference triangle showing the two largest eigenvalues of the fiber orientation tensor. The three extreme cases, i.e., (a) unidirectional, (b) isotropic and (c) planar isotropic fiber orientation are shown.

Thus, the material models to be identified are parameterized by a two-dimensional continuum. Furthermore, a certain continuity of the effective response of the material model, considered as a function of the fiber orientation tensor, is expected. Indeed, changing the fiber orientation only slightly is expected to change the effective mechanical response only slightly as well, at least for non-critical loading. Unfortunately, the typical multiscale approach based on representative volume elements is unable to leverage this continuity. Indeed, although the effective material response depends continuously on the fiber orientation tensor, the representative volume element does not. Indeed, due to the stochastic nature of such fiber-filled volume elements, infinitely many *different* representative volume elements may be used to give rise to the same effective response.

In particular, this reasoning has the following implication. Suppose that we furnish each point (λ_1, λ_2) in the fiber orientation triangle (2.10) with a corresponding representative volume element $Y_{\lambda_1\lambda_2}$. Even if all these elements have the same size, the function $(\lambda_1, \lambda_2) \mapsto Y_{\lambda_1\lambda_2}$ will not be continuous in any useful way.

As an alternative, Köbler et al. [56] proposed a fiber orientation interpolation procedure on the level of effective stresses. This idea avoids the difficulty of interpolating internal variables which live in different locations for different microstructures. However, this approach comes at a price: the number of stress evaluations is tripled by this approach. Indeed, for any fiber orientation state, the stress response associated to the three corners of the interpolating triangle needs to be evaluated.

2.3. Fiber orientation interpolation of deep material networks

Due to their specific structure, deep material networks may overcome the difficulties mentioned at the end of the previous section. Indeed, for fixed tree topology, the internal variables of the individual phases live on identical locations, also for different DMNs. Indeed, in any case, the internal variables are tied to the materials on the lowest level of the tree, see Fig. 2.

Of course, if DMNs are identified *independently* for each point in the fiber orientation triangle (2.10), the parameters of the DMN need not depend continuously on the fiber orientation tensor $\boldsymbol{A}_2$. Still, it appears reasonable to identify the DMN's parameters *jointly* over all fiber orientations in the fiber orientation triangle.

More precisely, we consider DMNs which are parameterized by points (λ_1, λ_2) inside the fiber orientation triangle. As observed by Liu et al. [51,52] and Gajek et al. [53], the DMN's weights after training are directly linked to the constituent volume fractions of the underlying microstructure. The former does not come as a surprise, since the effective elastic behavior is determined by the volume fraction to first order, see Milton [62, Ch. 14] or Torquato [63, Sec. 20.2.2], and the DMN is fitted on linear elastic data alone. Since, for generating the training data, we assume a constant volume fraction, independent of the fiber orientation, we seek weights which are *independent* of the fiber orientation.

In order to interpolate the lamination directions $\boldsymbol{n}_k^i$ on the fiber orientation triangle, we parameterize each normal $\boldsymbol{n}_k^i \in \boldsymbol{S}^2$ by spherical coordinates

$$\boldsymbol{n}_k^i = \begin{bmatrix} \sin\left(\alpha_k^i\right)\cos\left(\beta_k^i\right) \\ \sin\left(\alpha_k^i\right)\sin\left(\beta_k^i\right) \\ \cos\left(\alpha_k^i\right) \end{bmatrix} \tag{2.11}$$

Table 1
Shape functions used for the orientation interpolation.

Linear	$M = 3$	$\phi_1 = \varphi_1$	$\phi_2 = \varphi_2$	$\phi_3 = \varphi_3$
Tri-linear	$M = 4$	$\phi_1 = \varphi_1 - 9\varphi_1\varphi_2\varphi_3$ $\phi_4 = 27\varphi_1\varphi_2\varphi_3$	$\phi_2 = \varphi_2 - 9\varphi_1\varphi_2\varphi_3$	$\phi_3 = \varphi_3 - 9\varphi_1\varphi_2\varphi_3$
Quadratic	$M = 6$	$\phi_1 = \varphi_1\,(2\varphi_1 - 1)$ $\phi_4 = 4\varphi_1\varphi_2$	$\phi_2 = \varphi_2\,(2\varphi_2 - 1)$ $\phi_5 = 4\varphi_1\varphi_3$	$\phi_3 = \varphi_3\,(2\varphi_3 - 1)$ $\phi_6 = 4\varphi_2\varphi_3$

with angles $\alpha_k^i \in [0, \pi]$ and $\beta_k^i \in [0, 2\pi]$. Then, we interpolate the angles α_k^i and β_k^i on the fiber orientation triangle (2.10) by a global (finite element) shape function. Please note the difference to Köbler et al. [56], who rely upon linear interpolation on a subtriangulation of the fiber orientation triangle.

Particularly compact expressions for the finite element shape functions are obtained by transforming the parameters λ_1 and λ_2 to barycentric coordinates φ_1, φ_2 and φ_3, i.e., via solving the linear system

$$\begin{bmatrix} 1 & 1/3 & 1/2 \\ 0 & 1/3 & 1/2 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \varphi_1 \\ \varphi_2 \\ \varphi_3 \end{bmatrix} = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ 1 \end{bmatrix}, \tag{2.12}$$

see e.g., Vince [64]. We collect the parameters of the polynomial shape functions in a vector $\vec{\phi} = [\phi_1, \ldots, \phi_M]$, where M denotes the number of shape functions. Then, the interpolated angles may be expressed as

$$\alpha_k^i(\lambda_1, \lambda_2) = \vec{p}_k^{\,i\,\mathsf{T}}\vec{\phi}(\lambda_1, \lambda_2) \quad \text{and} \quad \beta_k^i(\lambda_1, \lambda_2) = \vec{q}_k^{\,i\,\mathsf{T}}\vec{\phi}(\lambda_1, \lambda_2) \tag{2.13}$$

in terms of the parameter vectors $\vec{p} = [p_1, \ldots, p_M] \in \mathbb{R}^M$ and $\vec{q} = [q_1, \ldots, q_M] \in \mathbb{R}^M$. In this article, we investigate linear, tri-linear and quadratic shape functions, see Table 1.

To sum up, extending DMNs to account for varying fiber orientation reduces to increasing the number of unknown parameters. Indeed, instead of identifying the angles α_k^i and β_k^i, the parameter vectors $\vec{p}_k^{\,i}$ and $\vec{q}_k^{\,i}$ are sought, in addition to the unknown weights w_{K+1}^i.

3. Implementation

In this section, we explain how to evaluate linear and nonlinear homogenization functions of an interpolated direct DMN efficiently.

3.1. Offline training

The goal of the offline training is to identify the free parameters of the DMN, namely the weights and the vectors $\vec{p}_k^{\,i}$ and $\vec{q}_k^{\,i}$ used for interpolating the angles (2.13), which we collect in "long" vectors

$$\vec{p} = \left[\vec{p}_K^{\,1}, \vec{p}_K^{\,2}, \ldots \vec{p}_K^{\,2^{K-1}}, \vec{p}_{K-1}^{\,1}, \vec{p}_{K-1}^{\,2}, \ldots, \vec{p}_{K-1}^{\,2^{K-2}}, \ldots, \vec{p}_2^{\,1}, \vec{p}_2^{\,2}, \vec{p}_1^{\,1}\right] \in \left(\mathbb{R}^M\right)^{2^K-1} \tag{3.1}$$

and

$$\vec{q} = \left[\vec{q}_K^{\,1}, \vec{q}_K^{\,2}, \ldots \vec{q}_K^{\,2^{K-1}}, \vec{q}_{K-1}^{\,1}, \vec{q}_{K-1}^{\,2}, \ldots, \vec{q}_{K-1}^{\,2^{K-2}}, \ldots, \vec{q}_2^{\,1}, \vec{q}_2^{\,2}, \vec{q}_1^{\,1}\right] \in \left(\mathbb{R}^M\right)^{2^K-1}. \tag{3.2}$$

We insert the parameters of laminates on level K first and add the parameters of laminates for decreasing level index in their corresponding order. We enforce the non-negativity constraint on the weights

$$w_{K+1}^i \geq 0 \tag{3.3}$$

by defining $w_{K+1}^i = \langle v_i \rangle_+$ in terms of unconstrained weights v_i, $i = 1, \ldots, 2^K$. Here, $\langle\cdot\rangle : \mathbb{R} \to \mathbb{R}_{\geq 0}$, $x \mapsto \max(0, x)$ denotes the Macauley bracket. By collecting v_i in a vector $\vec{v} = \left[v_1, \ldots, v_{2^K}\right] \in \mathbb{R}^{2^K}$, we represent the DMN's linear elastic homogenization function in the form

$$\bar{\mathbb{C}} = \mathcal{DMN}_\Lambda^{\mathcal{L}}\left(\mathbb{C}_1, \mathbb{C}_2, \lambda_1, \lambda_2, \vec{p}, \vec{q}, \vec{v}\right). \tag{3.4}$$

$\mathcal{DMN}_\Lambda^{\mathcal{L}}$ maps the input stiffnesses $\mathbb{C}_1$ and $\mathbb{C}_2$, the fiber orientation parameters λ_1 and λ_2 and the unknown fitting parameters $\vec{p}$, $\vec{q}$ and $\vec{v}$ to the DMN's effective stiffness. The specific binary tree structures of the DMN can be

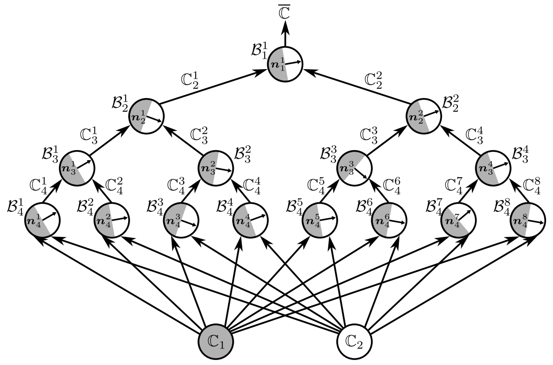

$\bar{\mathbb{C}}$
$\mathcal{B}_1^1$
$\mathbb{C}_2^1$
$\mathbb{C}_2^2$
$\mathcal{B}_2^1$
$\mathcal{B}_2^2$
$\mathbb{C}_3^1$
$\mathbb{C}_3^2$
$\mathbb{C}_3^3$
$\mathbb{C}_3^4$
$\mathcal{B}_3^1$
$\mathcal{B}_3^2$
$\mathcal{B}_3^3$
$\mathcal{B}_3^4$
$\mathbb{C}_4^1$ $\mathbb{C}_4^2$ $\mathbb{C}_4^3$ $\mathbb{C}_4^4$ $\mathbb{C}_4^5$ $\mathbb{C}_4^6$ $\mathbb{C}_4^7$ $\mathbb{C}_4^8$
$\mathcal{B}_4^1$ $\mathcal{B}_4^2$ $\mathcal{B}_4^3$ $\mathcal{B}_4^4$ $\mathcal{B}_4^5$ $\mathcal{B}_4^6$ $\mathcal{B}_4^7$ $\mathcal{B}_4^8$
$\mathbb{C}_1$
$\mathbb{C}_2$

Fig. 4. Schematic illustration of the stiffness propagation in a two-phase direct DMN of depth four.

exploited to evaluate $\mathcal{DMN}_\Lambda^\mathcal{L}$ efficiently. To this end, we assign the input stiffnesses $\mathbb{C}_1$ and $\mathbb{C}_2$ to the laminates of level K in an alternating fashion. The directions of lamination are interpolated on the fiber orientation triangle based on the given parameters λ_1, λ_2, $\vec{p}$ and $\vec{q}$, see Section 2.3. The input stiffnesses are homogenized at level K for each laminate independently. In the next step, the homogenized stiffnesses serve as the input for level $K-1$ and so forth, until the root is reached, giving rise to the DMN's effective stiffness $\bar{\mathbb{C}}$. The process of propagating stiffnesses from the Kth level to the root is illustrated in Fig. 4.

Thus, computing the effective stiffness of a DMN reduces to computing a sequence of effective stiffnesses of two-phase laminates. According to Section 9.5 in Milton's book [65], the linear elastic homogenization function of a laminate

$$\mathbb{C}_k^i = \mathcal{B}_k^i(\mathbb{C}_{k+1}^{2i-1}, \mathbb{C}_{k+1}^{2i}) \tag{3.5}$$

may be determined by solving the equation

$$\left(\mathbb{P}(\boldsymbol{n}_k^i) + \alpha\left[\mathbb{C}_k^i - \alpha\mathbb{I}_\text{s}\right]^{-1}\right)^{-1} = c_1\left(\mathbb{P}(\boldsymbol{n}_k^i) + \alpha\left[\mathbb{C}_{k+1}^{2i-1} - \alpha\mathbb{I}_\text{s}\right]^{-1}\right)^{-1} + c_2\left(\mathbb{P}(\boldsymbol{n}_k^i) + \alpha\left[\mathbb{C}_{k+1}^{2i} - \alpha\mathbb{I}_\text{s}\right]^{-1}\right)^{-1} \tag{3.6}$$

for the effective stiffness $\mathbb{C}_k^i$. Here, $\mathbb{I}_\text{s} : \text{Sym}(d) \rightarrow \text{Sym}(d)$ denotes the identity on $\text{Sym}(d)$, the set of symmetric $d \times d$ matrices, and $\mathbb{P} : \text{Sym}(d) \rightarrow \text{Sym}(d)$ stands for a projection operator, which depends on the direction of lamination $\boldsymbol{n}_k^i$, and reads

$$(\mathbb{P}(\boldsymbol{n}))_{mnop} = \frac{1}{2}(n_m\delta_{no}n_p + n_n\delta_{mo}n_p + n_m\delta_{np}n_o + n_n\delta_{mp}n_o) - n_mn_nn_on_p \tag{3.7}$$

in Cartesian coordinates. Here, δ denotes the Kronecker symbol and α is a parameter which needs to be chosen either sufficiently large or suitably small, see Kabel et al. [66].

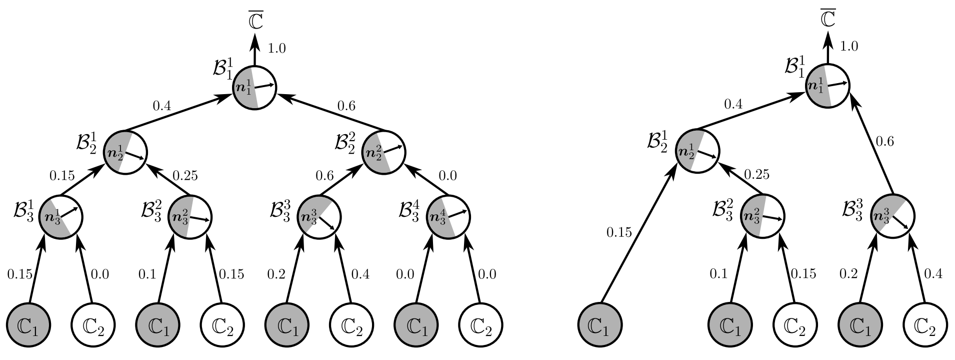
Keeping the former in mind, we turn our attention to the offline training. We sample N_s quadruples of input stiffnesses and fiber orientations $(\mathbb{C}_1^s, \mathbb{C}_2^s, \lambda_1^s, \lambda_2^s)$, generate the respective microstructures and compute the effective stiffnesses $\bar{\mathbb{C}}^s$. We denote the generated training data by a sequence of quintuples $\left\{\left(\bar{\mathbb{C}}^s, \mathbb{C}_1^s, \mathbb{C}_2^s, \lambda_1^s, \lambda_2^s\right)\right\}_{s=1}^{N_s}$ where s enumerates the sample index. The actual sampling process will be discussed in Section 4.4. For the moment, we assume the training data to be given and fixed.

We express the offline training as an optimization problem

$$J(\vec{p}, \vec{q}, \vec{v}) \longrightarrow \min_{\vec{p},\vec{q},\vec{v}} \tag{3.8}$$

involving the objective function

$$J(\vec{p}, \vec{q}, \vec{v}) = \frac{1}{N_b}\sqrt[q]{\sum_{s=1}^{N_b}\left(\frac{\|\bar{\mathbb{C}}^s - \mathcal{DMN}_\Lambda^\mathcal{L}\left(\mathbb{C}_1^s, \mathbb{C}_2^s, \lambda_1^s, \lambda_2^s, \vec{p}, \vec{q}, \vec{v},\right)\|_p}{\|\bar{\mathbb{C}}^s\|_p}\right)^q} + \lambda\left(\sum_{i=1}^{2^K}\langle v_i\rangle_+ - 1\right)^2. \tag{3.9}$$



(a) Perfect binary tree (b) Binary tree with removed laminates

Fig. 5. Binary tree compression to speed up offline training and online evaluation. Laminates corresponding to zero weights are removed from the binary tree.

The quadratic penalty term encodes the mixing constraint

$$\sum_{i=1}^{2^K} w_{K+1}^i = 1. \tag{3.10}$$

We implemented the offline training in PyTorch [67], see Gajek et al. [53], making use of the framework's automatic differentiation capabilities to solve the regression problem (3.8) by means of accelerated stochastic gradient descent methods using mini batches of size N_b. An epoch j consists of evaluating (3.4) for all samples of the respective mini batch, evaluating the loss function (3.9), computing the gradients $\partial J/\partial \vec{p}\left(\vec{p}_j, \vec{q}_j, \vec{v}_j\right)$, $\partial J/\partial \vec{q}\left(\vec{p}_j, \vec{q}_j, \vec{v}_j\right)$ and $\partial J/\partial \vec{v}\left(\vec{p}_j, \vec{q}_j, \vec{v}_j\right)$ by means of automatic differentiation and updating the fitting parameters

$$\vec{p}_{j+1} = \vec{p}_j - \alpha_{\text{p}} \frac{\partial J}{\partial \vec{p}}\left(\vec{p}_j, \vec{q}_j, \vec{v}_j\right), \quad \vec{q}_{j+1} = \vec{q}_j - \alpha_{\text{q}} \frac{\partial J}{\partial \vec{q}}\left(\vec{p}_j, \vec{q}_j, \vec{v}_j\right) \quad \text{and} \quad \vec{v}_{j+1} = \vec{v}_j - \alpha_{\text{v}} \frac{\partial J}{\partial \vec{v}}\left(\vec{p}_j, \vec{q}_j, \vec{v}_j\right). \tag{3.11}$$

During offline training, it may happen that a portion of weights w_{K+1}^i becomes equal to zero, and remains zero due to the vanishing gradient. Liu–Wu [52] removed such sub-trees from the binary tree by deleting nodes and merging the respective subtrees. In this work, we follow Liu–Wu and compress the binary tree to speed-up the training, and eventually, the online evaluation. Fig. 5 shows a schematic of how to remove laminates from the binary tree. The former illustrates a weighted tree with edge weights corresponding to the propagated weights w_{K+1}^i. Remember that the volume fractions of the individual laminates are computed from these weights by normalization.

During training, superfluous laminates are removed from the binary tree to minimize the number of computed laminate homogenizations (3.5). The former happens dynamically during every forward pass. For the example in Fig. 5, only four out of seven laminate homogenization functions are computed in a forward pass, resulting in a speed-up of about 43% compared to a perfect binary tree.

3.2. Online evaluation

Eventually, we seek to employ DMNs to speed up a two-scale simulation, i.e., for every Newton iteration and at every Gauss point of a finite element model, a deep material network needs to be integrated implicitly. At the same time, we want to account for arbitrary eigenvalues λ_1 and λ_2 of the fiber orientation tensor at every Gauss point of the macro simulation. In this work, we extend the fast and flexible solution technique, introduced in Gajek et al. [53], to compute the effective stress of a two-phase DMN. Please note that our *global* approach, which interprets the DMN as a single laminates with a complex kinematics, see Gajek et al. [53] for an illustration, is algebraically equivalent to Liu et al.'s [51,52] *distributed* approach which relies on forward and backward propagations of strains and stresses.

We restrict to the two-potential framework of small-strain isothermal generalized standard materials (GSM) [58]. In d spatial dimensions, a GSM is a quadruple $(Z, \psi, \phi, z_0) \in \mathcal{GSM}$ comprising a (Banach) vector space Z of internal variables, a free energy density $\psi : \mathrm{Sym}(d) \times Z \to \mathbb{R}$, a dissipation potential $\phi : Z \to \mathbb{R} \cup \{+\infty\}$ and $z_0 \in Z$ serves as the initial condition. We assume that the dissipation potential ϕ is proper, convex, lower semi-continuous and satisfies $\phi(0) = 0$ as well as $0 \in \partial\phi(0)$, where $\partial\phi$ denotes the sub-differential of the convex function ϕ. For the complete documentation of the GSM model structure in a continuous setting, see, e.g., Gajek et al. [53].

Suppose that the two GSMs $\mathcal{G}_1 = (Z_1, \psi_1, \phi_1, z_{0,1})$ and $\mathcal{G}_2 = (Z_2, \psi_2, \phi_2, z_{0,2})$ are given. A time discretization of both phases $i \in \{1, 2\}$ by the implicit Euler method gives rise to the formulae for discretized stress and Biot's equation, respectively,

$$\boldsymbol{\sigma}_i^{n+1} = \frac{\partial \psi_i}{\partial \boldsymbol{\varepsilon}}\left(\boldsymbol{\varepsilon}_i^{n+1}, z_i^{n+1}\right) \quad \text{and} \quad \frac{\partial \psi_i}{\partial z}\left(\boldsymbol{\varepsilon}_i^{n+1}, z_i^{n+1}\right) + \partial\phi_i\left(\frac{z_i^{n+1} - z_i^n}{\Delta t}\right) \ni 0. \tag{3.12}$$

Here, $\Delta t = t^{n+1} - t^n$ denotes the time increment and the superscript n refers to the nth time step at time t_n. Due to the time discretization and freezing of the internal variables z_i^n at time t_n, each GSM reduces to a nonlinear elastic material, see Lahellec–Suquet [68]. The condensed free energy $\Psi_i : \mathrm{Sym}(d) \times Z_i \to \mathbb{R}$,

$$\Psi_i\left(\boldsymbol{\varepsilon}_i^{n+1}, z_i^n\right) = \inf_{z_i^{n+1} \in Z_i} \left(\psi_i\left(\boldsymbol{\varepsilon}_i^{n+1}, z_i^{n+1}\right) + \Delta t\, \phi_i\left(\frac{z_i^{n+1} - z_i^n}{\Delta t}\right)\right) \tag{3.13}$$

is solely dependent on the input strain $\boldsymbol{\varepsilon}_i^{n+1}$ and the internal variables z_i^n of the last (converged) time step. Then, the stress response reads

$$\boldsymbol{\sigma}_i^{n+1} = \frac{\partial \Psi_i}{\partial \boldsymbol{\varepsilon}}\left(\boldsymbol{\varepsilon}_i^{n+1}, z_i^n\right). \tag{3.14}$$

For the sake of readability, we omit explicit reference to time step $n+1$. First, let us collect the lamination directions of all laminates in a single vector $\vec{\boldsymbol{n}} \in (\mathbb{R}^d)^{2^K-1}$ with the same ordering that we used for the vectors $\vec{p}$ and $\vec{q}$, i.e.,

$$\vec{\boldsymbol{n}} = \left[n_K^1, n_K^2, \ldots, n_K^{2^{K-1}}, n_{K-1}^1, n_{K-1}^2, \ldots, n_{K-1}^{2^{K-2}}, \ldots, n_2^1, n_2^2, n_1^1\right]. \tag{3.15}$$

We consider the displacement jump vector $\vec{\boldsymbol{a}} \in (\mathbb{R}^d)^{2^K-1}$, which inherits its ordering from $\vec{\boldsymbol{n}}$ and the vector of phase strains $\vec{\boldsymbol{\varepsilon}} = \left[\boldsymbol{\varepsilon}_1, \boldsymbol{\varepsilon}_2, \ldots, \boldsymbol{\varepsilon}_{2^K}\right] \in (\mathrm{Sym}(d))^{2^K}$. By introducing the gradient operator $\boldsymbol{A}_{\lambda_1\lambda_2} : (\mathbb{R}^d)^{2^K-1} \to (\mathrm{Sym}(d))^{2^K}$, we express the phase strains

$$\vec{\boldsymbol{\varepsilon}} = \vec{\bar{\boldsymbol{\varepsilon}}} + \boldsymbol{A}_{\lambda_1\lambda_2}\vec{\boldsymbol{a}} \tag{3.16}$$

w.r.t. the macro strain $\bar{\boldsymbol{\varepsilon}} \in \mathrm{Sym}(d)$ and the unknown displacement jumps $\vec{\boldsymbol{a}}$. Here, the shorthand notation $\vec{\bar{\boldsymbol{\varepsilon}}} = [\bar{\boldsymbol{\varepsilon}}, \bar{\boldsymbol{\varepsilon}}, \ldots, \bar{\boldsymbol{\varepsilon}}] \in (\mathrm{Sym}(d))^{2^K}$ is used. Indeed, for the work at hand, the gradient operator, which encodes the DMN's topology and lamination directions, depends on the fiber orientation parameters λ_1 and λ_2. To illustrate this concept, consider the following example. For a detailed derivation of the special structure of $\boldsymbol{A}_{\lambda_1\lambda_2}$, see Gajek et al. [53]. For a two-phase DMN of depth three as depicted in Fig. 5(a), $\boldsymbol{A}_{\lambda_1\lambda_2}$ takes the following form

$$\boldsymbol{A}_{\lambda_1\lambda_2} = \begin{bmatrix}
-c_3^2 N^1_{3,\lambda_1\lambda_2} & 0 & 0 & 0 & -c_2^2 N^1_{2,\lambda_1\lambda_2} & 0 & -c_1^2 N^1_{1,\lambda_1\lambda_2} \\
c_3^1 N^1_{3,\lambda_1\lambda_2} & 0 & 0 & 0 & -c_2^2 N^1_{2,\lambda_1\lambda_2} & 0 & -c_1^2 N^1_{1,\lambda_1\lambda_2} \\
0 & -c_3^4 N^2_{3,\lambda_1\lambda_2} & 0 & 0 & c_2^1 N^1_{2,\lambda_1\lambda_2} & 0 & -c_1^2 N^1_{1,\lambda_1\lambda_2} \\
0 & c_3^3 N^2_{3,\lambda_1\lambda_2} & 0 & 0 & c_2^1 N^1_{2,\lambda_1\lambda_2} & 0 & -c_1^2 N^1_{1,\lambda_1\lambda_2} \\
0 & 0 & -c_3^6 N^3_{3,\lambda_1\lambda_2} & 0 & 0 & -c_2^4 N^2_{2,\lambda_1\lambda_2} & c_1^1 N^1_{1,\lambda_1\lambda_2} \\
0 & 0 & c_3^5 N^3_{3,\lambda_1\lambda_2} & 0 & 0 & -c_2^4 N^2_{2,\lambda_1\lambda_2} & c_1^1 N^1_{1,\lambda_1\lambda_2} \\
0 & 0 & 0 & -c_3^8 N^4_{3,\lambda_1\lambda_2} & 0 & c_2^3 N^2_{2,\lambda_1\lambda_2} & c_1^1 N^1_{1,\lambda_1\lambda_2} \\
0 & 0 & 0 & c_3^7 N^4_{3,\lambda_1\lambda_2} & 0 & c_2^3 N^2_{2,\lambda_1\lambda_2} & c_1^1 N^1_{1,\lambda_1\lambda_2}
\end{bmatrix} \tag{3.17}$$

with the symmetrization operators w.r.t. the lamination direction $\boldsymbol{n}^i_{k,\lambda_1\lambda_2} = \boldsymbol{n}^i_k(\lambda_1, \lambda_2)$

$$\boldsymbol{N}^i_{k,\lambda_1\lambda_2}\boldsymbol{a} = \frac{1}{2}\left(\boldsymbol{a} \otimes \boldsymbol{n}^i_{k,\lambda_1\lambda_2} + \boldsymbol{n}^i_{k,\lambda_1\lambda_2} \otimes \boldsymbol{a}\right) \tag{3.18}$$

as building blocks. Since we defined $\boldsymbol{n}^i_k(\lambda_1\lambda_2)$ to depend on the parameters λ_1 and λ_2 explicitly, see Section 2.3, the gradient operator depends on the fiber orientation, as well. We account for this situation in our notation, i.e., we write $\boldsymbol{A}_{\lambda_1\lambda_2}$ and $\boldsymbol{N}^i_{k,\lambda_1\lambda_2}$. For the application at hand, i.e., integrating a DMN at every Gauss point during a two-scale simulation, the fiber orientation parameters λ_1 and λ_2 are held fixed. Indeed, we assume that the microstructure does not evolve under the applied load.

Let us define the vector of internal variables of the last converged time step $\vec{z}^{\,n} = \left[z^n_1, z^n_2, z^n_3, \dots, z^n_{2^K}\right] \in \mathcal{Z} := Z_1 \oplus Z_2 \oplus \cdots \oplus Z_1 \oplus Z_2$ and let $\bar{\Psi} : (\mathrm{Sym}(d))^{2^K} \times \mathcal{Z} \to \mathbb{R}$ be the averaged condensed free energy of the flattened laminate

$$\bar{\Psi}(\vec{\boldsymbol{\varepsilon}}, \vec{z}^{\,n}) = \sum_{i=1}^{2^K} w^i_{K+1}\,\Psi_i(\boldsymbol{\varepsilon}_i, z^n_i) \quad \text{where} \quad \Psi_i = \begin{cases} \Psi_1, & i \text{ odd}, \\ \Psi_2, & i \text{ even}, \end{cases} \tag{3.19}$$

alternating between the two given condensed free energies Ψ_1 and Ψ_2. Then, we wish to solve the Euler–Lagrange equation of the DMN

$$\boldsymbol{A}^{\mathsf{T}}_{\lambda_1\lambda_2}\boldsymbol{W}\vec{\boldsymbol{\sigma}}(\vec{\bar{\boldsymbol{\varepsilon}}} + \boldsymbol{A}_{\lambda_1\lambda_2}\vec{\boldsymbol{a}}, \vec{z}^{\,n}) = 0 \tag{3.20}$$

for the unknown displacement jumps $\vec{\boldsymbol{a}}$, where

$$\vec{\boldsymbol{\sigma}} = \left[\boldsymbol{\sigma}_1, \dots, \boldsymbol{\sigma}_{2^K}\right] \in (\mathrm{Sym}(d))^{2^K} \quad \text{with} \quad \boldsymbol{\sigma}_i = \frac{\partial \Psi_i}{\partial \boldsymbol{\varepsilon}}(\boldsymbol{\varepsilon}_i, z^n_i), \tag{3.21}$$

is the vector of phase stresses. For a derivation of the Euler–Lagrange equation (3.20), the reader is referred to Gajek et al. [53]. The strain-wise "mass" matrix $\boldsymbol{W} : \mathrm{Sym}(d)^{2^K} \to \mathrm{Sym}(d)^{2^K}$, representing the linear operator

$$\boldsymbol{W}(\vec{\boldsymbol{\varepsilon}}) = \left(w^1_{K+1}\boldsymbol{\varepsilon}_1, w^2_{K+1}\boldsymbol{\varepsilon}_2, \dots, w^{2^K}_{K+1}\boldsymbol{\varepsilon}_{2^K}\right), \tag{3.22}$$

associates the weight w^i_{K+1}, to the corresponding phase strain $\boldsymbol{\varepsilon}_i$, $i = 1 \dots 2^K$. In matrix notation, $\boldsymbol{W}$ is given by a diagonal matrix with the weights w^i_{K+1} on the diagonal. We solve the Euler–Lagrange equation (3.20) by Newton's method. For an initial guess $\vec{\boldsymbol{a}}_0 \in (\mathbb{R}^d)^{2^K-1}$, the displacement jump vector $\vec{\boldsymbol{a}}$ is iteratively updated $\vec{\boldsymbol{a}}_{j+1} = \vec{\boldsymbol{a}}_j + s_j\,\Delta\vec{\boldsymbol{a}}_j$, where the increment $\Delta\vec{\boldsymbol{a}}_j \in \left(\mathbb{R}^d\right)^{2^K-1}$ solves the linear system

$$\left[\boldsymbol{A}^{\mathsf{T}}_{\lambda_1\lambda_2}\boldsymbol{W}\frac{\partial\vec{\boldsymbol{\sigma}}}{\partial\vec{\boldsymbol{\varepsilon}}}(\vec{\bar{\boldsymbol{\varepsilon}}} + \boldsymbol{A}_{\lambda_1\lambda_2}\vec{\boldsymbol{a}}_j, \vec{z}^{\,n})\boldsymbol{A}_{\lambda_1\lambda_2}\right]\Delta\vec{\boldsymbol{a}}_j = -\boldsymbol{A}^{\mathsf{T}}_{\lambda_1\lambda_2}\boldsymbol{W}\vec{\boldsymbol{\sigma}}(\vec{\bar{\boldsymbol{\varepsilon}}} + \boldsymbol{A}_{\lambda_1\lambda_2}\vec{\boldsymbol{a}}_j, \vec{z}^{\,n}). \tag{3.23}$$

A step size $s_j \in (0, 1]$ strictly less than unity may arise by backtracking. The Jacobian $\partial\vec{\boldsymbol{\sigma}}/\partial\vec{\boldsymbol{\varepsilon}}(\vec{\bar{\boldsymbol{\varepsilon}}} + \boldsymbol{A}_{\lambda_1\lambda_2}\vec{\boldsymbol{a}}_j, \vec{z}^{\,n})$ is a block-diagonal matrix containing the algorithmic tangents of the DMN's input materials, i.e.,

$$\frac{\partial\vec{\boldsymbol{\sigma}}}{\partial\vec{\boldsymbol{\varepsilon}}}(\vec{\boldsymbol{\varepsilon}}, \vec{z}^{\,n}) = \text{block-diag}\left(\frac{\partial^2\Psi_1}{\partial\boldsymbol{\varepsilon}\partial\boldsymbol{\varepsilon}}(\boldsymbol{\varepsilon}_1, z^n_1), \dots, \frac{\partial^2\Psi_{2^K}}{\partial\boldsymbol{\varepsilon}\partial\boldsymbol{\varepsilon}}(\boldsymbol{\varepsilon}_{2^K}, z^n_{2^K})\right). \tag{3.24}$$

Upon convergence, the phase strains $\vec{\boldsymbol{\varepsilon}} = \vec{\bar{\boldsymbol{\varepsilon}}} + \boldsymbol{A}_{\lambda_1\lambda_2}\vec{\boldsymbol{a}}$ and, subsequently, the effective stress

$$\bar{\boldsymbol{\sigma}} = \sum_{i=1}^{2^K} w^i_{K+1}\boldsymbol{\sigma}_i(\boldsymbol{\varepsilon}_i, z^n_i) \tag{3.25}$$

are computed by averaging. Alternatively, the expression

$$\bar{\boldsymbol{\sigma}} = [\mathbb{I}_{\mathrm{s}}, \mathbb{I}_{\mathrm{s}}, \dots, \mathbb{I}_{\mathrm{s}}]^{\mathsf{T}}\boldsymbol{W}\vec{\boldsymbol{\sigma}}(\vec{\bar{\boldsymbol{\varepsilon}}} + \boldsymbol{A}_{\lambda_1\lambda_2}\vec{\boldsymbol{a}}, \vec{z}^{\,n}) \tag{3.26}$$

may be used for computing the volume average of the phase stresses, where $\boldsymbol{W}$ denotes the weight matrix and $[\mathbb{I}_{\mathrm{s}}, \mathbb{I}_{\mathrm{s}}, \dots, \mathbb{I}_{\mathrm{s}}] \in \mathrm{Sym}(d)^{2^K}$, $\mathbb{I}_{\mathrm{s}} : \mathrm{Sym}(d) \to \mathrm{Sym}(d)$ stands for a vector of the identity operators on $\mathrm{Sym}(d)$. In particular, the derivative of the effective stress (3.26) with respect to the macrostrain, the algorithmic tangent of the DMN, admits the representation

$$\mathbb{C}^{\mathrm{algo}} \equiv \frac{\partial\bar{\boldsymbol{\sigma}}}{\partial\bar{\boldsymbol{\varepsilon}}} = [\mathbb{I}_{\mathrm{s}}, \mathbb{I}_{\mathrm{s}}, \dots, \mathbb{I}_{\mathrm{s}}]^{\mathsf{T}}\boldsymbol{W}\left[\frac{\partial\vec{\boldsymbol{\sigma}}}{\partial\vec{\boldsymbol{\varepsilon}}}(\vec{\bar{\boldsymbol{\varepsilon}}} + \boldsymbol{A}_{\lambda_1\lambda_2}\vec{\boldsymbol{a}}, \vec{z}^{\,n}) + \frac{\partial\vec{\boldsymbol{\sigma}}}{\partial\vec{\boldsymbol{\varepsilon}}}(\vec{\bar{\boldsymbol{\varepsilon}}} + \boldsymbol{A}_{\lambda_1\lambda_2}\vec{\boldsymbol{a}}, \vec{z}^{\,n})\boldsymbol{A}_{\lambda_1\lambda_2}\frac{\partial\vec{\boldsymbol{a}}}{\partial\bar{\boldsymbol{\varepsilon}}}\right]. \tag{3.27}$$

Here, the expression $\partial\vec{\boldsymbol{\sigma}}/\partial\bar{\boldsymbol{\varepsilon}}$ encodes the vector of algorithmic tangents of the phase materials

$$\frac{\partial\vec{\boldsymbol{\sigma}}}{\partial\bar{\boldsymbol{\varepsilon}}}(\vec{\boldsymbol{\varepsilon}},\vec{z}^{\,n}) = \left[\frac{\partial^2\Psi_1}{\partial\boldsymbol{\varepsilon}\partial\boldsymbol{\varepsilon}}(\boldsymbol{\varepsilon}_1, z_1^n),\ldots,\frac{\partial^2\Psi_{2K}}{\partial\boldsymbol{\varepsilon}\partial\boldsymbol{\varepsilon}}(\boldsymbol{\varepsilon}_{2K}, z_{2K}^n)\right]. \tag{3.28}$$

To evaluate (3.27), the partial derivatives of the strain jumps with respect to the macrostrain $\partial\vec{\boldsymbol{a}}/\partial\bar{\boldsymbol{\varepsilon}}$ need to be determined first. To this end, differentiating the Euler–Lagrange equation (3.20) with respect to the macrostrain yields the linear system

$$\left[\boldsymbol{A}_{\lambda_1\lambda_2}^{\mathsf{T}}\boldsymbol{W}\frac{\partial\vec{\boldsymbol{\sigma}}}{\partial\bar{\boldsymbol{\varepsilon}}}(\vec{\bar{\boldsymbol{\varepsilon}}}+\boldsymbol{A}_{\lambda_1\lambda_2}\vec{\boldsymbol{a}}_j,\vec{z}^{\,n})\boldsymbol{A}_{\lambda_1\lambda_2}\right]\frac{\partial\vec{\boldsymbol{a}}}{\partial\bar{\boldsymbol{\varepsilon}}} = -\boldsymbol{A}_{\lambda_1\lambda_2}^{\mathsf{T}}\boldsymbol{W}\frac{\partial\vec{\boldsymbol{\sigma}}}{\partial\bar{\boldsymbol{\varepsilon}}}(\vec{\bar{\boldsymbol{\varepsilon}}}+\boldsymbol{A}_{\lambda_1\lambda_2}\vec{\boldsymbol{a}},\vec{z}^{\,n}), \tag{3.29}$$

which needs to be solved for $\partial\vec{\boldsymbol{a}}/\partial\bar{\boldsymbol{\varepsilon}}$. By comparing Eq. (3.29) to Eq. (3.23), we observe that both problems share the same linear operator, but with different right hand sides. When using a direct solver, e.g., a Cholesky decomposition, it is recommended to reuse the matrix decomposition for reasons of efficiency.

To reduce the number of degrees of freedom and, thus, to speed up the solution process, we exploit that some weights become zero during training as explained in the previous section. We learned that in the offline training, we can dynamically build a binary tree with simplified topology but identical effective behavior. This is also true for the online evaluation of the DMN. The DMN's topology is encoded by the gradient operator $\boldsymbol{A}_{\lambda_1\lambda_2}$. Deleting laminate blocks from the binary tree is equivalent to deleting the associated rows and columns of $\boldsymbol{A}_{\lambda_1\lambda_2}$. For the example shown in Fig. 5(b), we obtain a (reduced) gradient operator of the form

$$\boldsymbol{A}_{\lambda_1\lambda_2} = \begin{bmatrix} 0 & 0 & -c_2^2N_{2,\lambda_1\lambda_2}^1 & -c_1^2N_{1,\lambda_1\lambda_2}^1 \\ -c_3^4N_{3,\lambda_1\lambda_2}^2 & 0 & c_2^1N_{2,\lambda_1\lambda_2}^1 & -c_1^2N_{1,\lambda_1\lambda_2}^1 \\ c_3^3N_{3,\lambda_1\lambda_2}^2 & 0 & c_2^1N_{2,\lambda_1\lambda_2}^1 & -c_1^2N_{1,\lambda_1\lambda_2}^1 \\ 0 & -c_3^6N_{3,\lambda_1\lambda_2}^3 & 0 & c_1^1N_{1,\lambda_1\lambda_2}^1 \\ 0 & c_3^5N_{3,\lambda_1\lambda_2}^3 & 0 & c_1^1N_{1,\lambda_1\lambda_2}^1 \end{bmatrix}. \tag{3.30}$$

4. Identifying a DMN surrogate model

This section is dedicated to the identification of the DMN surrogate model. We discuss the pre-processing steps, i.e., finding the necessary resolution and the appropriate size of the volume elements, and investigate the discretization of the fiber orientation triangle. Subsequently, we explain the sampling of the training data, the offline training and the validation of the DMN surrogate model on the fiber orientation triangle. All computations were performed on a workstation equipped with two AMD EPYC 7642 with 48 physical cores each, enabled SMT and 1024 GB of DRAM.

4.1. Short glass fiber reinforced polyamide

For the work at hand, we focus on a short glass fiber reinforced polyamide. We consider E-glass fibers with a length of $L_{\mathrm{f}} = 200$ µm and a diameter of $D_{\mathrm{f}} = 10$ µm. The glass fibers are assumed to be isotropic, linear elastic. The fiber volume fraction is set to $c_{\mathrm{f}} = 16\%$ corresponding to a fiber mass fraction of approx. 30%. The matrix is assumed to be governed by J_2-elastoplasticity, see Chapter 3 in Simo–Hughes [69] with an exponential–linear hardening

$$\sigma_{\mathrm{Y}} = \sigma_0 + k_\infty\varepsilon_{\mathrm{p}} + (\sigma_\infty - \sigma_0)\left(1 - \exp\left(-\frac{k_0 - k_\infty}{\sigma_\infty - \sigma_0}\,\varepsilon_{\mathrm{p}}\right)\right). \tag{4.1}$$

The mechanical properties used in the simulation, taken from Doghri et al. [70], are summarized in Table 2. We rely upon the Sequential Addition and Migration (SAM) method [57] for generating periodic volume elements with prescribed volume fraction and second order fiber orientation tensor. The fiber length L_{f}, fiber diameter D_{f}, fiber volume fraction c_{f} and the axis aligned fiber orientation tensor $\boldsymbol{A}_2$, i.e., λ_1 and λ_2, serve as input parameters for the SAM method. Please note that we only consider fiber orientation states with $\lambda_3 \geq 0.01$, as purely planar fiber orientation states cannot be generated at high filler content essentially for geometric reasons, see Schneider [57] for a discussion.

Table 2
Material parameters for the short glass fiber reinforced polyamide [70].

Matrix	$E = 2.1$ GPa	$\nu = 0.3$	$\sigma_0 = 29$ MPa	$\sigma_\infty = 61.7$ MPa	$k_0 = 10.6$ GPa	$k_\infty = 139$ MPa
Fibers	$E = 72$ GPa	$\nu = 0.22$				

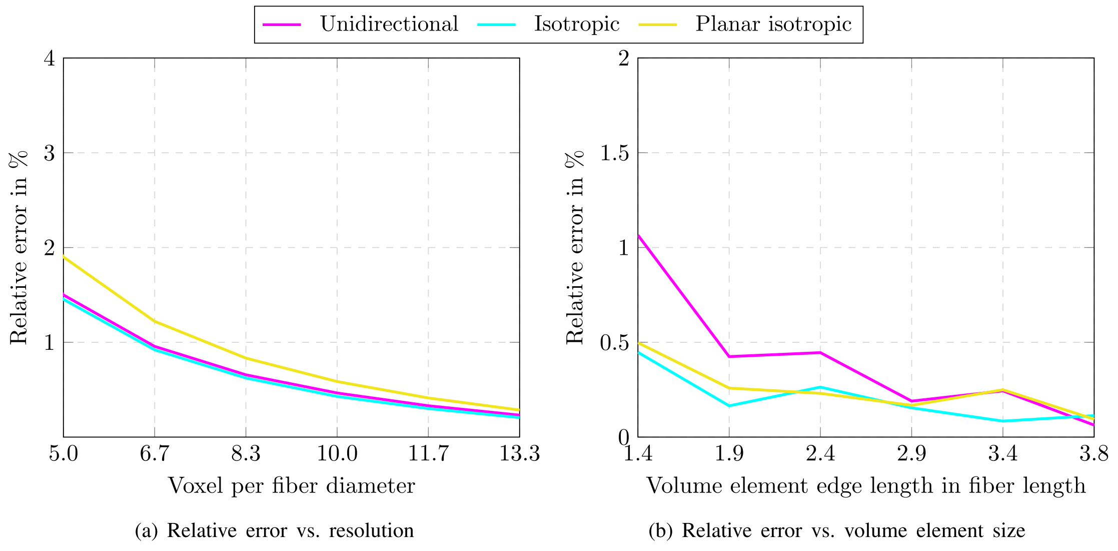


Fig. 6. Relative error of the effective stiffness $\mathbb{C}$ for different resolutions (a) and different volume element sizes (b). We consider the three extreme cases of unidirectional, isotropic and planar isotropic fiber orientation. The reference stiffnesses are obtained for a resolution of 20 voxels per fiber diameter (a) and a volume element with edge length of 7.68 fiber lengths (b).

4.2. On the necessary resolution and the size of the RVE

For a start, we study the resolution necessary to obtain accurate effective properties in the purely elastic case. For this purpose, we consider cubic microstructures with an edge length of $L = 384$ μm, i.e., roughly twice the fiber length of $L_\text{f} = 200$ μm. We compute the effective stiffness with the help of an FFT-based computational micromechanics code [7,8] as described in Schneider [71], using the staggered grid discretization [72,73] and the conjugate gradient solver [74,75].

We consider the extreme orientations individually, i.e., unidirectional, isotropic and planar isotropic fiber orientation as shown in Fig. 3, and vary the resolution from 1.7 to 13.3 voxels per fiber diameter in equidistant steps. This corresponds to volume element discretizations with 64^3 to 512^3 voxels. We measure the error relative to the effective stiffness $\bar{\mathbb{C}}$ and choose a resolution of 20 voxels per fiber diameter, i.e., a discretized by 768^3 voxels, as the reference.

Fig. 6(a) shows the relative error of the effective stiffness computed by the Frobenius norm of the corresponding Voigt matrices. For the crudest resolution, i.e., five voxels per fiber diameter, the relative error is well below 2% for all three considered fiber orientations. Notice that the relative error for the volume elements with planar isotropic fiber orientations is consistently larger than the error for the unidirectional and isotropic orientations. As expected, the relative error decreases with increasing resolution. At a resolution of 6.7 voxels per fiber diameter, the relative errors of the unidirectional and isotropic fiber orientation fall below 1%. For 8.3 voxels per fiber diameter, the relative error of the planar isotropic fiber orientation is below 1%, as well. For this article, we consider a resolution of 6.7 voxels per fiber diameter as sufficient, i.e., relative errors below 1% for isotropic and unidirectional fiber orientation and an error slightly above 1% for the planar isotropic case. We fix this resolution and focus on finding the size of a representative volume element. For a resolution of 6.7 voxels per fiber diameter, we investigate volume elements with edge lengths L ranging from 1.44 up to 3.84 fiber lengths corresponding to volume element discretizations with 192^3 up to 512^3 voxels.

To obtain our reference, we generated a volume element with edge lengths of 7.68 fiber lengths discretized with 1024^3 voxels. As for studying the necessary resolution, we again consider the relative error in the effective stiffness

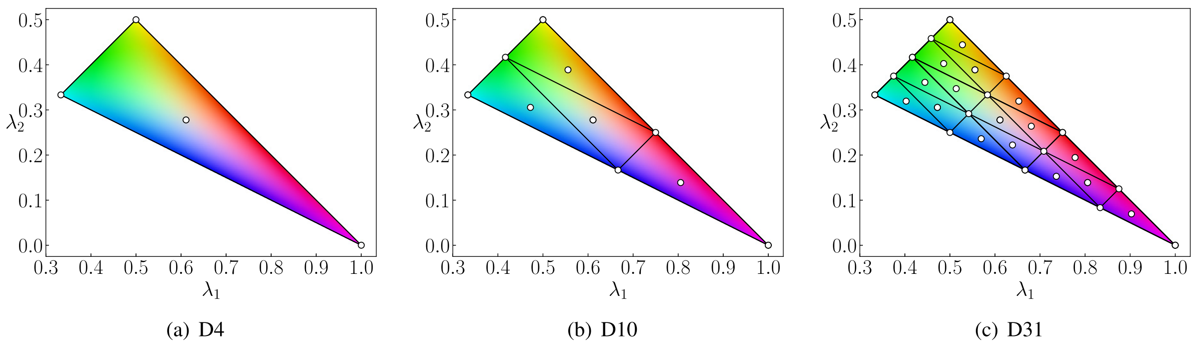


Fig. 7. The investigated fiber orientation discretizations: (a) four, (b) ten and (c) 31 sampling points.

as our error measure. For the volume elements with edge lengths of 1.9 fiber length and above, the relative error is well below 0.5% and does not further decrease significantly for increasing volume element size. For this reason, we consider volume elements with an edge length of $L = 384$ μm as sufficient. To sum up, we finally choose a resolution of 6.7 voxel per fiber length, i.e., a voxel size of 1.5 μm and a discretization with 256^3 voxels for the article at hand.

4.3. Discretization of the fiber orientation triangle

To generate the linear elastic training data, we seek to sample the space of input stiffnesses and fiber orientations uniformly. Apart from sampling the input stiffnesses, it is possible to sample λ_1 and λ_2 as well, e.g., via a low-discrepancy sequence such as the Sobol sequence [76] or via Latin hypercube sampling [77]. Due to the high dimension of the input space, we follow the former approach for generating tuples of input stiffnesses. The considered fiber orientations are parameterized by a two-dimensional space. Its discretization will follow geometric considerations, to be elaborated upon below.

We discretize the fiber orientation triangle by partitioning it into four self-similar triangles, which may be subsequently partitioned, as well. We select the three points on the vertices of each triangle plus the centers of the triangles as sampling points for the parameters λ_1 and λ_2. We start with the full orientation triangle, see Fig. 7(a). The four sampling points, illustrated by four hollow circles, are located at the three corners and the center of the orientation triangle. After the first splitting, the orientation triangle comprises four triangles and ten points, see Fig. 7(b). After dividing the triangles one more time, we arrive at 31 sampling points. For each of these points in fiber orientation space, we generate a single volume element using the SAM method [57] with edge lengths $L = 384$ μm and a discretization with 256^3 voxels, see Section 4.2. In this work, we consider the discretizations shown in Fig. 7, i.e., we discretize the orientation triangle with four, ten and 31 sampling points which we call D4, D10 and D31, respectively.

Choosing the sampling points in a hierarchical manner permits us to re-use already generated volume elements for the next finer discretization. For instance, going from D10 to D31 only requires generating 21 new volume elements. This restriction is not imposed by the SAM [57] algorithm since generating volume elements with the given resolution and edge length is just a matter of milliseconds to seconds. Being able to reuse results from coarser discretizations comes in handy for the inelastic validations we perform in Section 4.6. In Section 4.6, we generate 78 additional volume elements to validate the DMN outside of its training regime and to check if the DMN generalizes sufficiently on the orientation space. To this end, we compare the DMNs predicted effective stress to the effective stress obtained by full field simulations for twelve independent strain paths for every generated volume element and orientation state within the orientation discretization. Obtaining the full field solutions is computationally expensive. Therefore, being able to reuse the full-field solutions from coarser discretizations for validating the finer discretizations comes in handy to keep the validation effort manageable.

4.4. Material sampling

We turn our attention to the sampling of the input stiffnesses. There is some freedom in the selection of appropriate sampling strategies for the training data. For instance, Liu et al. [51], Liu and Wu [52] and Gajek et al. [53], sampled (axis aligned) orthotropic stiffnesses. In the work at hand, we seek to decrease the number of degrees of freedom further, from 17 to 8, by taking into account that the glass fibers are linear elastic and the polyamide matrix is governed by J_2-elastoplasticity. For that reason, we assume that the samples $\mathbb{C}_1$, corresponding to the glass fibers, are isotropic, i.e., the equation

$$\mathbb{C}_1 = 3K_1\,\mathbb{P}_1 + 2G_1\,\mathbb{P}_2 \tag{4.2}$$

holds, where $\mathbb{P}_1 : \mathrm{Sym}(d) \to \mathrm{Sph}(d)$ and $\mathbb{P}_2 : \mathrm{Sym}(d) \to \mathrm{Dev}(d)$ project onto the spherical and deviatoric subspaces of $\mathrm{Sym}(d)$, respectively. Secondly, we assume that the samples corresponding to the polyamide matrix are isotropic minus a rank-one perturbation, i.e.,

$$\mathbb{C}_2 = 3K_2\,\mathbb{P}_1 + 2G_2\left(\mathbb{P}_2 - a\,\boldsymbol{N}' \otimes \boldsymbol{N}'\right). \tag{4.3}$$

Here, the tensor $\boldsymbol{N}' \in \mathcal{N}$ is normalized and deviatoric, i.e.,

$$\mathcal{N} = \{\boldsymbol{N} \in \mathrm{Sym}(d) \mid \mathrm{tr}\,(\boldsymbol{N}) = 0,\ \|\,\boldsymbol{N}\,\|_{\mathrm{F}} = 1\}. \tag{4.4}$$

The structure of the second stiffness $\mathbb{C}_2$ encompasses the possible algorithmic tangents of J_2-elastoplasticity, see Chapter 3 in Simo–Hughes [69].

The set of all considered positive definite stiffness tuples $(\mathbb{C}_1, \mathbb{C}_2)$ may be parameterized via

$$\left(K_1, G_1, K_2, G_2, a, \boldsymbol{N}'\right) \in \mathbb{R}^4_{>0} \times [0, 1) \times \mathcal{N}, \tag{4.5}$$

where K_i and G_i have the dimensions of a Young's modulus and a and $\boldsymbol{N}'$ are dimensionless. Since the latter set is unbounded, we restrict to the subset of elements $\left(K_1, G_1, K_2, G_2, a, \boldsymbol{N}'\right)$ with

$$K_1 = 1\ \mathrm{GPa}, \quad G_1 = 10^{\mathrm{p}_1}\ \mathrm{GPa}, \quad K_2 = 10^{\mathrm{p}_2}\ \mathrm{GPa}, \quad G_2 = 10^{\mathrm{p}_3}\ \mathrm{GPa} \tag{4.6}$$

and exponents $p_1, p_2, p_3 \in [-3, 3]$. By fixing the compression modulus K_1, we removed the redundancy due to homothetic rescaling via $(\mathbb{C}_1, \mathbb{C}_2) \mapsto (\lambda\mathbb{C}_1, \lambda\mathbb{C}_2)$ for $\lambda > 0$. For parameterizing $\boldsymbol{N}'$, we make use of an eigenvalue decomposition $\boldsymbol{N}' = \boldsymbol{Q}\boldsymbol{N}\boldsymbol{Q}^{\mathsf{T}}$ with an orthogonal $\boldsymbol{Q} \in SO(3)$ and a diagonal $\boldsymbol{N} \in \mathcal{N}$ matrix. We parameterize $\boldsymbol{N}$ by spherical coordinates

$$\boldsymbol{N} = \mathrm{diag}\left(\sin(\alpha)\cos(\beta), \sin(\alpha)\sin(\beta), \cos(\alpha)\right) \tag{4.7}$$

ensuring the condition $\|\,\boldsymbol{N}\,\|_{\mathrm{F}} = 1$ to hold. To account for the vanishing trace, $\mathrm{tr}\,(\boldsymbol{N}) = 0$, we eliminate the angle α in Eq. (4.7) and arrive at the parameterization

$$\boldsymbol{N} = \frac{1}{\sqrt{\frac{\cos(\beta)\sin(\beta)+1}{2\cos(\beta)\sin(\beta)+1}}}\mathrm{diag}\left(-\frac{\sqrt{2}\cos(\beta)}{2\cos(\beta)+2\sin(\beta)}, -\frac{\sqrt{2}\sin(\beta)}{2\cos(\beta)+2\sin(\beta)}, \frac{1}{\sqrt{2}}\right) \tag{4.8}$$

in terms of an single remaining angle $\beta \in [0, 2\pi]$. As in Gajek et al. [53], the special orthogonal group is parameterized via an axis–angle representation

$$\boldsymbol{Q} : \mathbb{R}^3 \to \mathbb{R}^3,\ \boldsymbol{x} \mapsto \cos(\theta)\,\boldsymbol{x} + \sin(\theta)\,\boldsymbol{n} \times \boldsymbol{x} + (1 - \cos(\theta))(\boldsymbol{n} \cdot \boldsymbol{x})\boldsymbol{n}, \tag{4.9}$$

for the axis $\boldsymbol{n} = (\sin(\psi)\cos(\varphi), \sin(\psi)\sin(\varphi), \cos(\psi))$, and where the conditions $\theta - \sin(\theta) \in [0, \pi]$, $\psi \in [0, \pi]$ and $\varphi \in [0, 2\pi]$ hold, see Miles [78].

To sum up, we consider the following eight degrees of freedom

$$(a, p_1, p_2, p_3, \beta, \theta, \psi, \phi) \tag{4.10}$$

with their respective domains specified above. To sample the input space evenly, we sample the parameters (4.10) by the Sobol sequence and, subsequently, construct the stiffness tensors $(\mathbb{C}_1, \mathbb{C}_2)$.

Table 3
Number of generated samples and training and validation set sizes.

	D4	D10	D31
Total	800	1000	1550
Per microstructure	200	100	50
Training set	720	900	1395
Validation set	80	100	155

4.5. Offline training

For the offline training, we generate N_s pairs of stiffnesses $\left(\mathbb{C}_1^s, \mathbb{C}_2^s\right)$ by the protocol described in Section 4.4. Then, we assign each stiffness tuple to one of the previously generated volume elements in a cyclic fashion. For instance, for the orientation discretization D4, we assign $\left(\mathbb{C}_1^1, \mathbb{C}_2^1\right)$ to the volume element with fiber orientation $\left(\lambda_1^1, \lambda_2^1\right)$, $\left(\mathbb{C}_1^2, \mathbb{C}_2^2\right)$ to the volume element with $\left(\lambda_1^2, \lambda_2^2\right)$ and $\left(\mathbb{C}_1^5, \mathbb{C}_2^5\right)$ to the volume element with $\left(\lambda_1^1, \lambda_2^1\right)$ and so forth. For every quadruple $\left(\mathbb{C}_1^s, \mathbb{C}_2^s, \lambda_1^s, \lambda_2^s\right)$, we compute the associated effective stiffness $\bar{\mathbb{C}}^s$ with the help of an FFT-based computational micromechanics code [7,8]. The generated data $\left\{\left(\bar{\mathbb{C}}^s, \mathbb{C}_1^s, \mathbb{C}_2^s, \lambda_1^s, \lambda_2^s\right)\right\}_{i=1}^{N_s}$ serves as training data for identifying the DMN. The number of samples depends on the discretization of the orientation triangle and is summarized in Table 3. For D4, we generate 800 samples in total which corresponds to 200 samples per volume element. To keep the sampling and training effort manageable, we reduce the number of generated samples to 100 and 50 per microstructure when increasing the number of discrete orientations to ten and 31, respectively. For D4, D10 and D31, we randomly split the pre-computed samples into a training and validation set, comprising 90% and 10% of the samples. We train the deep material network on mini-batches with a batch size of $N_b = 32$ samples. More precisely, we draw the batches randomly from the training set and drop the last batch, should the remaining batch size be smaller than 32.

We consider deep material networks with eight layers. In general, eight layers are necessary to achieve a sufficient approximation quality, in particular for inelastic computations [51–53]. We train for 3000 epochs and rely upon the AMSGrad method [79,80] combined with the warm restart technique suggested by Loshchilov–Hutter [81]. Furthermore, we make use of a modulation of the learning rate between a minimum learning rate $\alpha_{\min}$ and a maximum learning rate $\alpha_{\max}$, i.e.,

$$\alpha : \mathbb{N} \to \mathbb{R}, \quad m \mapsto \gamma^m \left(\alpha_{\min} + \frac{1}{2}\left(\alpha_{\max} - \alpha_{\min}\right)\left(1 + \cos\left(\pi \frac{m}{M}\right)\right)\right), \tag{4.11}$$

where $2M$ corresponds to the period and $M = 50$ is chosen. Additionally, we decay the learning rate at a geometric rate with $\gamma = 0.999$.

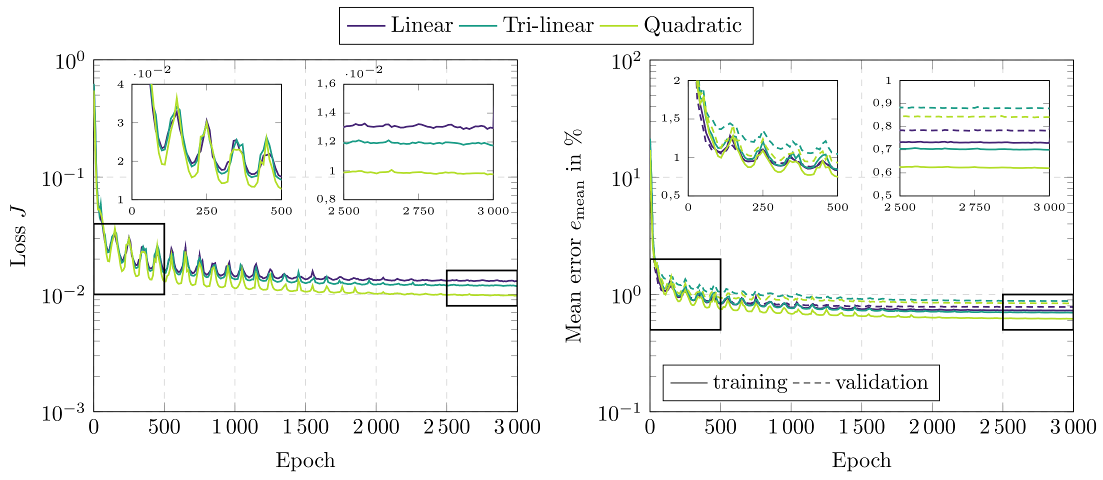
Since gradient descent is sensitive w.r.t. the proper choice of the step size, we determine the learning rates α_p, α_q and α_v by a learning rate sweep as introduced by Smith–Topin [82]. The resulting learning rates are almost identical for all three parameter groups, and, therefore, we set $\alpha_\mathrm{p} = \alpha_\mathrm{q} = \alpha_\mathrm{v} = \alpha$ with a maximum learning rate of $\alpha_{\max} = 1.5 \cdot 10^{-2}$. The minimum learning rate is chosen to be an order of magnitude smaller than the maximum learning rate, i.e., $\alpha_{\min} = 1.5 \cdot 10^{-3}$. We sample the initial weights $\vec{v}$ from a uniform distribution on $[0, 1]$ and rescale the weights to sum to unity. The entries of the parameter vectors $\vec{p}$ and $\vec{q}$ are sampled from a uniform distribution on $[0, 2\pi]$.

The penalty parameter of the objective function (3.9) is set to $\lambda = 10^3$. Additionally, we set the exponents to $p = 1$ and $q = 10$, see Gajek et al. [53], i.e., we enforce the maximum of the component-wise mean error to be minimized. To assess the accuracy of the fit, we define the sample-wise error

$$e_s = \frac{\left\| \mathcal{DMN}_{\mathcal{A}}^{\mathcal{L}}\left(\mathbb{C}_1, \mathbb{C}_2, \lambda_1, \lambda_2\right) - \bar{\mathbb{C}}^s \right\|_1}{\left\| \bar{\mathbb{C}}^s \right\|_1}, \tag{4.12}$$

where $\|\cdot\|_1$ refers to the Frobenius-1 norm defined by the ℓ^1-norm of the stiffness components in Voigt notation. Additionally, we define the maximum and mean errors of all samples

$$e_{\max} = \max_s \left(e_s\right) \quad \text{and} \quad e_{\text{mean}} = \frac{1}{N_s} \sum_{s=1}^{N_s} e_s, \tag{4.13}$$



(a) Loss function vs. training epoch

(b) Mean error vs. training epoch

Fig. 8. Loss function (a) and mean training and validation error (b) during training for D31.

where N_s denotes the number of elements in the training or validation set, depending on the considered scenario.

In Fig. 8, the training progress for the D31 orientation discretization and the investigated linear, tri-linear and quadratic orientation interpolations is shown. In the first 500 epochs, the effect of the learning rate modulation becomes apparent. The loss function and mean error fluctuate noticeably. In the last 500 epochs, the decay of the learning rate ensures convergence of the trained parameters. Conforming to intuition, increasing the degrees of freedom, i.e., choosing a tri-linear or quadratic orientation interpolation over a linear orientation interpolation, decreases the loss function at convergence. This trend carries over to the mean training errors, as well. For the mean validation error, however, the linear orientation interpolation provides the best mean validation error. Such overfitting phenomena are not uncommon for training deep neural networks, where increasing the degrees of freedom not necessarily yields better generalization and validation results. As training progresses, no increasing validation errors can be observed for linear, tri-linear and quadratic orientation interpolation. Thus, no significant model over-fitting is observed during training.

In Table 4, we summarized the training results for the investigated orientation discretization and interpolation. Additionally, we listed the number of non-zero weights w^i_{K+1} at the end of training. In general, we observe the following trends: Mean and maximum training errors will decrease if a tri-linear or quadratic orientation interpolation is chosen instead of a linear interpolation. The same observation holds for the mean validation error. The maximum validation error, on the other hand, does not necessarily decrease by introducing additional fitting parameters. Furthermore, the maximum validation error shows significant fluctuations. Indeed, for D4 and the linear interpolation, it reaches almost 12%. Since the maximum validation error is dominated by a single sample, see Liu et al. [51,52] or Gajek et al. [53], we consider the mean validation error to be a more appropriate indicator of the quality of the training results.

If we go from D4 to D10 and D31, the loss function as well as the training and validation errors will increase, in general. Indeed, the DMN has to predict the effective behavior of significantly more volume elements with different fiber orientations and this result does not come unexpected.

4.6. Online evaluation

We implemented Newton's method, as described in Section 3.2, as a user-material subroutine in ABAQUS. In terms of implementation, the major difference compared to Gajek et al. [53] is that, for the work at hand, the gradient operator depends on the fiber orientation parameters λ_1 and λ_2, see Section 3.2. This does not infer any additional challenges, since the microstructure characteristics do not change during computation. Both parameters λ_1 and λ_2

Table 4
Training results of the deep material network for different orientation interpolations and orientation discretizations.

		J	$e^{\text{tr}}_{\text{mean}}$	$e^{\text{tr}}_{\text{max}}$	$e^{\text{val}}_{\text{mean}}$	$e^{\text{val}}_{\text{max}}$	Active weights
D4	Linear	$7.378 \cdot 10^{-3}$	0.462%	1.236%	0.669%	7.088%	71%
	Tri-linear	$6.456 \cdot 10^{-3}$	0.413%	1.016%	0.572%	3.960%	73%
	Quadratic	$6.736 \cdot 10^{-3}$	0.431%	1.054%	0.563%	2.948%	71%
D10	Linear	$1.026 \cdot 10^{-2}$	0.638%	1.610%	0.982%	11.982%	70%
	Tri-linear	$9.109 \cdot 10^{-3}$	0.587%	1.266%	0.698%	4.933%	70%
	Quadratic	$8.156 \cdot 10^{-3}$	0.541%	1.127%	0.665%	2.969%	68%
D31	Linear	$1.296 \cdot 10^{-2}$	0.730%	2.173%	0.783%	3.935%	67%
	Tri-linear	$1.175 \cdot 10^{-2}$	0.700%	1.928%	0.879%	6.018%	65%
	Quadratic	$9.757 \cdot 10^{-3}$	0.620%	1.361%	0.841%	7.051%	68%

are fixed during the online evaluation, and after assembling $\boldsymbol{A}_{\lambda_1\lambda_2}$, the effective stress and algorithmic tangent are computed by Newton's method in the proposed manner. For a summary of the main steps of the algorithm, i.e., computing phase strains and stresses, updating the displacement jumps, updating internal variables and computing the DMN's effective stress, we refer to Gajek et al. [53], where the pseudo-code for the implementation of the online phase can be found.

Our goal in Section 5 is to employ deep material networks for a two-scale simulation using ABAQUS. For this purpose, we seek to speed up the inelastic computations as much as possible. As a first step, the elimination procedure described in Section 3 proves effective. The deleting is performed in an upstream pre-processing step after the offline training to avoid unnecessary computational overhead. Secondly, we exploit the sparsity pattern of all involved linear operators, both, the sparsity pattern of the gradient operator and the Jacobian $\partial\vec{\boldsymbol{\sigma}}/\partial\vec{\boldsymbol{\varepsilon}}$, containing the algorithmic tangents of the phases. To this end, we rely upon the library Eigen3 [83] for all linear algebra operations and use sparse matrices whenever possible. For Newton's method, we use the following convergence criterion

$$\frac{\left\| \boldsymbol{A}^{\mathsf{T}}_{\lambda_1\lambda_2} \boldsymbol{W} \vec{\boldsymbol{\sigma}}(\vec{\bar{\boldsymbol{\varepsilon}}}^{\,n+1} + \boldsymbol{A}_{\lambda_1\lambda_2} \vec{\boldsymbol{a}}^{\,n+1}, \vec{\boldsymbol{z}}^{\,n}) \right\|_{\text{F}}}{(2^K - 1) \left\| \bar{\boldsymbol{\sigma}}^{\,n+1} \right\|_{\text{F}}} \leq \text{tol}, \tag{4.14}$$

where we set the tolerance tol to 10^{-12} and $\|\cdot\|_{\text{F}}$ refers to the Frobenius norm defined by the ℓ^2-norm of the involved matrices in Voigt notation. We solve the linear system by means of a sparse Cholesky decomposition.

Before employing the DMN in a two-scale simulation, we turn our attention to validating the predicted effective stress of the deep material network. To investigate whether deep material networks are capable of accurately interpolating the effective stress outside of the training regime, additional volume elements were generated. We use D4 as our point of departure and subdivide the orientation triangle three more times yielding 105 additional sampling points on the orientation triangle. For D10 and D31, we subdivide the orientation triangle two and one more times, giving rise to 99 and 78 additional sampling points, respectively.

In Fig. 9, for all three discretizations, the sampling points used for obtaining the training data (hollow circles) and the sampling points exclusively used for the inelastic validations (black filled circles) are shown. For every additional sampling point, we generated a volume element using SAM [57]. We keep using the volume elements already generated for offline training such that we have 109 generated volume elements in total for every D4, D10 and D31.

Using the material parameters of the short fiber reinforced polyamide given in Section 4.1, we investigate six independent uniaxial strain loadings

$$\bar{\boldsymbol{\varepsilon}} = \frac{\bar{\varepsilon}}{2}\left(\boldsymbol{e}_i \otimes \boldsymbol{e}_j + \boldsymbol{e}_j \otimes \boldsymbol{e}_i\right), \quad (i,j) \in L_1 \equiv \{(1,1),(2,2),(3,3),(1,2),(1,3),(2,3)\}. \tag{4.15}$$

For every uniaxial strain loading direction in the index set L_1, a full hysteresis with a strain amplitude of $\bar{\varepsilon} = 2.5\%$ over a time of $T = 4$ s is computed in 80 equidistant load steps to account for load reversal. To consider more complex loading conditions, we investigate six independent biaxial strain loadings

$$\bar{\boldsymbol{\varepsilon}} = \bar{\varepsilon}_1\, \boldsymbol{e}_i \otimes \boldsymbol{e}_i + \bar{\varepsilon}_2\, \boldsymbol{e}_j \otimes \boldsymbol{e}_j, \quad (i,j) \in L_2 \equiv \{(1,2),(1,3),(2,1),(2,3),(3,1),(3,2)\} \tag{4.16}$$

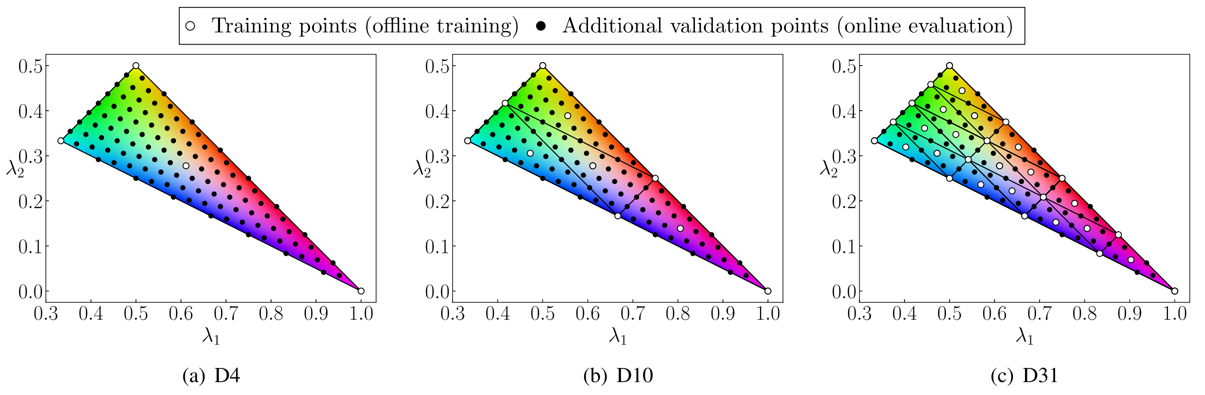


Fig. 9. Investigated fiber orientation discretizations comprising volume elements used in offline training and additionally generated volume elements for the online validation: (a) four training and 105 validation, (b) ten training and 99 validation and (c) 31 training and 78 validation points.

over a time of $T = 2$ s and computed in 40 equidistant load steps. For every biaxial loading direction in the index set L_2, a strain loading of $\bar{\varepsilon}_1 = 1.77\%$ is applied while the strain in the second direction is held constant $\bar{\varepsilon}_2 = 0\%$. After the load in the first direction is applied, a strain loading of $\bar{\varepsilon}_2 = 1.77\%$ is applied in the second direction as well. Both for the uniaxial and biaxial simulations, mixed boundary conditions [84], i.e., stress free loading perpendicular to the loading direction, were used. As reference, we compute the volume elements' effective stress $\bar{\boldsymbol{\sigma}}^{\text{FFT}}$ by means of an FFT-based computational micromechanics code and use an Eyre–Milton solver [85,86]. To evaluate the approximation error in a quantitative way, we introduce the following error measures. For fixed orientation parameters λ_1 and λ_2, we define the relative error in the stress component (i, j) as

$$\eta_{ij,\lambda_1\lambda_2}(t) = \frac{\left|\bar{\sigma}^{\text{DMN}}_{ij,\lambda_1\lambda_2}(t) - \bar{\sigma}^{\text{FFT}}_{ij,\lambda_1\lambda_2}(t)\right|}{\max\limits_{t\in\mathcal{T}} \left|\bar{\sigma}^{\text{FFT}}_{ij,\lambda_1\lambda_2}(t)\right|}, \tag{4.17}$$

where $\mathcal{T} = [0, T]$ denotes the considered time interval. Furthermore, the mean and the maximum error are defined by

$$\eta^{\text{mean}}_{\lambda_1\lambda_2} = \max_{i,j\in\{1,2,3\}} \frac{1}{T}\int_0^T \eta_{ij,\lambda_1\lambda_2}(t)\mathrm{d}t \tag{4.18}$$

and

$$\eta^{\text{max}}_{\lambda_1\lambda_2} = \max_{i,j\in\{1,2,3\}} \max_{t\in\mathcal{T}} \eta_{ij,\lambda_1\lambda_2}(t), \tag{4.19}$$

respectively. For each of the sampled 109 fiber orientation states, we compute the previously mentioned twelve independent loading paths and evaluate these error measures. The resulting mean and maximum errors for all orientations are shown in Fig. 10, for D31 and linear orientation interpolation. Comparing Figs. 10(a) and 10(b), we observe that the mean error fluctuates less on the orientation triangle than the maximum error, in particular in the vicinity of the isotropic fiber orientation. The maximum relative error attains its maximum value of around 5.5% for the unidirectional fiber orientation. The mean error, on the other hand, fluctuates less and attains its maximum also for the unidirectional case. Still, with a maximum error of 5.5%, the DMN is capable of predicting the effective stress of all investigated 109 discrete fiber orientation states with sufficient accuracy.

Fig. 11 gives an impression of how the mean and maximum errors shown in Fig. 10 translate into actual stress–time curves. For the three extreme cases and an uniaxial extension in the 11-direction, the 11-component of the effective stress predicted by the DMN and computed by an FFT-solver is shown in Fig. 11(a). For isotropic and planar isotropic fiber orientations, the DMN's effective stress $\bar{\sigma}^{\text{DMN}}_{11}$ and the full field solution $\bar{\sigma}^{\text{FFT}}_{11}$ are almost indistinguishable. Even for the unidirectional fiber orientation, which exhibits the previously mentioned maximum error of 5.5%, $\bar{\sigma}^{\text{DMN}}_{11}$ and $\bar{\sigma}^{\text{FFT}}_{11}$ show a good agreement. In Fig. 11(b), the stress components for a biaxial extension in the 11- and the 22-direction are illustrated. As noted previously, we observe the largest relative error for the

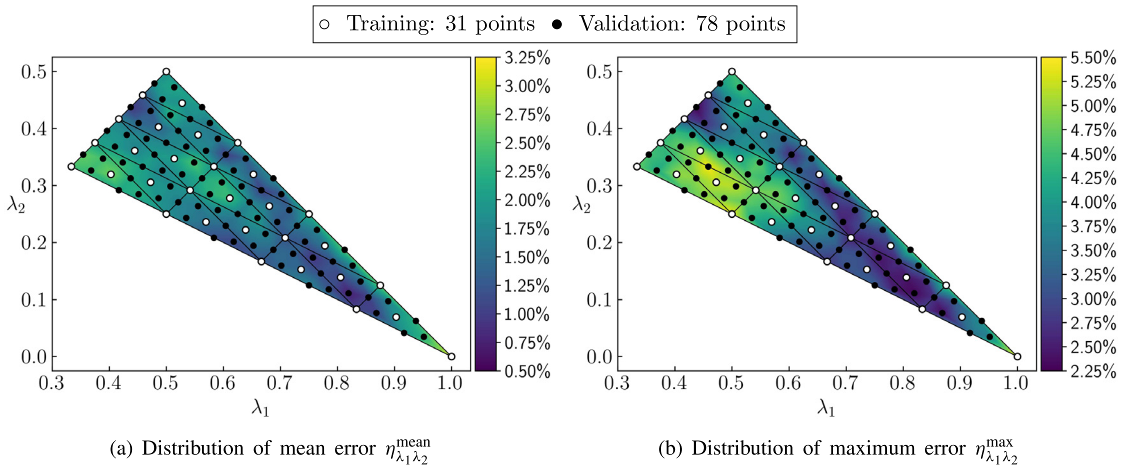


Fig. 10. Distribution of mean and maximum error on the orientation triangle for D31 and a linear orientation interpolation.

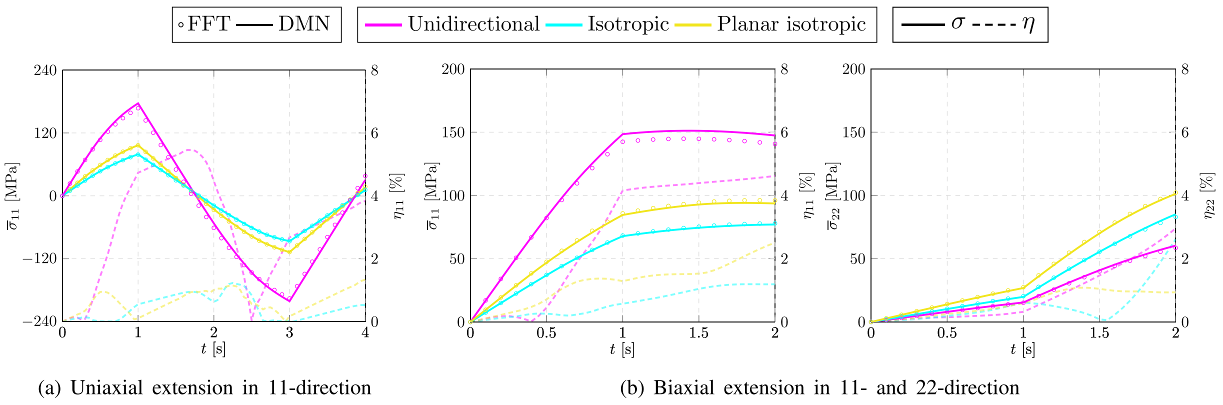


Fig. 11. Comparing the effective stresses predicted by the DMN and computed by an FFT-solver for an uniaxial extensions (a) and biaxial extension (b) for the three extreme fiber orientation states.

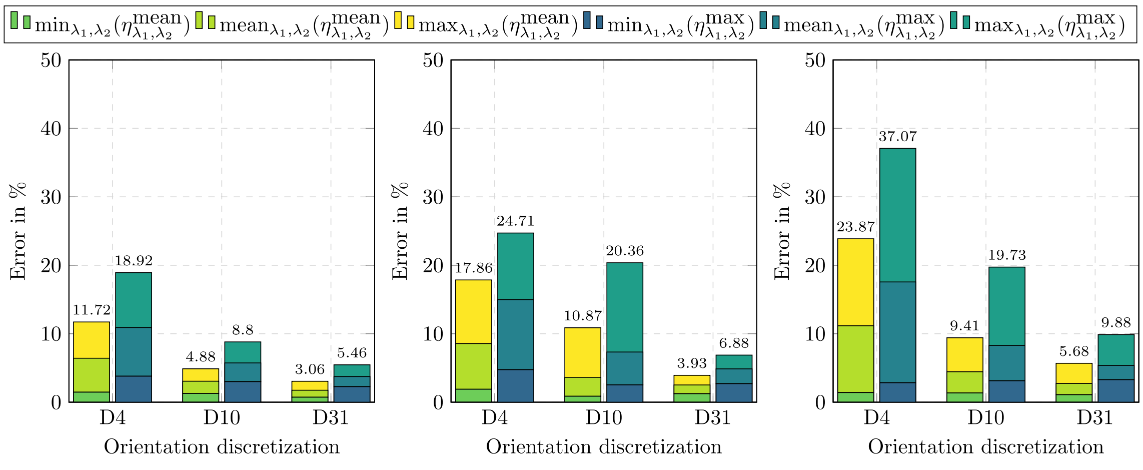
unidirectional fiber orientation, whereas the predictions of DMN and FFT for isotropic and planar isotropic fiber orientations are almost indistinguishable.

Fig. 12 summarizes the minimum, mean and maximum of the individual error measures with respect to the orientation parameters λ_1 and λ_2 for D4, D10, D31 and linear, tri-linear and quadratic orientation interpolation.

Fig. 12, which compares 11 772 computed DMN load paths with 1308 full field simulations, permits us to draw the following conclusions. For fixed type of orientation interpolation, a finer orientation discretization reduces the mean and maximum errors. The dependence of the mean and maximum error on the polynomial degree is more complicated. Compared to the linear orientation interpolation, both, the mean and the maximum error increase significantly for tri-linear and quadratic interpolation. The former holds true for D4, D10 and D31. Similar to the offline training, the loss and training errors decrease for higher order interpolations. Only the validation errors shows a slight increase. This suggests an overfitting with respect to the orientation interpolation, so that the linear approaches to the orientation interpolation is recommended, in the end.

5. A computational example

After validating that the DMN is able to provide sufficiently accurate results, we turn our attention to a component of industrial complexity. We choose a quadcopter frame, see Fig. 1(b), whose CAD geometry is publicly available [1]. We assume that the arms of the quadcopter are manufactured by injection molding. To obtain realistic



(a) Linear interpolation (b) Tri-linear interpolation (c) Quadratic interpolation

Fig. 12. Minimum, mean and maximum errors w.r.t. mean $\eta^{\text{mean}}_{\lambda_1\lambda_2}$ and maximum $\eta^{\max}_{\lambda_1\lambda_2}$ error for the three considered orientation interpolations.

Table 5
Parameters used for the injection molding simulation [89].

Mass density:	1410 kg/m^3	Folger–Tucker interaction coeff. :	$C_\text{i} = 0.01$
Injection temperature:	548.15 K	Particle number:	$N_\text{p} = 0$
Mold temperature:	313.15 K	Glass transition temperature:	$\theta_\text{ref} = 503.15$ K
Heat cap. at const. pressure:	2400 J/(K kg)	A_0:	0.1 s
Thermal conductivity:	0.25 W/(m K)	A_1:	0.65
Initial orient. distribution:	Isotropic	A_2:	0.021 1/K
Fiber aspect ratio:	$r_a = 20$	Reference shear viscosity η_0:	100 Pas

fiber orientation data, we conducted a mold filling simulation for a single quadcopter arm (and utilize these data for all four identical quadcopter arms). We use the publicly available software InjectionMoldingFoam [87], which is based upon the two-phase, incompressible flow solver of OpenFOAM [88]. We choose identical settings and material parameters as Köbler et al. [56], i.e., we assume a homogeneous fiber volume fraction and select the following Carreau–WLF equation [60]

$$\eta(\theta, \dot{\gamma}) = \eta_0 \frac{e^{-A_2(\theta-\theta_\text{ref})}}{\left(1 + (A_0\dot{\gamma})^2\right)^{\frac{1-A_1}{2}}}. \tag{5.1}$$

Here, θ denotes the absolute temperature and $\dot{\gamma}$ refers to the norm of the strain-rate tensor. The parameters for the injection molding simulation are summarized in Table 5 and originally stem from Bhat et al. [89]. The results of the injection molding simulation are shown in Fig. 13, both for a top and a side view, and at three distinct instances of time, corresponding to a volume coverage of 50%, 75% and 100%, respectively. The computed fiber orientations are represented by the color scale shown in Fig. 3. We observe that, at the flow fronts, planar and isotropic fiber orientations dominate. For the rest of the drone arm, fiber orientations close to the unidirectional case are prevalent. Fig. 13(d) illustrates the principal fiber orientations, i.e., the eigenvector corresponding to the largest eigenvalue of $\boldsymbol{A}_2$, after the mold is filled.

As a result of the location of the injection points, pronounced weld lines are formed on the left of the center of the drone arm, see Fig. 13(c).

Using the material behavior of the short fiber reinforced polyamide as given in Section 4.1, we perform a structural simulation on a single drone arm using the FE software ABAQUS. We mesh the drone arm by quadratic tetrahedron elements and investigate five different mesh densities ranging from 63 580 up to 1 005 862 elements in order to analyze convergence, wall–clock times and memory requirements, see Table 9. The computed fiber orientation tensors serve as the input for the simulation, i.e., the eigenvectors of $\boldsymbol{A}_2$ are mapped onto the ABAQUS

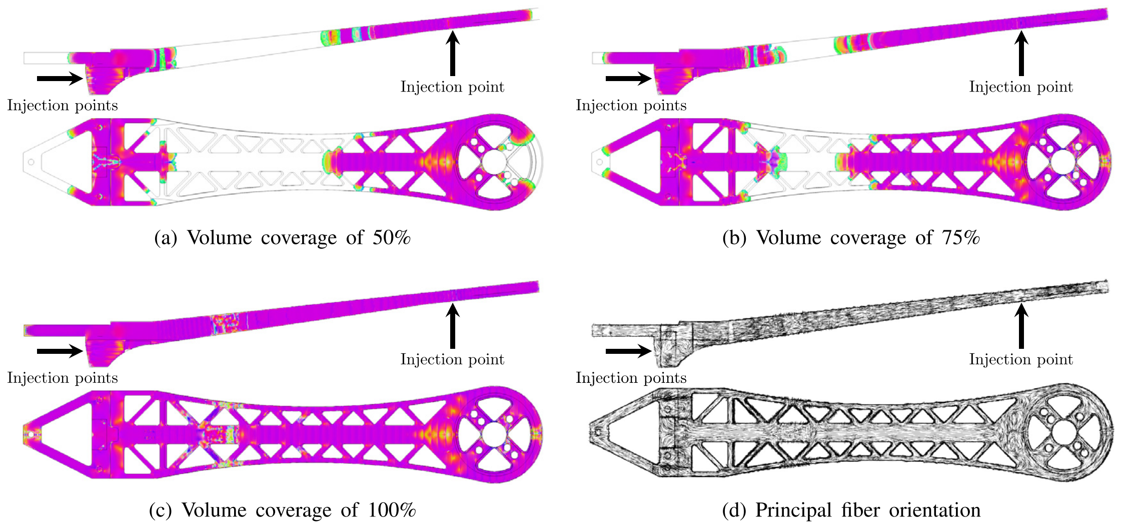

Injection points
Injection point
(a) Volume coverage of 50%
(b) Volume coverage of 75%
(c) Volume coverage of 100%
(d) Principal fiber orientation

Fig. 13. Injection molding simulation with volume coverage of 50%, 75% and 100% and principal fiber orientation after filling.

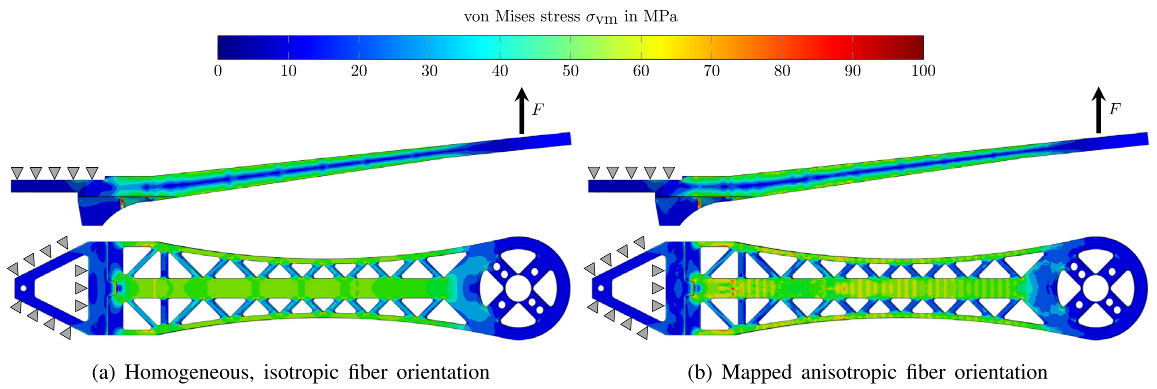

von Mises stress σ_{vm} in MPa
0
10
20
30
40
50
60
70
80
90
100
F
(a) Homogeneous, isotropic fiber orientation
(b) Mapped anisotropic fiber orientation

Fig. 14. Side and top view of the simulated drone arm for a homogeneous, isotropic fiber orientation (a) and the mapped anisotropic fiber orientation (b) from the injection molding simulation.

mesh and determine the material orientation. The eigenvalues λ_1 and λ_2 are provided to the DMN subroutine via pre-defined fields. We apply a loading of $F = 80$ N on the motor mount via a surface force and fix the left side of the drone arm, see Fig. 14.

The loading is applied in ten equidistantly spaced time increments. Fig. 14 shows the results for the finest discretization of about one million tetrahedron elements. The former took about 165 min to complete on 96 threads and required 133 GB of DRAM. On the left hand side, the von Mises stress distribution for the last time increment and for an assumed homogeneous and isotropic fiber orientation is shown. The right hand side of Fig. 14 shows the computed stress for the mapped anisotropic, inhomogeneous fiber orientation. For the mapped anisotropic fiber orientation, stress fluctuations, especially in the vicinity of weld lines are clearly visible. In contrast, stress and strain concentrations at weld lines cannot be predicted for a homogeneous fiber orientation. Accounting for the entire process chain appears imperative in order to exploit the full lightweight potential of injection-molded fiber-reinforced components, as becomes evident when comparing the predicted total deflections. Indeed, for the assumed isotropic fiber orientation, the macro simulation predicts a deflection of 6.13 mm. A deflection of 3.99 mm is predicted for the anisotropic fiber distribution. Thus, the isotropic variant underestimates the actual stiffness of the component by a factor of two.

To demonstrate the capabilities of the introduced multiscale method, we investigate the entire drone frame in a mechanical simulation, see Fig. 15. The four drone arms are manufactured from injection molded, short fiber

Table 6
Material parameters of the aluminum plates [90].

Aluminum	$E = 75$ GPa	$\nu = 0.3$	$\sigma_0 = 75$ MPa	$k = 416$ MPa	$m = 0.3895$

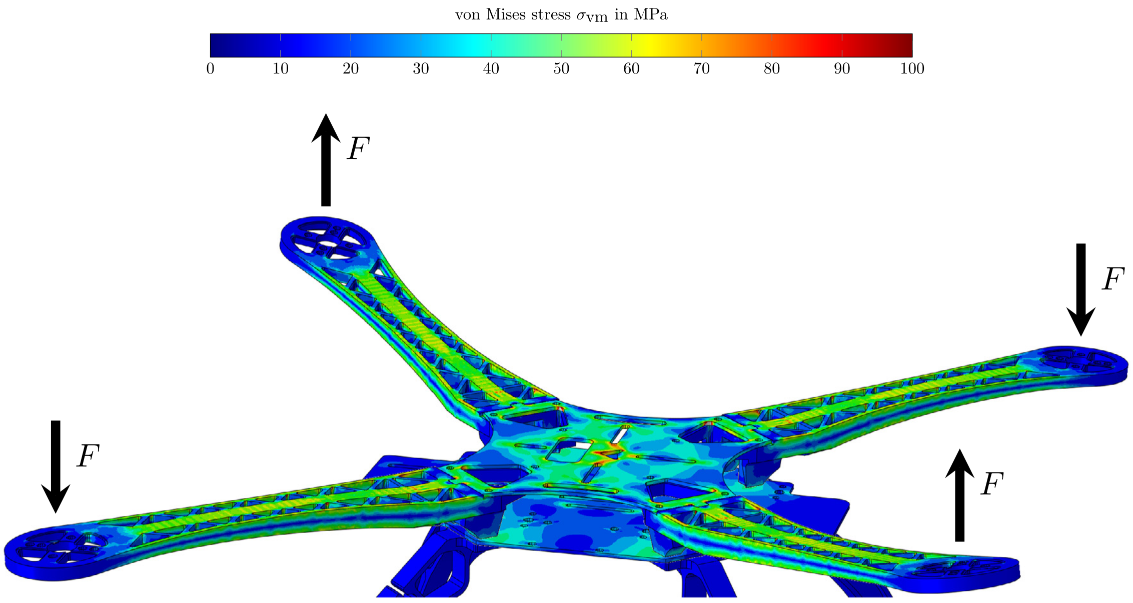


Fig. 15. Simulated quadcopter drone frame.

reinforced polyamide with mapped anisotropic fiber orientation. A deep material network is integrated at every Gauss point. Both, the upper and lower plates, which the drone arms are attached to, are made of aluminium. For this material, we use a J_2-elastoplasticity model [69] with power law hardening

$$\sigma_y = \sigma_0 + k\,\varepsilon_p^m. \tag{5.2}$$

The material parameters, i.e., Young's modulus E, Poisson's ratio ν and hardening parameters σ_0, k and m are taken from Segurado et al. [90] and summarized in Table 6. We assume the drone legs to be made of pure polyamide. For the latter, we assign a linear elastic material behavior, see Table 2. The simulation model consists of about two million elements with almost ten million degrees of freedom, see Table 10. The drone arms are loaded as shown in Fig. 15 with a force of $F = 80$ N. The loading is applied in ten equidistant load steps. The simulation took about 4.5 h wall–clock time on a consumer grade workstation, running on 96 threads in parallel. In particular, deep material networks enable high-fidelity two-scale simulations with reasonable effort. Indeed, by the results of Section 4.6, the maximum error of DMNs, evaluated for all investigated fiber orientations and load cases, does not exceed 5.5%. For most engineering applications, such an error appears reasonable.

6. Computational cost

Last but not least, we discuss the computational cost of deep material networks accounting for the offline training and online evaluation separately. The material sampling is performed in parallel, i.e., we compute six load steps in parallel using 16 threads for each individual simulation. All deep material networks are trained in parallel on 4 threads each. The wall–clock times of the material sampling and the offline training are summarized in Table 7. Apparently, the sampling and offline training effort increases linearly with the number of samples. Incidentally, the number of fitting parameters, which varies depending on the type of the orientation interpolation, has no significant influence on the runtime of the offline training.

Turning our attention to the online evaluation, we focus on the computational costs of the DMN evaluated at a single Gauss point. Integrating a deep material network at a single Gauss point for a prescribed macro strain

Table 7
Wall–clock times for sampling, training and number of fitting parameters.

		Wall–clock time			#Fitting parameters
		D4	D10	D31	
Sampling		21.48 h (800 samples)	27.02 h (1000 samples)	42.43 h (1550 samples)	
Training	Linear	2.43 h	3.18 h	4.60 h	1020
	Tri-linear	2.58 h	3.30 h	4.60 h	1275
	Quadratic	2.50 h	3.18 h	4.62 h	1785

Table 8
Wall–clock times and speed-up (compared to an FFT-based computational micromechanics solver) for a single time step of the inelastic micro simulation.

	FFT (1 thread)	DMN (1 thread)
Wall–clock time	242.69 s	2.03 ms
Speed-up	–	119 552
#DOF	6×256^3	513

Table 9
Wall–clock time, memory consumption and total Newton iterations of the single drone arm for different mesh sizes.

	ABAQUS (96 threads)				
Elements	63 580	121 416	247 444	488 689	1 005 862
#DOF	308 987	572 688	1 134 597	2 194 091	4 418 695
Wall–clock time	9 min	15 min	32 min	63 min	165 min
Memory consumption	9 GB	15 GB	33 GB	62 GB	133 GB
Total Newton iterations	18	18	19	20	20

increment takes about 2 ms on a single thread, see Table 8. This is about 120 000 times faster than a full-field simulation of the micro problem using an FFT-based computational mechanics solver on a microstructure discretized by 256^3 voxels, also running on a single thread. For the work at hand, we exclusively consider DMNs with eight layers. For applications which permit using DMNs with a smaller number of layers, even higher speed-up factors can be reached.

Next, we focus on the component scale simulation of the quadcopter arm. The wall–clock times of all five examined discretizations ranging from 63 580 up to 1 005 862 quadratic tetrahedron elements and, computed on 96 threads, are summarized in Table 9. We observe that the DRAM footprint is roughly proportional to the number of elements. The wall–clock times, however, increase super-linearly. We attribute this effect to the complexity of the direct solver used by ABAQUS. Apparently, the applicability of the method is more restricted by the memory requirements, and the computational effort plays a minor role. The required DRAM depends on the number of internal variables to be stored. For a DMN of eight layers, linear elastic fibers and an elastoplastic matrix, $128 \times (1 + 5) = 768$ floating-point numbers need to be stored for every Gauss point. To improve the convergence of Newton's method, the displacement jumps of the last converged time step are stored as well, i.e., $768 + 255 \times 3 = 1533$ scalars need to be kept in memory. Since we rely upon the thinned binary tree as introduced in Section 3.2, 1533 serves as an upper bound. For the application at hand, the actual number of internal variables of the DMN surrogate model is 1297. ABAQUS requires no more than two Newton iterations per load step, indicating a robust quadratic convergence independent of the mesh size. The slight increase from 1.8 to 2.0 Newton iterations per load step is minimal, and may be a result of the increased plastification of the material in the vicinity of finely resolved geometric features of the component.

To analyze the drone arm, using less than one million elements for the discretization would be sufficient. Rather, by choosing such a fine discretization, we demonstrate that deep material networks easily scale to component scale

Table 10
Wall–clock time, memory consumption and total Newton iterations for the simulation of the entire quadcopter frame.

Part	Materials	Discretization	
Arms	DMN	$4 \times 488\,689$	Quadratic tetrahedron elements
Bottom plate	Aluminum	39 422	Quadratic hexahedron elements
Top plate	Aluminum	19 028	Quadratic hexahedron elements
Legs	Polyamide	$4 \times 20\,054$	Quadratic tetrahedron elements
Total	–	2 093 422	–
#DOF			9 378 683
Wall–clock time			267 min
Memory consumption			252 GB
Total Newton iterations			22

simulations with higher complexity. The hardware requirements implicated by Table 9 can be provided by any state-of-the-art workstation.

Computing all ten load steps on 96 threads for the entire quadcopter took 22 Newton iterations, 267 min and required 252 GB or DRAM, see Table 10. This corresponds to over two million elements and about ten million degrees of freedom. In our opinion, this clearly shows that DMNs are a promising technique to enable two-scale simulations of industrial complexity with manageable resource expenditure.

7. Conclusion

In this work, we investigated the capabilities of deep material networks to provide a digital twin for short fiber reinforced plastic microstructures, which can be used in concurrent multiscale simulations. To realize the full lightweight potential of short fiber reinforced components, it is imperative to account for the locally varying fiber orientation in mechanical simulations on component scale. Building upon the work of Köbler et al. [56], we proposed a robust and computationally efficient approach to utilize direct deep material networks for variable fiber orientations. Instead of identifying multiple deep material networks and interpolating the effective stress, we interpolated the DMN's microstructure characteristics on the fiber orientation triangle. Assuming that the local fiber volume fractions of the individual laminates in the hierarchy are independent of the local fiber orientation, it suffices to fix the fiber volume fraction and to interpolate the lamination directions only. This procedure gives rise to a single DMN surrogate model covering all fiber orientations. Presumably, the scheme easily extends to incorporating local variations in the fiber volume fraction by interpolating the DMN's volume fractions as well. By sampling the training data from up to 31 microstructure realizations with different fiber orientation, we fitted the DMN to multiple fiber orientations simultaneously. Subsequently, we showed that the DMN generalizes to the entire fiber orientation triangle with small error, also for the inelastic regime.

To evaluate the ensuing performance of our approach, we simulated the entire process chain of a quadcopter frame starting from an injection molding simulation. We mapped the computed fiber orientations upon a finite element mesh of the complete quadcopter frame and conducted a two-scale simulation of the full component. Our results indicate that deep material networks enable two-scale simulations of structures with industrial complexity with moderate hardware requirements. In this way, the FE-DMN method finally realizes the promise of concurrent multiscale simulations and constitutes a powerful piece of technology which promises to become a standard tool for industrial applications.

It should be interesting to extend the FE-DMN method to problems involving damage or fracture [91–93], finite strains [94,95] or thermomechanical coupling [96,97] and to extend the range of applicability, for instance in the context of polycrystalline materials [98,99] or sheet molding compound composites [100,101] with continuous carbon fiber reinforcement. To be more specific, extending the proposed method to incorporate materials with micro-orientations, e.g., carbon fiber reinforced composites, is straightforward by introducing additional rotations at the bottom layer of the deep material networks and parameterizing these rotations in analogy to Section 2.3.

Despite the apparent success in practice, there is still a need for theoretical results which shed light on the approximation capabilities (and the limitations) of DMNs. Indeed, whether every fixed two-phase microstructure has a microstructure twin of a hierarchical laminate with identical effective properties, appears to be unresolved, see Problem 4 in Milton [102]. Interestingly, there exist counter examples for five-phase composites where the former is false, see Milton [62].

Declaration of competing interest

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

Acknowledgments

SG, MS and TB acknowledge financial support by the German Research Foundation (DFG) within the International Research Training Group "Integrated engineering of continuous–discontinuous long fiber reinforced polymer structures" (GRK 2078/2). The support by the German Research Foundation (DFG) is gratefully acknowledged. We thank the anonymous reviewers for their interest in this work, and the helpful comments.

References

[1] GRABCAD community, 2020, (accessed 27 October 2020) https://grabcad.com/library/s500-frame-1.
[2] J. Renard, M.F. Marmonier, Etude de l'initiation de l'endommagement dans la matrice d'un materiau composite par une methode d'homogenisation, Aerospace Sci. Technol. 9 (1987) 37–51.
[3] R.J.M. Smit, W.A.M. Brekelmans, H.E.H. Meijer, Prediction of the mechanical behavior of nonlinear heterogeneous systems by multi-level finite element modeling, Comput. Methods Appl. Mech. Eng. 155 (1) (1998) 181–192.
[4] F. Feyel, Multiscale FE2 elastoviscoplastic analysis of composite structures, Comput. Mater. Sci. 16 (1) (1999) 344–354.
[5] F. Feyel, J.-L. Chaboche, FE2 multiscale approach for modelling the elastoviscoplastic behaviour of long fibre SiC/Ti composite materials, Comput. Methods Appl. Mech. Eng. 183 (3-4) (2000) 309–330.
[6] F. Feyel, A multilevel finite element method FE2 to describe the response of highly non-linear structures using generalized continua, Comput. Methods Appl. Mech. Eng. 192 (28-30) (2003) 3233–3244.
[7] H. Moulinec, P. Suquet, A fast numerical method for computing the linear and nonlinear mechanical properties of composites, C. R. Acad. Sci. II 318 (11) (1994) 1417–1423.
[8] H. Moulinec, P. Suquet, A numerical method for computing the overall response of nonlinear composites with complex microstructure, Comput. Methods Appl. Mech. Engrg. 157 (1998) 69–94.
[9] M. Schneider, A review of nonlinear FFT-based computational homogenization methods, Acta Mech. online (2021) 1–50.
[10] J. Spahn, H. Andrä, M. Kabel, R. Müller, A multiscale approach for modeling progressive damage of composite materials using fast fourier transforms, Comput. Methods Appl. Mech. Eng. 268 (2014) 871–883.
[11] J. Kochmann, S. Wulfinghoff, S. Reese, J.R. Mianroodi, B. Svendsen, Two-scale FE–FFT- and phase-field-based computational modeling of bulk microstructural evolution and macroscopic material behavior, Comput. Methods Appl. Mech. Eng. 305 (2016) 89 – 110.
[12] J. Kochmann, L. Ehle, S. Wulfinghoff, J. Mayer, B. Svendsen, S. Reese, Efficient multiscale FE-FFT-based modeling and simulation of macroscopic deformation processes with non-linear heterogeneous microstructures, in: Multiscale Modeling of Heterogeneous Structures, Springer International Publishing, 2018, pp. 129–146.
[13] T. Mori, K. Tanaka, Average stress in matrix and average elastic energy of materials with misfitting inclusions, Acta Metall. 21 (5) (1973) 571–574.
[14] R. Hill, A self-consistent mechanics of composite materials, J. Mech. Phys. Solids 13 (4) (1965) 213–222.
[15] G. Dvorak, Y. Benveniste, On transformation strains and uniform fields in multiphase elastic media, Proc. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci. 437 (1992) 291–310.
[16] G. Dvorak, Y. Bahei-El-Din, A. Wafa, Implementation of the transformation field analysis, Comput. Mech. 14 (14) (1994) 201–228.
[17] G. Dvorak, Y. Bahei-El-Din, A. Wafa, The modeling of inelastic composite materials with the transformation field analysis, Model. Simul. Mater. Sci. Eng. 2 (2) (1994) 571–586.
[18] J.-L. Chaboche, P. Kanouté, A. Roos, On the capabilities of mean-field approaches for the description of plasticity in metal matrix composites, Int. J. Plast. 21 (7) (2005) 1409–1434.
[19] Z. Liu, M.A. Bessa, W.K. Liu, Self-consistent clustering analysis: An efficient multi-scale scheme for inelastic heterogeneous materials, Comput. Methods Appl. Mech. Engrg. 306 (2016) 319–341.
[20] Z. Liu, O.L. Kafka, C. Yu, W.K. Liu, Data-driven self-consistent clustering analysis of heterogeneous materials with crystal plasticity, in: Advances in Computational Plasticity, Springer, 2018, pp. 221–242.
[21] Z. Liu, M. Fleming, W.K. Liu, Microstructural material database for self-consistent clustering analysis of elastoplastic strain softening materials, Comput. Methods Appl. Mech. Engrg. 330 (2018) 547–577.
[22] Z. Hashin, S. Shtrikman, Note on a variational approach to the theory of composite elastic materials, J. Franklin Inst. B 271 (4) (1961) 336–341.
[23] Z. Hashin, S. Shtrikman, A variational approach to the theory of the elastic behaviour of polycrystals, J. Mech. Phys. Solids 10 (4) (1962) 343–352.

[24] S. Wulfinghoff, F. Cavaliere, S. Reese, Model order reduction of nonlinear homogenization problems using a Hashin–Shtrikman type finite element method, Comput. Methods Appl. Mech. Engrg. 330 (2018) 149–179.

[25] J.-L. Chaboche, P. Kanouté, A. Roos, On the capabilities of mean-field approaches for the description of plasticity in metal matrix composites, Int. J. Plast. 21 (2005) 1409–1434.

[26] M. Schneider, On the mathematical foundations of the self-consistent clustering analysis for non-linear materials at small strains, Comput. Methods Appl. Mech. Engrg. 354 (2019) 783–801.

[27] P. Ponte Castañeda, P. Suquet, Nonlinear Composites, in: Advances in Applied Mechanics, vol. 34, Elsevier, 1997, pp. 171–302.

[28] J.C. Michel, P. Suquet, Nonuniform transformation field analysis, Int. J. Solids Struct. 40 (2003) 6937–6955.

[29] F. Fritzen, T. Böhlke, Reduced basis homogenization of viscoelastic composites, Compos. Sci. Technol. 76 (2013) 84–91.

[30] R. Largenton, J.-C. Michel, P. Suquet, Extension of the nonuniform transformation field analysis to linear viscoelastic composites in the presence of aging and swelling, Mech. Mater. 73 (2014) 76–100.

[31] J.-C. Michel, P. Suquet, A model-reduction approach in micromechanics of materials preserving the variational structure of constitutive relations, J. Mech. Phys. Solids 90 (2016) 254–285.

[32] J.-C. Michel, P. Suquet, A model-reduction approach to the micromechanical analysis of polycristalline materials, Comput. Mech. 57 (3) (2016) 483–508.

[33] J.-C. Michel, P. Suquet, Effective potentials in nonlinear polycrystals and quadrature formulae, Proc. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci. 473 (2017) 20170213.

[34] F. Fritzen, M. Leuschner, Reduced basis hybrid computational homogenization based on a mixed incremental formulation, Comput. Methods Appl. Mech. Engrg. 260 (2013) 143–154.

[35] F. Fritzen, M. Hodapp, M. Leuschner, GPU accelerated computational homogenization based on a variational approach in a reduced basis framework, Comput. Methods Appl. Mech. Engrg. 278 (2014) 186–217.

[36] J. Köbler, N. Magino, H. Andrä, F. Welschinger, R. Müller, M. Schneider, A computational multi-scale model for the stiffness degradation of short-fiber reinforced plastics subjected to fatigue loading, Comput. Methods Appl. Mech. Engrg. 373 (2021) 113522.

[37] F. Fritzen, M. Hodapp, The finite element square reduced (FE2R) method with GPU acceleration: towards three-dimensional two-scale simulations, Internat. J. Numer. Methods Engrg. 107 (10) (2016) 853–881.

[38] J. Yvonnet, D. Gonzalez, Q.-C. He, Numerically explicit potentials for the homogenization of nonlinear elastic heterogeneous materials, Comput. Methods Appl. Mech. Eng. 198 (2009) 2723–2737.

[39] J. Yvonnet, E. Monteiro, Q.-C. He, Computational homogenization method and reduced database model for hyperelastic heterogeneous structures, Int. J. Multiscale Comput. Eng. 11 (3) (2013) 201–225.

[40] B.A. Le, J. Yvonnet, Q.-C. He, Computational homogenization of nonlinear elastic materials using neural networks, Internat. J. Numer. Methods Engrg. 104 (12) (2015) 1061–1084.

[41] M.N. Jadid, Prediction of stress-strain relationships for reinforced concrete sections by implementing neural network techniques, J. King Saud Univ., Eng. Sci. 9 (2) (1997) 169–188.

[42] D. Penumadu, R. Zhao, Triaxial compression behavior of sand and gravel using artificial neural networks (ANN), Comput. Geotech. 24 (3) (1999) 207 – 230.

[43] G. Srinivasu, R.N. Rao, T.K. Nandy, A. Bhattacharjee, Artificial neural network approach for prediction of titanium alloy stress-strain curve, Procedia Eng. 38 (2012).

[44] M. Mozaffar, R. Bostanabad, W. Chen, K. Ehmann, J. Cao, M.A. Bessa, Deep learning predicts path-dependent plasticity, Proc. Natl. Acad. Sci. 116 (52) (2019) 26414–26420.

[45] A. Koeppe, F. Bamer, B. Markert, An efficient Monte Carlo strategy for elasto-plastic structures based on recurrent neural networks, Acta Mech. 230 (2019) 3279–3293.

[46] M.B. Gorji, M. Mozaffar, J.N. Heidenreich, J. Cao, D. Mohr, On the potential of recurrent neural networks for modeling path dependent plasticity, J. Mech. Phys. Solids 143 (2020) 103972.

[47] F. Ghavamian, A. Simone, Accelerating multiscale finite element simulations of history-dependent materials using a recurrent neural network, Comput. Methods Appl. Mech. Eng. 357 (2019) 112594.

[48] K. Xu, D.Z. Huang, E. Darve, Learning constitutive relations using symmetric positive definite neural networks, J. Comput. Phys. 428 (2021) 110072.

[49] L. Wu, V.D. Nguyen, N.G. Kilingar, L. Noels, A recurrent neural network-accelerated multi-scale model for elasto-plastic heterogeneous materials subjected to random cyclic and non-proportional loading paths, Comput. Methods Appl. Mech. Eng. 369 (2020) 113234.

[50] F. Fritzen, M. Fernández, F. Larsson, On-the-fly adaptivity for nonlinear twoscale simulations using artificial neural networks and reduced order modeling, Front. Mater. 6 (2019) 75.

[51] Z. 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.

[52] Z. 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.

[53] S. Gajek, M. Schneider, T. Böhlke, On the micromechanics of deep material networks, J. Mech. Phys. Solids 142 (2020) 103984.

[54] Z. Liu, Deep material network with cohesive layers: Multi-stage training and interfacial failure analysis, Comput. Methods Appl. Mech. Eng. 363 (2020) 112913.

[55] Z. Liu, H. Wei, T. Huang, C.T. Wu, Intelligent multiscale simulation based on process-guided composite database, 2020, arXiv preprint arXiv:2003.09491.

[56] J. Köbler, M. Schneider, F. Ospald, H. Andrä, R. Müller, Fiber orientation interpolation for the multiscale analysis of short fiber reinforced composite parts, Comput. Mech. 61 (6) (2018) 729–750.

[57] M. Schneider, The sequential addition and migration method to generate representative volume elements for the homogenization of short fiber reinforced plastics, Comput. Mech. 59 (2017) 247–263.
[58] N. Halphen, Q. Nguyen, Sur les matériaux standards generalisés, J. Méc. 14 (1975) 508–520.
[59] S.G. Advani, C.L. Tucker, The use of tensors to describe and predict fiber orientation in short fiber composites, J. Rheol. 31 (8) (1987) 751–784.
[60] P.K. Kennedy, Flow Analysis of Injection Molds, second ed., Hanser, Munich, 2013.
[61] S. Montgomery-Smith, W. He, D.A. Jack, D.E. Smith, Exact tensor closures for the three-dimensional Jeffery's equation, J. Fluid Mech. 680 (2011) 321–335.
[62] G.W. Milton, Modelling the properties of composites by laminates, in: Homogenization and Effective Moduli of Materials and Media, Springer New York, New York, 1986, pp. 150–174.
[63] S. Torquato, Random Heterogeneous Materials: Microstructure and Macroscopic Properties, in: Interdisciplinary Applied Mathematics, Springer, New York, 2005.
[64] J. Vince, Mathematics for Computer Graphics, in: Undergraduate Topics in Computer Science, Springer London, 2017.
[65] G.W. Milton, the Theory of Composites, Cambridge University Press, Cambridge, 2002.
[66] M. Kabel, D. Merkert, M. Schneider, Use of composite voxels in FFT-based homogenization, Comput. Methods Appl. Mech. Engrg. 294 (2015) 168–188.
[67] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer, Automatic differentiation in PyTorch, in: NIPS Autodiff Workshop, 2017.
[68] N. Lahellec, P. Suquet, On the effective behavior of nonlinear inelastic composites: I. incremental variatonal principles, J. Mech. Phys. Solids 55 (2007) 1932–1963.
[69] J.C. Simo, T.J.R. Hughes, Computational Inelasticity, Springer, New York, 1998.
[70] I. Doghri, L. Brassart, L. Adam, J.-S. Gérard, A second-moment incremental formulation for the mean-field homogenization of elasto-plastic composites, Int. J. Plast. 27 (2011) 352–371.
[71] M. Schneider, On the barzilai-borwein basic scheme in FFT-based computational homogenization, Internat. J. Numer. Methods Engrg. 118 (8) (2019) 482–494.
[72] F. Willot, B. Abdallah, Y.-P. Pellegrini, Fourier-based schemes with modified green operator for computing the electrical response of heterogeneous media with accurate local fields, Internat. J. Numer. Methods Engrg. 98 (7) (2014) 518–533.
[73] F. Willot, Fourier-based schemes for computing the mechanical response of composites with accurate local fields, C. R. Méc. 343 (3) (2015) 232–245.
[74] J. Zeman, J. Vondřejc, J. Novak, I. Marek, Accelerating a FFT-based solver for numerical homogenization of periodic media by conjugate gradients, J. Comput. Phys. 229 (21) (2010) 8065–8071.
[75] S. Brisard, L. Dormieux, FFT-based methods for the mechanics of composites: A general variational framework, Comput. Mater. Sci. 49 (3) (2010) 663–671.
[76] I.M. Sobol, Distribution of points in a cube and approximate evaluation of integrals, U.S.S.R Comput. Maths. Math. Phys. 7 (1967) 86–112.
[77] M.D. McKay, R.J. Beckman, W.J. Conover, A comparison of three methods for selecting values of input variables in the analysis of output from a computer code, Technometrics 21 (2) (1979) 239–245.
[78] R.E. Miles, On random rotations in Rˆ3, Biometrika 52 (3/4) (1965) 636–639.
[79] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: 3rd International Conference on Learning Representations, ICLR 2015, 2015.
[80] S.J. Reddi, S. Kale, S. Kumar, On the convergence of Adam and beyond, in: International Conference on Learning Representations, 2018.
[81] I. Loshchilov, F. Hutter, SGDR: Stochastic gradient descent with warm restarts, in: International Conference on Learning Representations (ICLR) 2017 Conference Track, 2017.
[82] L.N. Smith, N. Topin, Super-convergence: very fast training of neural networks using large learning rates, in: T. Pham (Ed.), Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications, Vol. 11006, SPIE, International Society for Optics and Photonics, 2019, pp. 369–386.
[83] G. Guennebaud, B. Jacob, et al., Eigen v3, 2010, http://eigen.tuxfamily.org.
[84] M. Kabel, S. Fliegener, M. Schneider, Mixed boundary conditions for FFT-based homogenization at finite strains, Comput. Mech. 57 (2) (2016) 193–210.
[85] D.J. Eyre, G.W. Milton, A fast numerical scheme for computing the response of composites using grid refinement, Eur. Phys. J.-Appl. Phys. 6 (1) (1999) 41–47.
[86] M. Schneider, D. Wicht, T. Böhlke, On polarization-based schemes for the FFT-based computational homogenization of inelastic materials, Comput. Mech. 64 (4) (2019) 1073–1095.
[87] F. Ospald, Numerical simulation of injection molding using OpenFOAM, PAMM 14 (1) (2014) 673–674.
[88] H.G. Weller, G. Tabor, H. Jasak, C. Fureby, A tensorial approach to computational continuum mechanics using object-oriented techniques, Comput. Phys. 12 (6) (1998) 620–631.
[89] H.A. Bhat, S. Subramaniam, A. Pillai, L.E. Krishnan, A.E. M., Analysis and design of mold for plastic side release buckle using moldflow software, Int. J. Res. Eng. Technol. 03 (2014) 366–372.
[90] J. Segurado, J. Llorca, C. González, On the accuracy of mean-field approaches to simulate the plastic deformation of composites, Scr. Mater. 46 (7) (2002) 525–529.
[91] F. Ernesti, M. Schneider, T. Böhlke, Fast implicit solvers for phase field fracture problems on heterogeneous microstructures, Comput. Methods Appl. Mech. Engrg. 363 (2020) 112793.

[92] M. Schneider, An FFT-based method for computing weighted minimal surfaces in microstructures with applications to the computational homogenization of brittle fracture, Internat. J. Numer. Methods Engrg. 121 (7) (2020) 1367–1387.

[93] F. Ettemeyer, P. Lechner, T. Hofmann, H. Andrä, M. Schneider, D. Grund, W. Volk, D. Günther, Digital Sand Core Physics: Predicting physical properties of sand cores by simulations on digital microstructures, Int. J. Solids Struct. 188–189 (2020) 155–168.

[94] F. Ospald, N. Goldberg, M. Schneider, A fiber orientation-adapted integration scheme for computing the hyperelastic Tucker average for short fiber reinforced composites, Comput. Mech. 60 (4) (2017) 595–611.

[95] F. Ospald, M. Schneider, M. Kabel, A model order reduction method for computational homogenization at finite strains on regular grids using hyperelastic laminates to approximate interfaces, Comput. Methods Appl. Mech. Engrg. 309 (2016) 476–496.

[96] G. Chatzigeorgiou, N. Charalambakis, Y. Chemisky, F. Meraghni, Periodic homogenization for fully coupled thermomechanical modeling of dissipative generalized standard materials, Int. J. Plast. 81 (2016) 18–39.

[97] D. Wicht, M. Schneider, T. Böhlke, Computing the effective response of heterogeneous materials with thermomechanically coupled constituents by an implicit FFT-based approach, Internat. J. Numer. Methods Engrg. online (2020) 1–31.

[98] J. Kuhn, M. Schneider, P. Sonnweber-Ribic, T. Böhlke, Fast methods for computing centroidal laguerre tessellations for prescribed volume fractions with applications to microstructure generation of polycrystalline materials, Comput. Methods Appl. Mech. Engrg. 369 (2020) 113175.

[99] D. Wicht, M. Schneider, T. Böhlke, An efficient solution scheme for small-strain crystal-elasto-viscoplasticity in a dual framework, Comput. Methods Appl. Mech. Engrg. 358 (2020) 112611.

[100] J. Görthofer, N. Meyer, T.D. Pallicity, L. Schöttl, A. Trauth, M. Schemmann, M. Hohberg, P. Pinter, P. Elsner, F. Henning, A. Hrymak, T. Seelig, K. Weidenmann, L. Kärger, T. Böhlke, Virtual process chain of sheet molding compound: Development, validation and perspectives, Composites B 169 (2019) 133–147.

[101] J. Görthofer, M. Schneider, F. Ospald, A. Hrymak, T. Böhlke, Computational homogenization of sheet molding compound composites based on high fidelity representative volume elements, Comput. Mater. Sci. 174 (2020) 109456.

[102] G.W. Milton, Some open problems in the theory of composites, 2020, arXiv preprint arXiv:2008.03394.