Learning constitutive models from microstructural simulations via a non-intrusive reduced basis method: Extension to geometrical parameterizations

Theron Guo[a,c,*], Francesco A.B. Silva[a], Ondřej Rokoš[b,c], Karen Veroy[a,c]

[a] *Centre for Analysis, Scientific Computing and Applications, Department of Mathematics and Computer Science, Eindhoven University of Technology, 5612 AZ Eindhoven, The Netherlands*
[b] *Mechanics of Materials, Department of Mechanical Engineering, Eindhoven University of Technology, 5612 AZ Eindhoven, The Netherlands*
[c] *Institute for Complex Molecular Systems, Eindhoven University of Technology, 5612 AZ Eindhoven, The Netherlands*

Abstract

Two-scale simulations are often employed to analyze the effect of the microstructure on a component's macroscopic properties. Understanding these structure–property relations is essential in the optimal design of materials for specific applications. However, these two-scale simulations are typically computationally expensive and infeasible in multi-query contexts such as optimization and material design. To make such analyses amenable, the microscopic simulations can be replaced by inexpensive-to-evaluate surrogate models. Such surrogate models must be able to handle microstructure parameters in order to be used for material design. A previous work focused on the construction of an accurate surrogate model for microstructures under varying loading and material parameters by combining proper orthogonal decomposition and Gaussian process regression. However, that method works only for a fixed geometry, greatly limiting the design space. This work hence focuses on extending the methodology to treat geometrical parameters. To this end, a method that transforms different geometries onto a parent domain is presented, that then permits existing methodologies to be applied. We propose to solve an auxiliary problem based on linear elasticity to obtain the geometrical transformations. The method has a good reducibility and can therefore be quickly solved for many different geometries. Using these transformations, combined with the nonlinear microscopic problem, we derive a fast-to-evaluate surrogate model with the following key features: (1) the predictions of the effective quantities are independent of the auxiliary problem, (2) the predicted stress fields automatically fulfill the microscopic balance laws and are periodic, (3) the method is non-intrusive, (4) the stress field for all geometries can be recovered, and (5) the sensitivities are available and can be readily used for optimization and material design. The proposed methodology is tested on several composite microstructures, where rotations and large variations in the shape of inclusions are considered. Finally, a two-scale example is shown, where the surrogate model achieves a high accuracy and significant speed up, thus demonstrating its potential in two-scale shape optimization and material design problems.

---

* Corresponding author at: Centre for Analysis, Scientific Computing and Applications, Department of Mathematics and Computer Science, Eindhoven University of Technology, 5612 AZ Eindhoven, The Netherlands.
E-mail addresses: t.guo@tue.nl (T. Guo), f.a.b.silva@tue.nl (F.A.B. Silva), o.rokos@tue.nl (O. Rokoš), k.p.veroy@tue.nl (K. Veroy).

1. Introduction

With recent advances in additive manufacturing and metamaterials, tailoring the microstructure of materials to obtain desired engineering properties has become possible and crucial. In general, the structure–property relations of such microstructures are not well understood and need to be numerically investigated by fully resolving all microstructural details. Computational homogenization (CH) is typically employed, where the macro- and microstructure are modeled simultaneously. This results in a two-scale formulation which is either solved by a nested Finite Element (FE) scheme, also known as FE^2 (see, e.g. [1–5]), or by a combination of the FE method with a Fast Fourier Transform (FFT) solver, also known as FE-FFT (see, e.g. [6,7]).

Due to high computational costs involved in solving two-scale simulations, computational homogenization cannot be readily used for material design or optimization problems, where numerous forward simulations must be run. For this reason, several methods have been proposed in the literature that attempt to replace the parametric microscopic model with a fast-to-evaluate while accurate surrogate model. Most of these methods require training data collected by running full microscopic simulations for different parameters. They generally differ in the following aspects:

1. the amount of data needed for obtaining an accurate surrogate model,
2. the ability to make physically sound predictions, and
3. the way they handle parameters, which can be classified as loading parameters (the applied macroscopic strain) or microstructure descriptors (e.g., material properties of individual phases, volume ratios, etc.). If only loading parameters can be handled, the surrogate model has no design space and can only be used for forward computations.

One popular framework, termed data-driven computational mechanics, was introduced in Kirchdoerfer et al. [8] and extended in follow-up works; see, e.g., [9,10]. Given a collection of stress–strain pairs, this method uses a distance minimizing scheme to directly find the corresponding global stress and strain states of the macroscopic system that satisfy the balance laws as closely as possible, and hence bypasses the empirical material modeling step. This method typically requires very large datasets to be accurate and cannot handle any microstructure descriptors.

Another popular approach is to learn an effective constitutive model from the available data. Usually, neural networks are employed for this regression. In [11–13], the authors used recurrent neural networks, a special type of neural network, to learn a path-dependent elasto-plastic model for a composite RVE. Additionally, in Mozaffar et al. [13], microstructure descriptors were included to predict the stresses for a class of composite RVEs. Similarly, in Le et al. [14], the authors used neural networks to learn a constitutive model for a nonlinear elastic microstructure, considering also microstructural parameters. Recently, this methodology was also applied on mechanical metamaterials, where the authors considered topological microstructural parameters [15,16]. Despite the successes of this approach, it has two inherent drawbacks: requirements of large datasets for training and no possibility to guarantee that predictions fulfill physical laws.

To overcome these problems, several works recently proposed to embed the physics in neural networks by choosing special network architectures. In [17], by assuming a specific form of the strain energy density function, the authors incorporated theoretical knowledge from materials theory in order to predict physical results. A similar idea was implemented in [18], where a library of strain energy density functions, taken from literature, was used as the basis for the approximated constitutive model. Although these methods fulfill physics by construction and can treat some material parameters, they still require a large amount of data for the training phase, and the material parameters often have a rather difficult interpretation, i.e., different from microstructure descriptors such as fiber size or volume ratio.

The above-mentioned techniques essentially construct the surrogate model in a purely data-driven fashion that neglects the microscopic simulation entirely. On the other hand, also approaches exist that accelerate the microscopic simulation. One example is the Transformation Field Analysis (TFA) which was proposed in Dvorak [19] and later extended to the Nonuniform Transformation Field Analysis (NTFA) in Michel and Suquet [20]. This method is specifically suited for models including internal variables such as plasticity. The internal variables are clustered together into groups to reduce the number of degrees of freedom and averaged evolution laws are developed for each group. A similar approach, termed self-consistent clustering analysis [21], was developed that finds these clusters from linear elastic precomputations. These methods require only little data and yield physical predictions. However, they are limited to a fixed microstructure and can therefore not be used for microstructural design.

Another popular technique for dimensionality reduction is the Reduced Basis (RB) method [22,23], which can be applied to general parametric partial differential equations (pPDEs). In this method, the solution to a PDE is sought on a reduced basis spanned by global parameter-independent basis functions. Given a collection of precomputed simulations, often referred to as snapshots, the proper orthogonal decomposition (POD) can be employed to find the reduced basis. The solution of the PDE for any parameter value can then be found by either solving the reduced problem (see, e.g., [24–27]) or by using a regression-based approach (see, e.g., [28–31]). In the case where the PDE allows an affine decomposition, the method permits an offline–online decomposition and the reduced set of equations can be solved very efficiently online. However, in the case of a nonlinear problem, an affine decomposition usually does not exist and a further approximation called hyperreduction needs to be implemented. In the context of computational homogenization, Hernandez et al. used a discrete empirical interpolation method (DEIM) (see, e.g. [32,33]) on the stress field to recover the affine decomposition and accelerated an elasto-plastic RVE simulation [24]. Unfortunately, the reduced stiffness matrix is not symmetric. Soldner et al. compared three different hyperreduction approaches for a hyperelastic microstructure and showed that the non-symmetric stiffness matrix leads to convergence problems in several scenarios [25].

Instead of solving the reduced system, the regression-based approach directly predicts the parameter-dependent coefficients. With the already computed POD basis, the solution field can therefore be directly obtained. In our previous work [31], we successfully utilized this approach on a hyperelastic composite microstructure with varying stiffness of the constituents, by combining POD with Gaussian Process Regression (GPR) [34]. However, this method is not able to treat geometrical parameters such as fiber radius or shape of inclusion, which is important for material design, where optimized shapes are sought.

Within the POD framework, geometrical parameters are typically addressed through transformations that map each snapshot onto a parent domain. The surrogate model can then be constructed and solved on the parent domain. There are typically two kinds of methods to describe such domain transformations: space deformations and surface-based deformations [35]. For the former, popular techniques comprise the Free Form Deformation (FFD) [36] and Radial Basis Functions (RBF) [37], where the transformation maps are governed by the movement of control points. These methods have been used in the context of RB in various papers, see, e.g., [38–45]. The biggest benefit is that the auxiliary problem that determines the transformation map is independent of the discretization mesh of the full simulation and depends only on the number of control points used. However, since the deformation can only be prescribed at the control points, many points might be needed to sufficiently describe complicated transformations. Moreover, the boundary nodes on opposite edges of the simulation mesh of a periodic domain, which is usually assumed for a RVE, should be transformed identically, which might not be straightforward to enforce.

For surface-based deformations, knowledge of a parent simulation mesh is needed and then an auxiliary linear elliptic PDE problem is solved to obtain the transformation. Such deformations have been used in shape optimization (see, e.g., [46]) and for finite volume schemes (see, e.g., [47]). These auxiliary problems are typically more expensive to solve than the ones arising for the space deformation methods, since they scale with the full simulation mesh. On the other hand, they allow for full flexibility in terms of deformations, and periodicity can be easily enforced. Moreover, this auxiliary problem can be reduced with standard RB techniques [45], resulting in an efficient online stage. Hence, in this work, we propose an auxiliary problem based on linear elasticity to obtain the transformation maps and show how a fast-to-evaluate effective constitutive model can be constructed for a nonlinear microscopic problem. This surrogate model has the following features:

1. the predictions of the effective quantities are independent of the auxiliary problem,
2. the predicted stress fields automatically fulfill the microscopic balance laws and the periodic boundary conditions,
3. it can handle both loading parameters and microstructure descriptors (e.g., stiffness and shapes of the constituents) and thus can be used for material design,
4. it is non-intrusive and can be easily implemented into a macroscopic solver, and
5. the microscopic stress field can be recovered and visualized.

The remainder of this paper is organized as follows. Section 2 introduces the multi-scale problem based on first-order computational homogenization. In Section 3, the construction of the surrogate model for the microscopic simulation is presented in depth. Then, in Section 4, the proposed method is validated on multiple composite microstructures, where variations in the shape of the inclusions are considered, and where a full two-scale example is presented. Section 5 concludes this work with a summary on the findings and some final remarks.

In this work, the following notational convention is adopted. To differentiate between macroscopic and microscopic variables, an overline is added for the macroscopic ones. Italic bold symbols are used for coordinates $\boldsymbol{X}$ and vectorial or tensorial fields, such as the displacement $\boldsymbol{u}$ or stress field $\boldsymbol{P}$. Upright bold symbols are used for algebraic vectors and matrices, such as the identity matrix $\mathbf{I}$ or the macroscopic deformation gradient at a fixed macroscopic point $\bar{\mathbf{F}}$. A field quantity $\boldsymbol{u}$ for given parameters $\boldsymbol{\mu}$ is denoted as $\boldsymbol{u}(\boldsymbol{X}; \boldsymbol{\mu})$. Given second-order tensors $\boldsymbol{A}$ and $\boldsymbol{B}$, fourth-order tensor $\boldsymbol{C}$, and vector $\boldsymbol{v}$, the following operations are used: $\boldsymbol{A}\boldsymbol{B} = A_{ij}B_{jk}$, $\boldsymbol{A} : \boldsymbol{B} = A_{ij}B_{ij}$, $\boldsymbol{A} \otimes \boldsymbol{B} = A_{ij}B_{kl}$, $\boldsymbol{A} \otimes \boldsymbol{v} = A_{ij}v_k$, $\boldsymbol{A} : \boldsymbol{C} : \boldsymbol{B} = A_{ij}C_{ijkl}B_{kl}$ and $\boldsymbol{A}\boldsymbol{v} = A_{ij}v_j$, where the Einstein summation convention is used.

2. Problem statement

When considering engineering systems with complex microstructures, the micro- and macrostructure are both modeled and solved simultaneously in a coupled manner. Governing equations at the level of individual scales are formulated as described below.

2.1. Macroscopic problem

Consider a solid body in the reference configuration $\bar{\Omega}_0$. Under prescribed boundary conditions, every point $\bar{\boldsymbol{X}} \in \bar{\Omega}_0$ is mapped by a deformation $\bar{\boldsymbol{\Phi}}$ onto $\bar{\boldsymbol{x}} = \bar{\boldsymbol{\Phi}}(\bar{\boldsymbol{X}}) \in \bar{\Omega}$, where $\bar{\Omega}$ is the current configuration. The displacement is then defined as $\bar{\boldsymbol{u}}(\bar{\boldsymbol{X}}) := \bar{\boldsymbol{x}} - \bar{\boldsymbol{X}} = \bar{\boldsymbol{\Phi}}(\bar{\boldsymbol{X}}) - \bar{\boldsymbol{X}}$. The governing partial differential equation (PDE) describing the effective physics of the system is given by the quasi-static linear momentum balance,

$$
\begin{aligned}
\text{Div}\,\bar{\boldsymbol{P}} + \bar{\boldsymbol{B}}_0 &= \boldsymbol{0} && \text{on } \bar{\Omega}_0, \\
\bar{\boldsymbol{P}}\bar{\boldsymbol{N}}_0 &= \bar{\boldsymbol{t}}_0 && \text{on } \partial\bar{\Omega}_0^N, \text{ and} \\
\bar{\boldsymbol{u}} &= \bar{\boldsymbol{u}}_0 && \text{on } \partial\bar{\Omega}_0^D,
\end{aligned}
\tag{1}
$$

where Div denotes the divergence operator with respect to the reference configuration, $\bar{\boldsymbol{P}}$ is the macroscopic first Piola–Kirchhoff (PK1) stress tensor, $\bar{\boldsymbol{B}}_0$ are the macroscopic body forces, $\bar{\boldsymbol{N}}_0$ is the outward normal on the surface of the body, $\bar{\boldsymbol{t}}_0$ and $\bar{\boldsymbol{u}}_0$ are the prescribed tractions and displacements, and $\partial\bar{\Omega}_0^N$, $\partial\bar{\Omega}_0^D$ denote the Neumann and Dirichlet boundaries with $\partial\bar{\Omega}_0 = \partial\bar{\Omega}_0^N \cup \partial\bar{\Omega}_0^D$ and $\partial\bar{\Omega}_0^N \cap \partial\bar{\Omega}_0^D = \emptyset$. The stress tensor $\bar{\boldsymbol{P}}$ is a nonlinear operator that in general depends on parameters $\boldsymbol{\mu}$ and acts on the deformation gradient $\bar{\boldsymbol{F}}$ which is defined as

$$
\bar{\boldsymbol{F}} := \frac{\partial\bar{\boldsymbol{x}}}{\partial\bar{\boldsymbol{X}}} = \mathbf{I} + \frac{\partial\bar{\boldsymbol{u}}}{\partial\bar{\boldsymbol{X}}},
\tag{2}
$$

with $\mathbf{I}$ the identity matrix. The weak form of Eq. (1) is given as

$$
\bar{G} := \int_{\bar{\Omega}_0} \frac{\partial\delta\bar{\boldsymbol{u}}}{\partial\bar{\boldsymbol{X}}} : \bar{\boldsymbol{P}}\,dV - \int_{\bar{\Omega}_0} \bar{\boldsymbol{B}}_0 \cdot \delta\bar{\boldsymbol{u}}\,dV - \int_{\partial\bar{\Omega}_0^N} \bar{\boldsymbol{t}}_0 \cdot \delta\bar{\boldsymbol{u}}\,dA = 0, \qquad \forall\delta\bar{\boldsymbol{u}} \in H_0^1(\bar{\Omega}_0),
\tag{3}
$$

where $H_0^1(\bar{\Omega}_0) = \{\boldsymbol{v} \in H^1(\bar{\Omega}_0) \mid \boldsymbol{v} = 0 \text{ on } \partial\bar{\Omega}_0^D\}$ is the test function space with $H^1(\bar{\Omega}_0)$ a Hilbert space, and a solution for the displacement, $\bar{\boldsymbol{u}} \in H^1(\bar{\Omega}_0)$, is sought that fulfills $\bar{\boldsymbol{u}} = \bar{\boldsymbol{u}}_0$ on $\partial\bar{\Omega}_0^D$. For the Newton–Raphson method, the Gâteaux derivative of $\bar{G}$ at the current deformation $\bar{\boldsymbol{u}}$ in the direction $\Delta\bar{\boldsymbol{u}}$ is needed,

$$
D\bar{G}\big|_{\bar{\boldsymbol{u}}} \cdot \Delta\bar{\boldsymbol{u}} = \int_{\bar{\Omega}_0} \frac{\partial(\delta\bar{\boldsymbol{u}})}{\partial\bar{\boldsymbol{X}}} : \bar{\boldsymbol{A}} : \frac{\partial\Delta\bar{\boldsymbol{u}}}{\partial\bar{\boldsymbol{X}}}\,dV,
\tag{4}
$$

where $\bar{\boldsymbol{A}} := \dfrac{\partial\bar{\boldsymbol{P}}}{\partial\bar{\boldsymbol{F}}}$ is the fourth-order stiffness tensor. To find a solution to this problem, a material model needs to be specified, also known as a constitutive law, which relates the stress tensor $\bar{\boldsymbol{P}}$ to the deformation gradient $\bar{\boldsymbol{F}}$ given a set of parameters $\boldsymbol{\mu}$ (e.g., material, geometry, or loading). This can be an empirical law, e.g., obtained by fitting experimental data; however, finding an empirical law with meaningful parameters is often rather difficult. Furthermore, phenomenological laws are often insufficiently rich to capture the full complexity of the microstructural behavior. Therefore, instead, homogenization techniques are used, where a microscopic problem, defined on a representative volume element (RVE), is solved for an effective stress $\bar{\boldsymbol{P}}$ and stiffness $\bar{\boldsymbol{A}}$, given the macroscopic deformation gradient $\bar{\boldsymbol{F}}$ and a set of parameters $\boldsymbol{\mu}$, resulting in a two-scale formulation. A visualization of the two-scale simulation is shown in Fig. 1.
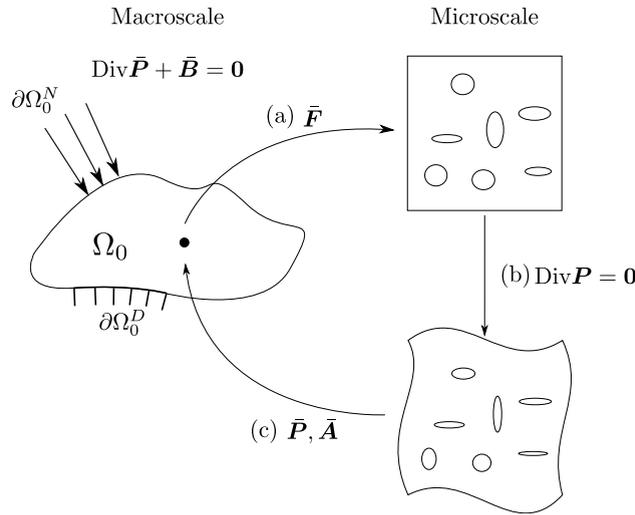
Fig. 1. Coupling of two scales. (a) In every macroscopic point the macroscopic deformation gradient is used to specify the microscopic problem. (b) The microscopic problem is then solved to obtain a microscopic stress field. (c) The stress field is subsequently averaged to obtain the effective stress and stiffness which are transferred back to the macroscale. Note that only the effective stress and stiffness are needed for the solution of the macroscopic problem.

Remark 2.1. Utilizing the polar decomposition, the deformation gradient $\bar{F}$ can always be multiplicatively decomposed into a rotation $\bar{R}$ and stretch tensor $\bar{U}$. Then, the stretch tensor $\bar{U}$ is used to evaluate the effective stress and stiffness. To obtain the effective stress and stiffness with respect to the deformation gradient $\bar{F}$, the quantities can be rotated accordingly with $\bar{R}$, for details see [48]. The advantage of this decomposition is that the number of loading parameters is reduced, since the stretch tensor $\bar{U}$ is symmetric, unlike $\bar{F}$.

2.2. Microscopic problem

In first order homogenization, the microscopic displacement field $\boldsymbol{u}(\bar{X}, X)$, with $\bar{X}$ and X denoting the macroscopic and microscopic coordinates, is assumed to consist of a macroscopic mean field $(\bar{F}(\bar{X}) - \mathbf{I})X$ and a zero-mean microscopic fluctuation field $\boldsymbol{w}(\bar{X}, X)$, i.e.

$$\boldsymbol{u}(\bar{X}, X) = (\bar{F}(\bar{X}) - \mathbf{I})X + \boldsymbol{w}(\bar{X}, X), \tag{5}$$

with the macroscopic deformation gradient $\bar{F}(\bar{X})$ depending only on the macroscopic point $\bar{X}$. For conciseness, in the following the dependence on $\bar{X}$ is omitted and the following equations are given for a fixed macroscopic point $\bar{X}$. The microscopic deformation gradient then reads

$$\boldsymbol{F}(X) := \mathbf{I} + \frac{\partial \boldsymbol{u}}{\partial X} = \bar{\mathbf{F}} + \frac{\partial \boldsymbol{w}}{\partial X}, \tag{6}$$

with

$$\langle \boldsymbol{F} \rangle = \bar{\mathbf{F}}, \tag{7}$$

where $\langle (\bullet) \rangle := |\Omega|^{-1} \int_{\Omega}(\bullet)dV$ denotes the averaging operator with $|\Omega|$ the volume of the RVE Ω. The microscopic governing PDE has the same shape as Eq. (1) and is defined on the RVE Ω:

$$\begin{aligned} \mathrm{Div}\boldsymbol{P} &= \mathbf{0} & &\text{on } \Omega, \\ \boldsymbol{u}^{+} - \boldsymbol{u}^{-} &= (\bar{\mathbf{F}} - \mathbf{I})(X^{+} - X^{-}) & &\text{on } \partial\Omega, \end{aligned} \tag{8}$$

where the body forces are neglected, $\boldsymbol{P}$ denotes the microscopic PK1 stress, and periodic boundary conditions are prescribed, with $(\bullet)^{+}$ and $(\bullet)^{-}$ denoting a quantity on opposite boundaries of the RVE. The weak form is then given

as

$$G = \int_{\Omega} \frac{\partial \delta \boldsymbol{u}}{\partial \boldsymbol{X}} : \boldsymbol{P} \, dV = 0, \qquad \forall \delta \boldsymbol{u} \in H_0^1(\Omega), \tag{9}$$

and the Gâteaux derivative in the direction $\Delta \boldsymbol{u}$ around the current state $\boldsymbol{u}$ is given by

$$DG|_{\boldsymbol{u}} \cdot \Delta \boldsymbol{u} = \int_{\Omega} \frac{\partial (\delta \boldsymbol{u})}{\partial \boldsymbol{X}} : \boldsymbol{A} : \frac{\partial \Delta \boldsymbol{u}}{\partial \boldsymbol{X}} dV. \tag{10}$$

The periodic boundary conditions can be enforced by using either Lagrange multipliers [4] or the condensation method [49].

In this work, we model each phase of the microstructure as a hyperelastic Neo-Hookean material with strain energy density function

$$W(\boldsymbol{F}, \boldsymbol{\lambda}) = C_1(\text{Tr}(\boldsymbol{C}) - 3 - 2 \ln J) + D_1(J - 1)^2, \tag{11}$$

where C_1 and D_1 are the material parameters stored in $\boldsymbol{\lambda} = [C_1, D_1]^T$, $\text{Tr}(\bullet)$ denotes the trace operator, $\boldsymbol{C} = \boldsymbol{F}^T \boldsymbol{F}$ the right Cauchy–Green deformation tensor and $J = \det(\boldsymbol{F})$ is the determinant of $\boldsymbol{F}$. The material parameters C_1 and D_1 can be rewritten into the corresponding Young's modulus E and Poisson's ratio ν with

$$E = \frac{2C_1(3D_1 + 2C_1)}{C_1 + D_1}, \quad \nu = \frac{D_1}{2(C_1 + D_1)}. \tag{12}$$

The stress and stiffness tensor are found by differentiating Eq. (11):

$$\boldsymbol{P} = \frac{\partial W}{\partial \boldsymbol{F}}, \quad \boldsymbol{A} = \frac{\partial \boldsymbol{P}}{\partial \boldsymbol{F}}. \tag{13}$$

After solving the microscopic problem, the effective stress $\bar{\boldsymbol{P}}$ can be obtained by averaging the microscopic stress, i.e.,

$$\bar{\boldsymbol{P}} = \langle \boldsymbol{P} \rangle. \tag{14}$$

However, computing the effective stiffness $\bar{\boldsymbol{A}}$ is more complicated, since

$$\langle \boldsymbol{A} \rangle \neq \bar{\boldsymbol{A}}; \tag{15}$$

see, e.g., [50]. Different ways of computing a consistent effective stiffness have been derived in [4,49]. Another possible way is to numerically approximate it with a finite difference scheme [51].

3. Surrogate modeling

Since the microscopic problem needs to be solved at every quadrature point of the macroscopic problem in each Newton iteration, running even a single full two-scale simulation is expensive. In multi-query contexts such as optimization or material design, it is therefore needed to accelerate the simulation. Several methods have been proposed to replace the microscopic model with a cheap-to-evaluate surrogate model. One of the most powerful tools for dimensionality reduction is the so-called Reduced Basis (RB) method. The reduced basis is often obtained by employing a Proper Orthogonal Decomposition (POD). Even though powerful for loading and material parameters, geometrical parameters cannot be easily treated since the snapshots generally need to be transformed first onto a parent domain for accurate approximations. To address this problem, we present in this section a surface-based deformation method for finding such geometrical transformations by solving an auxiliary problem based on linear elasticity. We then subsequently show how to construct the surrogate model and replace the microscopic simulation.

3.1. Proper orthogonal decomposition

For convenience, we differentiate between macroscopic loading parameters $\bar{\boldsymbol{U}}$, material parameters $\boldsymbol{\lambda}$ and geometrical parameters $\boldsymbol{\mu}$. Given a set of snapshots $\{\boldsymbol{P}^i\}_{i=1}^{N_s} \in \mathcal{V}_h$, where $\mathcal{V}_h$ is a discretized function space with $\dim \mathcal{V}_h = \mathcal{N}$, N_s is the number of snapshots and $\boldsymbol{P}^i$ is the field variable corresponding to a given choice of

parameters $(\bar{\mathbf{U}}^i, \boldsymbol{\lambda}^i, \boldsymbol{\mu}^i)$, the proper orthogonal decomposition utilizes the correlation between the snapshots to find an optimal subspace in $\mathcal{V}_h$. Then, the field variable $\boldsymbol{P}$ can be approximated with

$$\boldsymbol{P}(\boldsymbol{X}; \bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \approx \sum_{n=1}^{N} \alpha_n(\bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \boldsymbol{B}_n(\boldsymbol{X}), \tag{16}$$

where the $\boldsymbol{B}_n(\boldsymbol{X})$ are global parameter-independent basis functions, $\alpha_n(\bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu})$ are parameter-dependent coefficients, and N is the number of basis functions, where, ideally, $N \ll \mathcal{N}$. More details on the computation of the basis functions $\boldsymbol{B}_n(\boldsymbol{X})$ can be found in [31].

3.2. Transformation of snapshots

For geometrical parameters, a reduced basis obtained with POD will in general perform poorly and might exhibit artificial oscillatory effects [52]. Furthermore, changing the geometry will also affect the simulation mesh, leading to incompatible snapshots, since the field variable is only known at different discrete points and some type of extrapolation and interpolation needs to be adopted. In order to apply POD on such a set of snapshots, one would first need to interpolate each snapshot onto the same underlying grid of discrete points, leading to additional interpolation errors. To overcome these issues, the snapshots can be transformed onto a parent domain first. In most existing works, such transformations are obtained with space deformation techniques (see, e.g., [38–44]) relying only on a few control points. The biggest advantage of these methods is that obtaining the transformation map only scales with the number of control points and is independent of the original simulation mesh. However, several control points might be needed to describe complicated geometries.

In the case of surface-based deformation techniques, an auxiliary diffusion-type PDE problem is posed and solved on the simulation mesh of the original problem. Hence, the transformation map has exactly the same flexibility as the solution field in the original simulation. Furthermore, one can ensure by construction that the obtained transformation is a bijective map. To achieve these properties, we propose to find the transformations by solving an auxiliary problem based on linear elasticity, which is closely related to techniques presented in [35,46,47]. Even though the auxiliary problem formulated this way is more expensive to solve than the one arising from space deformation techniques, the solution can be substantially accelerated with standard reduced basis (RB) methods, as shown in Section 3.2.1 below.

The main idea is as follows: for any geometrical parameterization, the movement of certain parts of the geometry is known, while other parts are fixed. This movement can be prescribed onto a parent mesh and then an auxiliary problem is solved to obtain the transformation map for all geometrical parameters. The example below illustrates this approach.

2D RVE with elliptical inclusion. Consider a RVE defined on $\Omega = [0, 1]^2$ consisting of an elliptical inclusion embedded in a homogeneous matrix. The shape of the inclusion is parameterized, i.e., the major and minor axis can be changed. Assuming a parent domain Ω^p with a given circular inclusion, it can be deformed into any of the parameterized domains $\Omega^{\boldsymbol{\mu}}$, by moving the points on the circular interface into the shape of the elliptical inclusion while keeping the outer boundaries constant, see Fig. 2. By defining this transformation map as

$$\boldsymbol{\Phi}_{\boldsymbol{\mu}} : \Omega^p \to \Omega^{\boldsymbol{\mu}}, \boldsymbol{X}^p \mapsto \boldsymbol{X}^{\boldsymbol{\mu}} = \boldsymbol{\Phi}_{\boldsymbol{\mu}}(\boldsymbol{X}^p) = \boldsymbol{X}^p + \boldsymbol{d}(\boldsymbol{X}^p; \boldsymbol{\mu}) \tag{17}$$

with $\boldsymbol{d}$ the transformation displacement, this can now be translated into the following linear-elastic auxiliary problem:

$$\mathrm{Div}\left(\mathbb{C} : \frac{1}{2}\left(\frac{\partial \boldsymbol{d}}{\partial \boldsymbol{X}^p} + \left(\frac{\partial \boldsymbol{d}}{\partial \boldsymbol{X}^p}\right)^T\right)\right) = \boldsymbol{0} \qquad \text{in } \Omega^p, \tag{18a}$$

$$\boldsymbol{d} = \boldsymbol{0} \qquad \text{on } \partial\Omega^p, \tag{18b}$$

$$\boldsymbol{d} = \boldsymbol{X}^{\boldsymbol{\mu}}(\boldsymbol{X}^p) - \boldsymbol{X}^p \text{ on } \partial\Omega_{\mathrm{int}}^p, \tag{18c}$$

where $\partial\Omega^p = \partial\Omega_L^p \cup \partial\Omega_R^p \cup \partial\Omega_B^p \cup \partial\Omega_T^p$ denotes the union of the left, right, bottom and top RVE boundaries, and $\boldsymbol{X}^{\boldsymbol{\mu}}(\boldsymbol{X}^p)$ is known for all points on the parent interface $\partial\Omega_{\mathrm{int}}^p$. The elasticity tensor $\mathbb{C}$ is assumed to be constant throughout the whole domain, assumed in the form of Hooke's law, and specified by the Young's modulus and
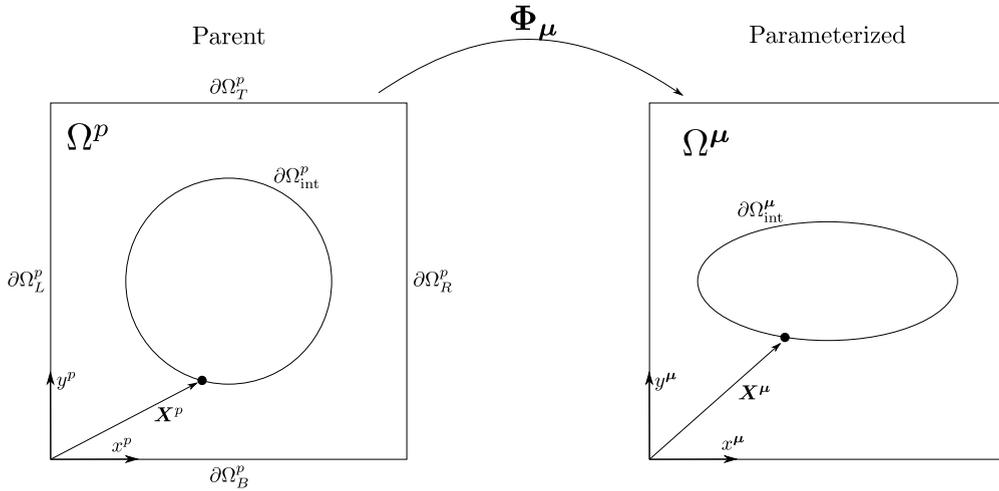
Fig. 2. Definition of parent and parameterized domain. The transformation map $\boldsymbol{\Phi_\mu}$ maps from the parent Ω^p to the parameterized Ω^μ domain. The transformation displacement $\boldsymbol{d}$ on the boundaries is fixed. The points on both the parent $\partial\Omega_{\text{int}}^p$ and parameterized interface $\partial\Omega_{\text{int}}^\mu$ are known and used to prescribe the transformation along the interface. As an example, the indicated point $\boldsymbol{X}^p$ in the parent domain is displaced onto the indicated point $\boldsymbol{X}^\mu$ in the parameterized domain.

Poisson's ratio ξ. Since the problem is cast in a purely geometric manner, the Young's modulus has no influence on the transformation map, as it only changes the magnitude of the stresses that are of no interest here. The Poisson's ratio ξ changes the compressibility of the material, hence affecting the transformation, and its influence on the approximation will be investigated in Section 4. The boundary condition in Eq. (18b) is chosen such that the parameterized domain $\boldsymbol{\Phi_\mu}(\Omega^p) = [0, 1]^2$ remains the same, i.e., covers the parent domain Ω^p. This also means that the transformation preserves the volume, i.e. $|\Omega^p| = |\boldsymbol{\Phi_\mu}(\Omega^p)|$. Moreover, this way a periodic quantity remains periodic after transformation. Finally, Eq. (18c) prescribes the transformation displacements along the interface to deform the circle into an ellipse.

Remark 3.1. In principle, one could replace Eq. (18b) with periodic boundary conditions and fix the RVE at the corner points to allow more flexible transformations. However, the added complexity did not yield any significant improvement in accuracy in our test problems.

3.2.1. Discretization and numerical solution of auxiliary problem

Discretizing the auxiliary problem in Eq. (18) yields a linear system of equations

$$\mathbf{A}\mathbf{d} = \mathbf{b}(\boldsymbol{\mu}), \tag{19}$$

where the size of $\mathbf{A}$, $\mathbf{d}$ and $\mathbf{b}(\boldsymbol{\mu})$ scale with the full mesh dimensionality $\mathcal{N}$ and Eq. (19) needs to be solved for many right-hand sides, where each solution is typically computationally expensive. However, there are several ways to accelerate the solution: since the stiffness matrix $\mathbf{A}$ does not depend on the parameters $\boldsymbol{\mu}$ and is symmetric positive definite, for moderate values of $\mathcal{N}$, a Cholesky decomposition $\mathbf{A} = \mathbf{L}\mathbf{L}^T$ can be computed once and then a forward and backward substitution can be used to solve Eq. (19) for any $\mathbf{b}(\boldsymbol{\mu})$. For even larger values of $\mathcal{N}$, POD can be used to find a reduced basis for the transformation displacement $\mathbf{d} = \mathbf{V}\hat{\mathbf{d}}$ with $\dim(\hat{\mathbf{d}}) = N \ll \mathcal{N}$ and the reduced system is obtained via a Galerkin projection, i.e.,

$$\mathbf{V}^T\mathbf{A}\mathbf{V}\hat{\mathbf{d}} = \mathbf{V}^T\mathbf{b}(\boldsymbol{\mu}), \tag{20}$$

$$\Rightarrow \hat{\mathbf{A}}\hat{\mathbf{d}} = \mathbf{V}^T\mathbf{b}(\boldsymbol{\mu}), \tag{21}$$

where the reduced stiffness matrix $\hat{\mathbf{A}} := \mathbf{V}^T\mathbf{A}\mathbf{V}$ only needs to be computed once. In Section 4, it will be shown that N is equal to the number of geometrical parameters $\boldsymbol{\mu}$. However, as pointed out in [45], the term on the right hand side $\mathbf{V}^T\mathbf{b}(\boldsymbol{\mu})$ is in general not affinely decomposable, meaning that for each new value of $\boldsymbol{\mu}$, a matrix–vector

product $\mathbf{V}^T\mathbf{b}(\boldsymbol{\mu})$ must be computed which depends linearly on the full problem size $\mathcal{O}(\mathcal{N})$. If this is prohibitive, then a discrete empirical interpolation method [32,33] could be used to approximate $\mathbf{b}(\boldsymbol{\mu})$ and then the complexity of the solution of the auxiliary problem becomes independent of $\mathcal{N}$.

After solving Eq. (19), the transformation displacements $\boldsymbol{d}(X^p;\boldsymbol{\mu})$ are specified everywhere and hence the map $\boldsymbol{\Phi}_{\boldsymbol{\mu}}$ is obtained. To obtain the snapshots on the parent domain $\boldsymbol{P}^p(X^p)$, the original snapshots $\boldsymbol{P}(X^{\boldsymbol{\mu}})$, computed on the parameterized domain $\Omega^{\boldsymbol{\mu}}$, have to be evaluated at the transformed parent coordinates, i.e.,

$$\boldsymbol{P}(X^{\boldsymbol{\mu}}) = \boldsymbol{P}(\boldsymbol{\Phi}_{\boldsymbol{\mu}}(X^p)) =: \boldsymbol{P}^p(X^p). \tag{22}$$

Remark 3.2. If the original snapshot $\boldsymbol{P}(X^{\boldsymbol{\mu}})$ is obtained on an independent mesh, one needs to interpolate it onto the transformed parent coordinates $\boldsymbol{\Phi}_{\boldsymbol{\mu}}(X^p)$. However, this interpolation introduces non-physical effects, i.e., periodicity and/or linear momentum balance might not be fulfilled anymore. To ensure a physical transformation, one can use the auxiliary problem in Eq. (18) to generate a simulation mesh for the physical simulation, because that way the snapshots are directly found on the transformed parent coordinates and no interpolation step is needed anymore.

3.3. Surrogate model for microsimulation

In order to replace the microscopic simulation, a surrogate model for the effective stress and stiffness is required, which represents an effective constitutive law. In our previous work [31], we constructed the surrogate model by finding a reduced representation of the stress field via POD,

$$\boldsymbol{P}(X;\bar{\mathbf{U}},\boldsymbol{\lambda}) \approx \sum_{n=1}^{N} \alpha_n(\bar{\mathbf{U}},\boldsymbol{\lambda})\boldsymbol{B}_n(X), \tag{23}$$

where geometrical parameters were not considered. The effective stress was then expressed as

$$\bar{\mathbf{P}}(\bar{\mathbf{U}},\boldsymbol{\lambda}) \approx \sum_{n=1}^{N} \alpha_n(\bar{\mathbf{U}},\boldsymbol{\lambda})\bar{\mathbf{B}}_n, \tag{24}$$

where $\bar{\mathbf{B}}_n := |\Omega|^{-1}\int_{\Omega}\boldsymbol{B}_n dX$. Regression maps $(\bar{\mathbf{U}},\boldsymbol{\lambda}) \mapsto \alpha_n$ for the coefficients $\alpha_n(\bar{\mathbf{U}},\boldsymbol{\lambda})$ were learned with Gaussian Process Regression (GPR) [28,34]. Since each stress field snapshot used for the POD is periodic and fulfills the linear momentum balance, the basis functions $\boldsymbol{B}_n(X)$ fulfill these conditions as well. This is because a sum of periodic functions is still periodic, while the linear momentum balance follows from the linearity of the approximation and the fact that each $\boldsymbol{B}_n$ satisfies the linear momentum balance.

For geometrically parameterized domains, however, stress snapshots cannot be used directly. In order to satisfy balance of linear momentum in the parameterized domain, stress snapshots need to be mapped and transformed from the parent to parameterized domain through a corresponding mapping $\boldsymbol{\Phi}_{\boldsymbol{\mu}}$ and its deformation gradient $\boldsymbol{F}_{\boldsymbol{\mu}} := \frac{\partial\boldsymbol{\Phi}_{\boldsymbol{\mu}}}{\partial X^p}$, derived as follows. The weak form of the linear momentum balance on the parameterized domain $\Omega^{\boldsymbol{\mu}}$ reads

$$\int_{\Omega^{\boldsymbol{\mu}}} \frac{\partial(\delta\boldsymbol{u}(X^{\boldsymbol{\mu}}))}{\partial X^{\boldsymbol{\mu}}} : \boldsymbol{P}(X^{\boldsymbol{\mu}};\bar{\mathbf{U}},\boldsymbol{\lambda},\boldsymbol{\mu})dX^{\boldsymbol{\mu}} = 0, \tag{25}$$

where $\delta\boldsymbol{u} \in H_0^1(\Omega^{\boldsymbol{\mu}})$ is a test function and $X^{\boldsymbol{\mu}}$ denotes the coordinates on the parameterized domain. By introducing the transformation from Eq. (17) and $dX^{\boldsymbol{\mu}} = |\det\boldsymbol{F}_{\boldsymbol{\mu}}|dX^p$, the left hand side of Eq. (25) becomes

$$\int_{\Omega^{\boldsymbol{\mu}}} \frac{\partial(\delta\boldsymbol{u}(X^{\boldsymbol{\mu}}))}{\partial X^{\boldsymbol{\mu}}} : \boldsymbol{P}(X^{\boldsymbol{\mu}};\bar{\mathbf{U}},\boldsymbol{\lambda},\boldsymbol{\mu})dX^{\boldsymbol{\mu}} = \int_{\Omega^p} \frac{\partial(\delta\boldsymbol{u}^p(X^p))}{\partial X^p}\boldsymbol{F}_{\boldsymbol{\mu}}^{-1} : \boldsymbol{P}(\boldsymbol{\Phi}_{\boldsymbol{\mu}}(X^p);\bar{\mathbf{U}},\boldsymbol{\lambda},\boldsymbol{\mu})|\det\boldsymbol{F}_{\boldsymbol{\mu}}|dX^p$$
$$= \int_{\Omega^p} \frac{\partial(\delta\boldsymbol{u}^p(X^p))}{\partial X^p} : \boldsymbol{P}(\boldsymbol{\Phi}_{\boldsymbol{\mu}}(X^p);\bar{\mathbf{U}},\boldsymbol{\lambda},\boldsymbol{\mu})\boldsymbol{F}_{\boldsymbol{\mu}}^{-T}|\det\boldsymbol{F}_{\boldsymbol{\mu}}|dX^p, \tag{26}$$

where $\delta\boldsymbol{u}^p(X^p) = \delta\boldsymbol{u}(\boldsymbol{\Phi}_{\boldsymbol{\mu}}(X^p))$ is a test function in the parent configuration. From Eq. (26) we can see that if the stress field $\boldsymbol{P}(\boldsymbol{\Phi}_{\boldsymbol{\mu}}(X^p);\bar{\mathbf{U}},\boldsymbol{\lambda},\boldsymbol{\mu})$ were directly approximated using POD, then the linear momentum balance Eq. (25) on the parameterized domain would not necessarily hold due to the effects of $\boldsymbol{F}_{\boldsymbol{\mu}}$. However, if these effects are approximated together with the stress as a weighted stress, i.e.,

$$\boldsymbol{P}(\boldsymbol{\Phi}_{\boldsymbol{\mu}}(X^p);\bar{\mathbf{U}},\boldsymbol{\lambda},\boldsymbol{\mu})\boldsymbol{F}_{\boldsymbol{\mu}}^{-T}|\det\boldsymbol{F}_{\boldsymbol{\mu}}| \approx \sum_{n=1}^{N} \alpha_n(\bar{\mathbf{U}},\boldsymbol{\lambda},\boldsymbol{\mu})\boldsymbol{B}_n(X^p), \tag{27}$$

then each basis function $\boldsymbol{B}_n$ automatically fulfills the linear momentum balance on any of the parameterized domains, since the basis functions are computed from linear combinations of snapshots that fulfill the balance equation on different domains. This can be shown by inserting Eq. (27) into Eq. (26):

$$\int_{\Omega^P} \frac{\partial(\delta \boldsymbol{u}^P(\boldsymbol{X}^P))}{\partial \boldsymbol{X}^P} : \boldsymbol{P}(\boldsymbol{\Phi}_{\boldsymbol{\mu}}(\boldsymbol{X}^P); \bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \boldsymbol{F}_{\boldsymbol{\mu}}^{-T} \left| \det \boldsymbol{F}_{\boldsymbol{\mu}} \right| d\boldsymbol{X}^P \tag{28}$$

$$\approx \sum_{n=1}^{N} \alpha_n(\bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \int_{\Omega^P} \frac{\partial(\delta \boldsymbol{u}^P(\boldsymbol{X}^P))}{\partial \boldsymbol{X}^P} : \boldsymbol{B}_n(\boldsymbol{X}^P) d\boldsymbol{X}^P. \tag{29}$$

Since the basis functions are linear combinations of N_s training snapshots, i.e.,

$$\boldsymbol{B}_n(\boldsymbol{X}^P) = \sum_{l=1}^{N_s} a_{nl} \boldsymbol{P}^{(l)} \boldsymbol{F}_{\boldsymbol{\mu}^{(l)}}^{-T} \left| \det \boldsymbol{F}_{\boldsymbol{\mu}^{(l)}} \right|, \tag{30}$$

where a_{nl} are the corresponding coefficients determined by POD and the superscript (l) denotes the lth weighted stress snapshot, obtained for parameters $(\bar{\mathbf{U}}^{(l)}, \boldsymbol{\lambda}^{(l)}, \boldsymbol{\mu}^{(l)})$, inserting Eq. (30) into Eq. (29) yields

$$\sum_{n=1}^{N} \alpha_n(\bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \int_{\Omega^P} \frac{\partial(\delta \boldsymbol{u}^P(\boldsymbol{X}^P))}{\partial \boldsymbol{X}^P} : \boldsymbol{B}_n(\boldsymbol{X}^P) d\boldsymbol{X}^P \tag{31}$$

$$= \sum_{n=1}^{N} \alpha_n(\bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \sum_{l=1}^{N_s} a_{nl} \underbrace{\int_{\Omega^P} \frac{\partial(\delta \boldsymbol{u}^P(\boldsymbol{X}^P))}{\partial \boldsymbol{X}^P} : \boldsymbol{P}^{(l)} \boldsymbol{F}_{\boldsymbol{\mu}^{(l)}}^{-T} \left| \det \boldsymbol{F}_{\boldsymbol{\mu}^{(l)}} \right| d\boldsymbol{X}^P}_{=0}, \tag{32}$$

where each of the integrals equals 0, as every training snapshot fulfills Eq. (25). Therefore, any predicted weighted stress will always fulfill Eq. (25) for any parameterized domains $\Omega^{\boldsymbol{\mu}}$. To obtain the stress field from the predicted weighted stress, one therefore has to multiply it with $\boldsymbol{F}_{\boldsymbol{\mu}}^T \left| \det \boldsymbol{F}_{\boldsymbol{\mu}} \right|^{-1}$. Furthermore, periodicity follows from the periodicity of the stress field and the transformation map.

3.3.1. Effective stress

The effective (average) stress on the parameterized domain $\Omega^{\boldsymbol{\mu}}$ is computed as

$$\bar{\mathbf{P}}(\bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = |\Omega^{\boldsymbol{\mu}}|^{-1} \int_{\Omega^{\boldsymbol{\mu}}} \boldsymbol{P}(\boldsymbol{X}^{\boldsymbol{\mu}}; \bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) d\boldsymbol{X}^{\boldsymbol{\mu}}, \tag{33}$$

where $|\Omega^{\boldsymbol{\mu}}|$ denotes the volume of the RVE. Note that $|\Omega^{\boldsymbol{\mu}}| = |\Omega^P|$ for all $\boldsymbol{\mu}$ due to the volume-preserving transformation map. Pulling the integral back into the parent domain gives

$$\bar{\mathbf{P}}(\bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = |\Omega^P|^{-1} \int_{\Omega^P} \boldsymbol{P}(\boldsymbol{\Phi}_{\boldsymbol{\mu}}(\boldsymbol{X}^P); \bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \left| \det \boldsymbol{F}_{\boldsymbol{\mu}} \right| d\boldsymbol{X}^P, \tag{34}$$

and using the approximation for $\boldsymbol{P}(\boldsymbol{\Phi}_{\boldsymbol{\mu}}(\boldsymbol{X}^P); \bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu})$ from Eq. (27) then yields

$$\bar{\mathbf{P}}(\bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \approx |\Omega^P|^{-1} \int_{\Omega^P} \left(\sum_{n=1}^{N} \alpha_n(\bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \boldsymbol{B}_n(\boldsymbol{X}^P) \right) \boldsymbol{F}_{\boldsymbol{\mu}}^T \frac{1}{|\det \boldsymbol{F}_{\boldsymbol{\mu}}|} |\det \boldsymbol{F}_{\boldsymbol{\mu}}| d\boldsymbol{X}^P \tag{35}$$

$$= |\Omega^P|^{-1} \int_{\Omega^P} \left(\sum_{n=1}^{N} \alpha_n(\bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \boldsymbol{B}_n(\boldsymbol{X}^P) \right) \boldsymbol{F}_{\boldsymbol{\mu}}^T d\boldsymbol{X}^P. \tag{36}$$

Since α_n does not depend on $\boldsymbol{X}^P$, it can be taken out of the integral, yielding

$$\bar{\mathbf{P}}(\bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = |\Omega^P|^{-1} \sum_{n=1}^{N} \alpha_n(\bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \int_{\Omega^P} \boldsymbol{B}_n(\boldsymbol{X}^P) \boldsymbol{F}_{\boldsymbol{\mu}}^T d\boldsymbol{X}^P. \tag{37}$$

In order to have a rapid online phase, the integrals must be precomputed in the offline stage. Due to the specific form of the basis functions $\boldsymbol{B}_n$, it can be shown that the integral in Eq. (37) is invariant with respect to $\boldsymbol{F}_{\boldsymbol{\mu}}^T$, i.e.,

$$\int_{\Omega^P} \boldsymbol{B}_n(\boldsymbol{X}^P) \boldsymbol{F}_{\boldsymbol{\mu}}^T d\boldsymbol{X}^P = \int_{\Omega^P} \boldsymbol{B}_n(\boldsymbol{X}^P) d\boldsymbol{X}^P, \tag{38}$$

which can be precomputed since $\boldsymbol{B}_n$ is known; the proof of the identity in Eq. (38) is provided in Appendix A. With

$$\bar{\mathbf{B}}_n := |\Omega^p|^{-1} \int_{\Omega^p} \boldsymbol{B}_n(X^p) dX^p, \tag{39}$$

the expression for the effective stress in Eq. (37) becomes

$$\bar{\mathbf{P}}(\bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \sum_{n=1}^{N} \alpha_n(\bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \bar{\mathbf{B}}_n, \tag{40}$$

which remarkably has exactly the same form as Eq. (24), with the additional geometrical parameter dependence in the coefficients α_n. Hence, the prediction of the effective stresses is completely independent of the auxiliary problem, which only needs to be computed to recover the stress field.

3.3.2. Effective stiffness and sensitivities

The effective constitutive stiffness and sensitivities with respect to material and geometrical parameters are given as

$$\bar{A} = \frac{\partial \bar{\mathbf{P}}}{\partial \bar{\mathbf{U}}} = \sum_{n=1}^{N} \bar{\mathbf{B}}_n \otimes \frac{\partial \alpha_n}{\partial \bar{\mathbf{U}}}, \tag{41}$$

$$\frac{\partial \bar{\mathbf{P}}}{\partial \boldsymbol{\lambda}} = \sum_{n=1}^{N} \bar{\mathbf{B}}_n \otimes \frac{\partial \alpha_n}{\partial \boldsymbol{\lambda}}, \tag{42}$$

$$\frac{\partial \bar{\mathbf{P}}}{\partial \boldsymbol{\mu}} = \sum_{n=1}^{N} \bar{\mathbf{B}}_n \otimes \frac{\partial \alpha_n}{\partial \boldsymbol{\mu}}. \tag{43}$$

Given coefficients α_n and their derivatives, the effective stress, stiffness and sensitivities can therefore be directly obtained. These coefficients can be approximated with different regression models, such as radial basis functions [53], GPR [28,54] or neural networks [29]. A comparison in this context was carried out in [55], revealing that GPRs can be as accurate as NNs, while being easier to train. Moreover, a GP model also returns an uncertainty measure for every prediction, which can be utilized to construct active learning schemes, see, e.g., [28,54,56]. Due to these advantages, in this work, we learn regression models for $\alpha_n(\bar{\mathbf{U}}, \boldsymbol{\lambda}, \boldsymbol{\mu})$ with GPRs. Note that in this work, we do not use the uncertainty measure to adaptively enrich the training set, as in [28,54,56]; this will be explored in future work. A broad overview and theory on GPRs can be found in [34], and is omitted here for brevity. For all GPR models, the Python library GPy [57] with automatic relevance determination squared exponential kernels has been used. The optimal hyperparameters of the kernels are determined by a maximum likelihood estimation, as presented in [28,34], with the L-BFGS-B algorithm. In the numerical tests, we did not encounter any problems during hyperparameter tuning and the convergence of the L-BFGS-B algorithm was smooth.

3.4. Offline–online decomposition

For convenience, the full offline–online decomposition is summarized in Algorithm 1.

4. Example problems

In this section, the proposed method, in the following referred to as PODGPR, is first applied onto two geometrically parameterized microstructures to showcase the generality of the approach and its accuracy. Then, a two-scale Cook's membrane problem is shown to illustrate the speed-up and its potential applications. All examples are defined in 2D under plane strain conditions, although the proposed methodology can easily be extended to 3D microstructures.

For convenience, all quantities in all examples are normalized, and dimensionless quantities are considered. At the same time, all RVEs are assumed to be of size $[0, 1]^2$.

Algorithm 1 Offline–online decomposition of the proposed PODGPR framework with microstructures parameterized with external loading $\bar{\mathbf{U}}$, microstructural material parameters $\boldsymbol{\lambda}$ and geometrical features $\boldsymbol{\mu}$.

Offline Stage:

1: Define a parent domain Ω^p and its finite element discretization.
2: Generate parameter samples $\{\bar{\mathbf{U}}^i, \boldsymbol{\lambda}^i, \boldsymbol{\mu}^i\}_{i=1}^{N_s}$ from a random distribution.
3: For each different set of geometrical parameters $\boldsymbol{\mu}^i$, solve the auxiliary problem in Eq. (18) to obtain the transformation map $\boldsymbol{\Phi}_{\boldsymbol{\mu}^i}$.
4: Use the transformation map $\boldsymbol{\Phi}_{\boldsymbol{\mu}^i}$ to generate simulation meshes for each parameter sample $\boldsymbol{\mu}^i$ and then run full simulations to obtain stress snapshots $\boldsymbol{P}^i(\boldsymbol{\Phi}_{\boldsymbol{\mu}^i}(\boldsymbol{X}^p); \bar{\mathbf{U}}^i, \boldsymbol{\lambda}^i, \boldsymbol{\mu}^i)$.
5: Compute POD of the weighted stress snapshots $\boldsymbol{P}^i(\boldsymbol{\Phi}_{\boldsymbol{\mu}^i}(\boldsymbol{X}^p); \bar{\mathbf{U}}^i, \boldsymbol{\lambda}^i, \boldsymbol{\mu}^i)\boldsymbol{F}_{\boldsymbol{\mu}^i}^{-T}|\det \boldsymbol{F}_{\boldsymbol{\mu}^i}|$ on the parent domain, cf. Eq. (27).
6: Project weighted stress snapshots onto POD basis and learn GPRs for the POD coefficients.
7: Compute $\bar{\mathbf{B}}_n$ using Eq. (39).

Online Stage:

1: Given a new parameter set $(\bar{\mathbf{U}}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$, evaluate α_n using GPRs.
2: Compute effective stress using Eq. (40) and effective sensitivities with Eqs. (41)–(43)

The nonlinear physical simulations are solved within the Finite Element framework MOOSE [58] and the linear auxiliary problems are solved with an in-house code[1] written in Python. As mentioned in Remark 3.2, to obtain physically consistent transformations without the need for interpolation, we use the auxiliary problem to generate simulation meshes. To quantify the quality of the approximation, the following two error measures are introduced:

1. Relative error of stress field

$$\mathcal{E}_{\boldsymbol{P}} = \frac{\|\boldsymbol{P}^{\text{truth}} - \boldsymbol{P}^{\text{surrogate}}\|_{L^2(\Omega^{\boldsymbol{\mu}})}}{\|\boldsymbol{P}^{\text{truth}}\|_{L^2(\Omega^{\boldsymbol{\mu}})}}, \tag{44}$$

2. Relative error of effective stress

$$\mathcal{E}_{\bar{\mathbf{P}}} = \frac{\|\bar{\mathbf{P}}^{\text{truth}} - \bar{\mathbf{P}}^{\text{surrogate}}\|_F}{\|\bar{\mathbf{P}}^{\text{truth}}\|_F}, \tag{45}$$

where $(\bullet)^{\text{truth}}$ and $(\bullet)^{\text{surrogate}}$ indicate the full and approximate solution, and $\|(\bullet)\|_{L^2(\Omega^{\boldsymbol{\mu}})}$ and $\|(\bullet)\|_F$ denote the $L^2(\Omega^{\boldsymbol{\mu}})$ and the Frobenius norm, with $\Omega^{\boldsymbol{\mu}}$ the parameterized domain. The average errors for given testing datasets are defined as:

$$\bar{\mathcal{E}}_{\boldsymbol{P}} = \frac{1}{N_{\text{test}}} \sum_{n=1}^{N_{\text{test}}} \mathcal{E}_{\boldsymbol{P}}^n, \qquad \bar{\mathcal{E}}_{\bar{\mathbf{P}}} = \frac{1}{N_{\text{test}}} \sum_{n=1}^{N_{\text{test}}} \mathcal{E}_{\bar{\mathbf{P}}}^n, \tag{46}$$

where N_{test} is the number of testing snapshots and $\mathcal{E}_{\boldsymbol{P}}^n$ and $\mathcal{E}_{\bar{\mathbf{P}}}^n$ correspond to the relative errors of the nth snapshot.

For comparison, we also trained several deep feed-forward neural networks for the effective stress using the same data for each example, similar to [31]. All considered neural networks have as many inputs as the number of parameters, 4 outputs for each stress component and 2 hidden layers each with N_n neurons. We trained four architectures with $N_n \in \{50, 100, 200, 300\}$. These four architectures will be referred to as NN1, NN2, NN3 and NN4. ELU activation functions are applied on each layer apart from the last layer. For the optimization, the mean squared error loss function is chosen and optimized with the *Adam* optimizer [59] with a learning rate of 1×10^{-4} and a batch size of 32 for 10000 epochs. The training is performed with the Python package PyTorch [60].

4.1. Composite microstructure with an elliptical fiber

In this example, a composite structure, consisting of a soft matrix and an elliptical stiff fiber around the center of the domain $\mathbf{X}_c = [0.5, 0.5]^T$, is considered. Three geometrical parameters that parameterize the fiber shape,

---

[1] The implementation can be found on https://github.com/theronguo/auxiliary-problem.
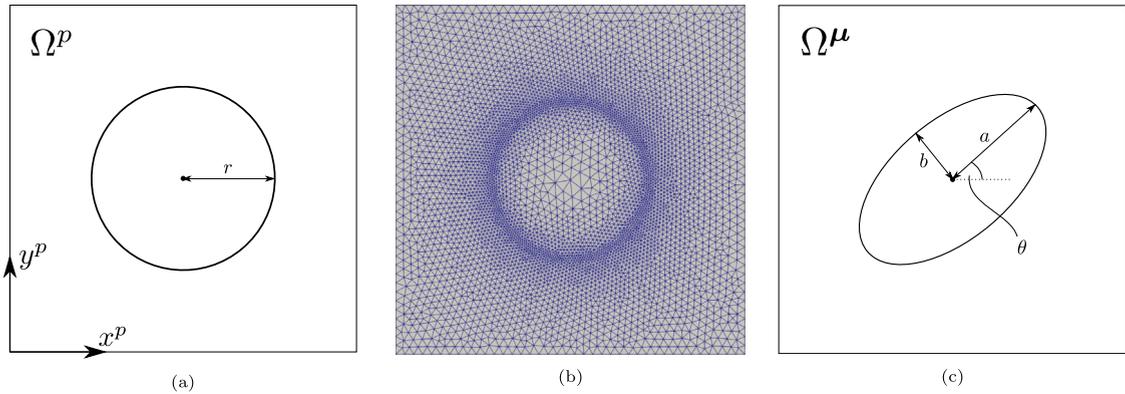
Fig. 3. Parent and parameterized domain. (a) The chosen parent domain with a circular interface with fixed radius $a = b = r = 0.225$ with (b) its corresponding mesh, consisting of a total of 20769 nodes and 10665 6-node triangular elements. (c) The parameterized domain, characterized by semi-major axis a, semi-minor axis b and angle θ.

the semi-major axis a, semi-minor axis b and a rotation angle θ, are considered, see Fig. 3(c). Together with the three loading directions $\bar{U}_{xx}$, $\bar{U}_{yy}$ and $\bar{U}_{xy}$, this problem has 6 parameters. For the matrix material a Neo–Hookean material model with $C_1 = 1$ and $D_1 = 1$ is chosen, while for the fiber material a Neo-Hookean material model with $C_1 = 100$ and $D_1 = 100$ is assumed, corresponding to a stiffness that is 100 times higher than the matrix material. Both materials have Poisson's ratio 0.25. The considered parameter ranges are given in Table 1. Lower and upper bounds for $\bar{\mathbf{U}}$ are chosen such that the solution of the microstructural problem converges for all parameter configurations. For some geometrical parameters, a few elements of the parent mesh might become highly distorted after applying the geometrical transformation. Together with the high contrast of material stiffness of both materials, larger magnitudes than 0.15 of the components $\bar{\mathbf{U}} - \mathbf{I}$ lead to convergence issues. If larger magnitudes have to be considered, a different parent domain could be employed or a smaller parameter space of the geometrical parameters could be chosen.

4.1.1. Setup of the auxiliary problem

For the parent domain, a domain with a circular inclusion with radius $r = 0.225$ is chosen, see Fig. 3(a). The corresponding mesh, consisting of 20769 nodes and 10665 6-node triangular elements, is shown in Fig. 3(b). The transformation of the circular interface into the elliptical interface can be given as:

$$\mathbf{X}^{\mu}(\mathbf{X}^p) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} a/r & 0 \\ 0 & b/r \end{bmatrix} \begin{pmatrix} \tilde{r}(\mathbf{X}^p - \mathbf{X}_c)\cos(\tilde{\theta}(\mathbf{X}^p - \mathbf{X}_c) - \theta) \\ \tilde{r}(\mathbf{X}^p - \mathbf{X}_c)\sin(\tilde{\theta}(\mathbf{X}^p - \mathbf{X}_c) - \theta) \end{pmatrix} + \mathbf{X}_c, \tag{47}$$

where

$$\tilde{r}(\mathbf{X}^p) = \sqrt{\mathbf{X}^{p^T}\mathbf{X}^p} \tag{48}$$

$$\tilde{\theta}(\mathbf{X}^p) = \text{arctan2}(x^p, y^p), \tag{49}$$

with $\mathbf{X}^p = [x^p, y^p]^T$ a column vector of each of the nodal positions located at the interface. The arctan2-function is an extension of the arctan-function and is defined as,

$$(x, y) \mapsto \begin{cases} \arctan(y/x) & x > 0, \\ \arctan(y/x) + \pi & x < 0, y \geq 0, \\ \arctan(y/x) - \pi & x < 0, y < 0, \\ \pi/2 & x = 0, y > 0, \\ -\pi/2 & x = 0, y < 0, \\ \text{undefined} & x = 0, y = 0. \end{cases} \tag{50}$$

The Poisson's ratio ξ for the auxiliary problem is first chosen to be 0.3. In Section 4.1.4 below, its influence on the accuracy of obtained results is discussed.

Table 1
Example 1–6 parameters with corresponding ranges. The parameters a, b and θ are the geometrical parameters describing the interface while $\bar{U}_{xx}$, $\bar{U}_{yy}$ and $\bar{U}_{xy}$ are external loading parameters.

a	b	θ	$\bar{U}_{xx}$	$\bar{U}_{yy}$	$\bar{U}_{xy}$
[0.1, 0.35]	[0.1, 0.35]	$[-\pi/2, \pi/2]$	[0.85, 1.15]	[0.85, 1.15]	[−0.15, 0.15]

(a) Weighted stress　　　　　　　　　　(b) Transformation displacement
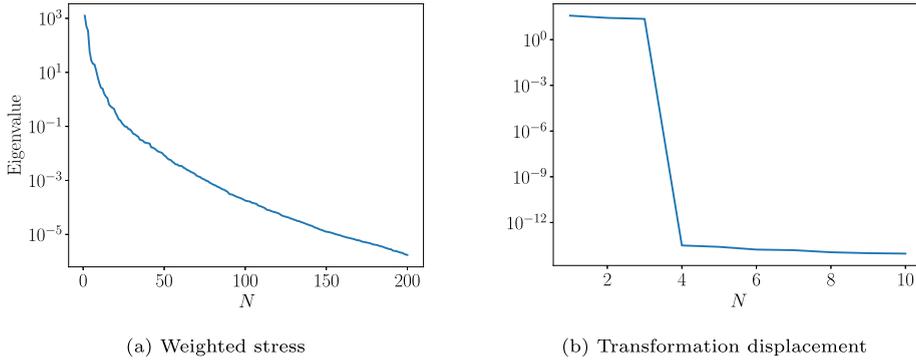
Fig. 4. Eigenvalues of the correlation matrix for (a) weighted stress and (b) transformation displacement. For the weighted stress the eigenvalues decay exponentially, while for the transformation displacement only the first three modes are nonzero. This means that the transformation displacement can be represented with just three basis functions.

4.1.2. Data generation

In total 1000 training snapshots are generated, of which $2^6 = 64$ snapshots are at the corners of the 6-dimensional parameter space and the rest are sampled from a Sobol sequence [61]. For testing, another 500 snapshots are generated from a random uniform distribution.

4.1.3. Results

POD of weighted stress and transformation displacement. The eigenvalues of the correlation matrix for the weighted stress are given in Fig. 4(a). It can be observed that the eigenvalues decay exponentially, indicating a good reducibility. To show that the auxiliary problem in Eq. (19) can be reduced drastically, the eigenvalues of the correlation matrix for the transformation displacement are also shown in Fig. 4(b). All but three eigenvalues are essentially zero, showing that the auxiliary problem can be solved with three basis functions.

Approximation errors. The average approximation error of the effective stress on the 500 testing snapshots of PODGPR for different numbers of basis functions N and training snapshots N_s is shown in Fig. 5. All N_s training snapshots are used for both the POD and GPR. In Fig. 5(a), the error decays rapidly for the first few basis functions. For $N = 20$ an error of roughly 0.5% is reached. However, taking into account more basis functions barely improves the performance since the coefficients get increasingly more oscillatory with increasing number and hence more difficult to approximate with a GPR model, see [31]. From Fig. 5(b), we see that a higher number of snapshots is crucial for the accuracy of PODGPR. Data shown corresponds to $N = 20$ basis functions. For small datasets the error increases exponentially, indicating a poor approximation of the first 20 POD coefficients.

In Table 2, the approximation quality of PODGPR with $N = 20$ basis functions is compared with the four neural networks. The best approximation error (defined as the error of projecting the truth solution onto the reduced basis) with $N = 20$ basis functions is also given. PODGPR approximates the effective stress better than all the NN models at least by a factor of 2. Moreover, it is nearly able to reach the error of the best approximation in both error measures, indicating that with $N_s = 1000$ training snapshots the first 20 POD coefficients can be well captured. The error in the stress field is less than 0.1%, while for the effective stress the error is 0.5%. Furthermore, by comparing the results with Fig. 5(b), it can be seen that the best results obtained by the neural networks are reached by PODGPR with only $N_s = 600$ snapshots, showing that PODGPR is more data efficient than the neural networks in this case.

(a)

(b)
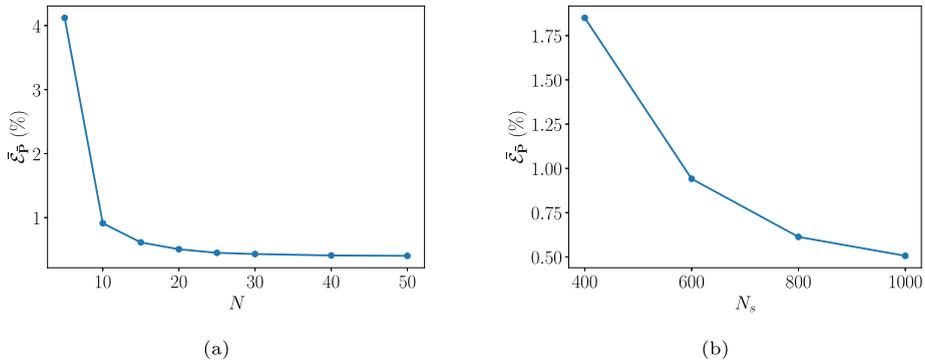
Fig. 5. (a) Average error of the effective stress over number of basis functions with $N_s = 1000$ training snapshots. The error curve decays rapidly for the first few basis functions and then flattens out. (b) Average error of the effective stress over number of training snapshots used for $N = 20$ basis functions. The error increases drastically when fewer training snapshots are used.

(a) $\xi = 0.0$ (b) $\xi = 0.49$

(c) Error for different Poisson's ratios
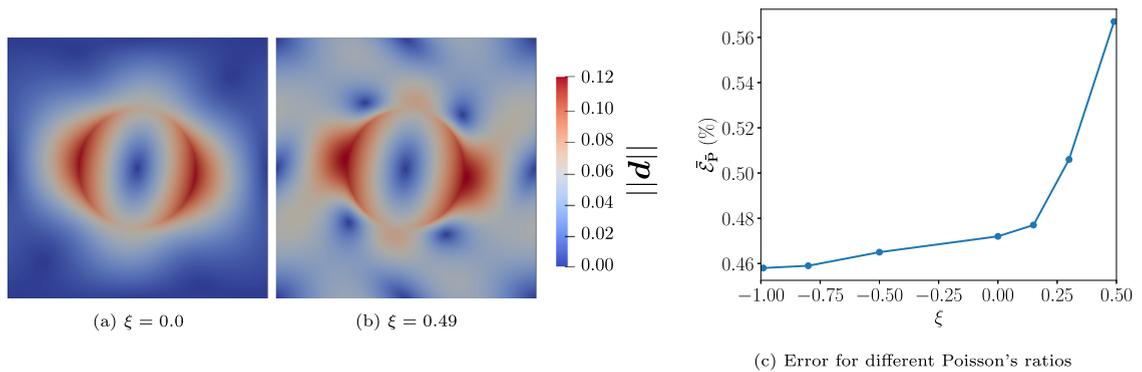
Fig. 6. Norm of transformation displacement with $a = 0.104$, $b = 0.291$, $\theta = -8.44°$ for (a) $\xi = 0.0$ and (b) $\xi = 0.49$. While the former leads to localized deformations, the latter affects the entire domain. (c) The mean error plotted over the Poisson's ratio ξ. The errors decrease with decreasing ξ, but remain on the same order of magnitude $\mathcal{O}(10^{-3})$.

Table 2
Approximation errors for different methods. The trained surrogate models are tested on 500 testing snapshots. PODGPR outperforms all NN models by a factor 2.

	Best approximation	PODGPR	NN1	NN2	NN3	NN4
$\bar{\mathcal{E}}_{\boldsymbol{P}}$	8.17×10^{-4}	9.49×10^{-4}	n.a.	n.a.	n.a.	n.a.
$\bar{\mathcal{E}}_{\bar{\mathbf{P}}}$	2.74×10^{-3}	5.06×10^{-3}	2.66×10^{-2}	1.53×10^{-2}	1.02×10^{-2}	9.8×10^{-3}

4.1.4. Influence of Poisson's ratio on the auxiliary problem

In this section, the influence of the Poisson's ratio on the approximation of the effective stress is investigated, considering $\xi \in \{-0.99, -0.8, -0.5, 0.0, 0.15, 0.3, 0.49\}$. Example transformation maps for $\xi \in \{0.0, 0.49\}$ are shown in Figs. 6(a) and 6(b) and the obtained errors are plotted over the Poisson's ratio in Fig. 6(d). For this example, the lower the Poisson's ratio the better the approximation. Nevertheless, all error values are close to each other (ranging from 0.46% to 0.58%), although the transformation displacement field differs significantly, see Figs. 6(a) and 6(b). From this empirical result, it seems that the choice of the Poisson's ratio is insignificant and, since there is no practical way of finding the best value, $\xi = 0.3$ is adopted hereafter.

4.2. Composite microstructure with a B-spline controlled inclusion shape

In the second example, an inclusion with a shape that is described by a B-spline with eight control points is considered, see Fig. 7(a). The x-coordinate of the left and right control point and the y-coordinate of the top and

Table 3
Example 2–7 parameters with corresponding ranges. The parameters a, b, c and d are geometrical parameters describing the interface, see Fig. 7 for the explanation, while $\bar{U}_{xx}$, $\bar{U}_{yy}$ and $\bar{U}_{xy}$ are external loading parameters.

a	b	c	d	$\bar{U}_{xx}$	$\bar{U}_{yy}$	$\bar{U}_{xy}$
[0.1, 0.4]	[0.1, 0.4]	[0.6, 0.9]	[0.6, 0.9]	[0.85, 1.15]	[0.85, 1.15]	[−0.15, 0.15]
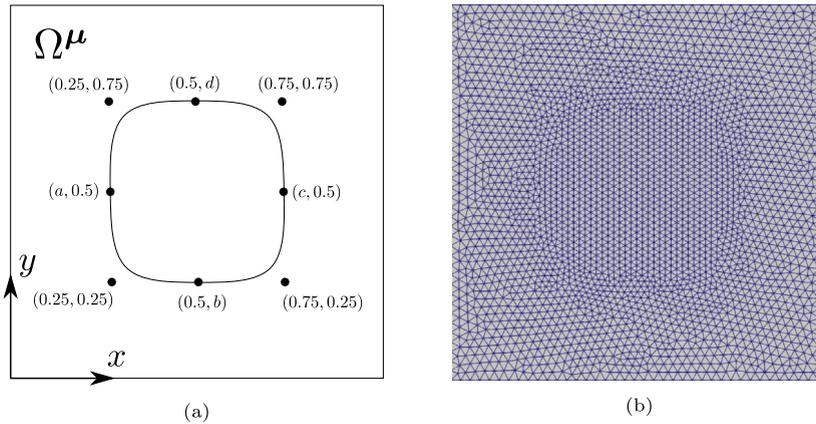
Fig. 7. Parameterized domain. (a) The interface is spanned by eight control points. Out of those, four control points can move in one direction, which are controlled by the geometrical parameters a, b, c and d. (b) The parent geometry with $a = b = 0.25$ and $c = d = 0.75$ is chosen and the mesh consists of 11296 nodes and 5833 6-node triangular elements.

(a) $a = b = 0.1$, $c = d = 0.9$ (b) $a = 0.4$, $b = 0.1$, $c = 0.6$, $d = 0.9$ (c) $a = b = 0.4$, $c = d = 0.6$
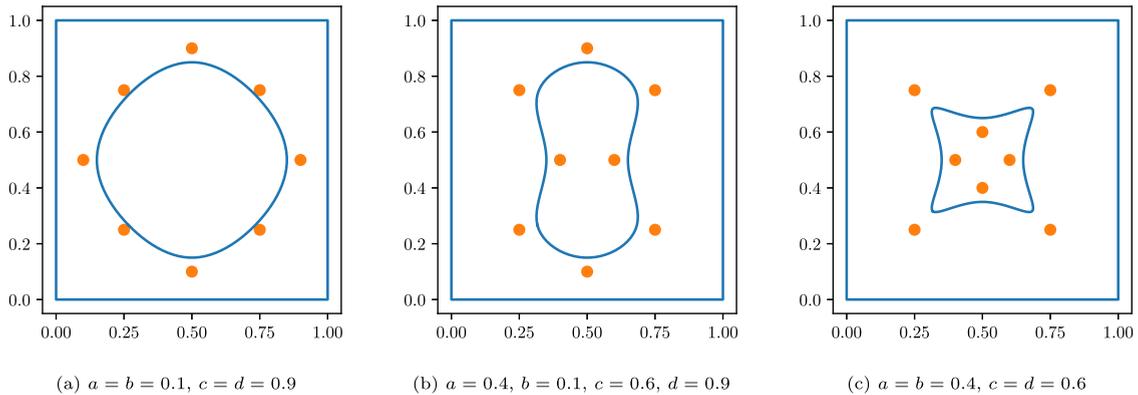
Fig. 8. Example geometries. The control points are shown in orange color and the resulting interface in blue color. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

bottom control point are parameterized, resulting in four geometrical parameters a, b, c and d. The curve is then interpolated with cubic polynomials using the Python library NURBS-Python [62]. The same material parameters are chosen as in the first example, with parameter ranges given in Table 3. A few example geometries are shown in Fig. 8 to show the variety of shapes covered by this parameterization. The lower and upper bounds for $\bar{U}$ are chosen in the same way as in Section 4.1.

4.2.1. Setup of the auxiliary problem

For the parent domain, the midpoint of the parameter domain is selected, i.e., $a = b = 0.25$ and $c = d = 0.75$. The corresponding geometry and mesh, consisting of 11296 nodes and 5833 6-node triangular elements, are shown in Fig. 7.

(a) Weighted stress   (b) Transformation displacement
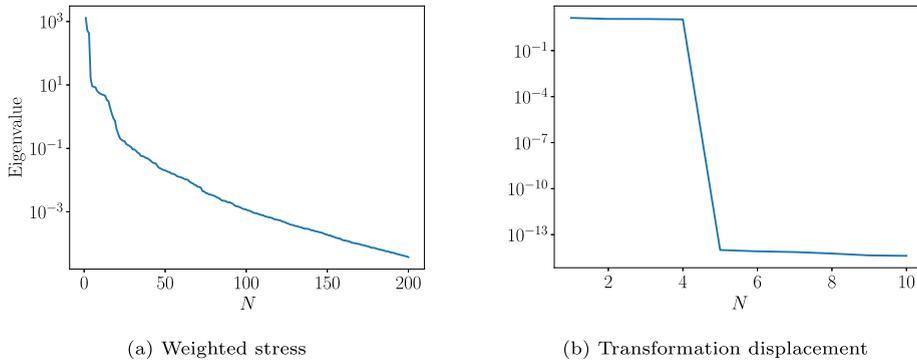
Fig. 9. Eigenvalues of the correlation matrix for (a) weighted stress and (b) transformation displacement. Same as for the last example, the eigenvalues of weighted stress decay exponentially. For the transformation displacement only the first 4 modes are nonzero, meaning it can be represented with 4 basis functions.

Table 4
Mean errors for different methods. The trained surrogate models are tested on 500 testing snapshots. PODGPR outperforms all NN models by a factor 2.

	Best approximation	PODGPR	NN1	NN2	NN3	NN4
$\bar{\mathcal{E}}_{\boldsymbol{P}}$	1.79×10^{-4}	1.85×10^{-4}	n.a.	n.a.	n.a.	n.a.
$\bar{\mathcal{E}}_{\bar{\boldsymbol{P}}}$	1.09×10^{-3}	1.31×10^{-3}	3.3×10^{-3}	2.5×10^{-3}	2.2×10^{-3}	2.3×10^{-3}

4.2.2. Data generation

In total 1000 training snapshots are again generated from a Sobol sequence [61], while another 500 snapshots are generated from a random uniform distribution for testing.

4.2.3. Results

POD of weighted stress and transformation displacement. In Fig. 9, the eigenvalues of the correlation matrix for both weighted stress and transformation displacement are depicted. An exponential decay can be observed for the weighted stress, while all but four eigenvalues are essentially zero for the transformation displacement.

Approximation errors. The average approximation errors on the 500 testing snapshots of the best approximation (projection of truth solution onto the reduced basis) with $N = 50$ basis functions, PODGPR with $N = 50$ basis functions and the four neural networks are reported in Table 4. From the results it can be observed that PODGPR nearly reaches the best approximation, showing that the first 50 coefficients are well approximated by the GPR models. Furthermore, it outperforms all neural network architectures by a factor of 2, reaching an average error of 0.131% in effective stress.

4.3. Two-scale Cook's membrane problem

While the last two examples dealt with the construction of the surrogate model for the microscale, in this example the surrogate model is employed in a full two-scale Cook's membrane problem. Here, the microstructure from Section 4.1 with an elliptical inclusion is considered. The geometry of the Cook's membrane and its mesh are shown in Fig. 10. The mesh consists of 200 quadrilateral elements with 4 quadrature points, resulting in 800 microstructure evaluations required for a single Newton iteration. The microstructural parameters a and b are assumed to be constant with $a = 0.35$ and $b = 0.1$, corresponding to an ellipse, while the angle θ is a function in the x-coordinate with $\theta(x) = (\pi \sin x)/2$, in order to test the performance of the surrogate model under rapidly varying fiber directions (see Fig. 10(c) for an illustration of the function). A similar test problem was considered in [63]. The left side of the membrane is fixed, while a vertical traction of 0.05 is applied on the right edge, which leads to overall deformations within the training range of the surrogate model ($|\bar{U}_{xx} - 1|, |\bar{U}_{yy} - 1|, |\bar{U}_{xy}| < 0.15$).

A full FE^2 simulation and a FE simulation using PODGPR are run and the obtained displacement fields $\bar{\boldsymbol{u}}$ are compared in Fig. 11. The PODGPR surrogate model constructed with $N = 20$ basis functions from $N_s = 1000$
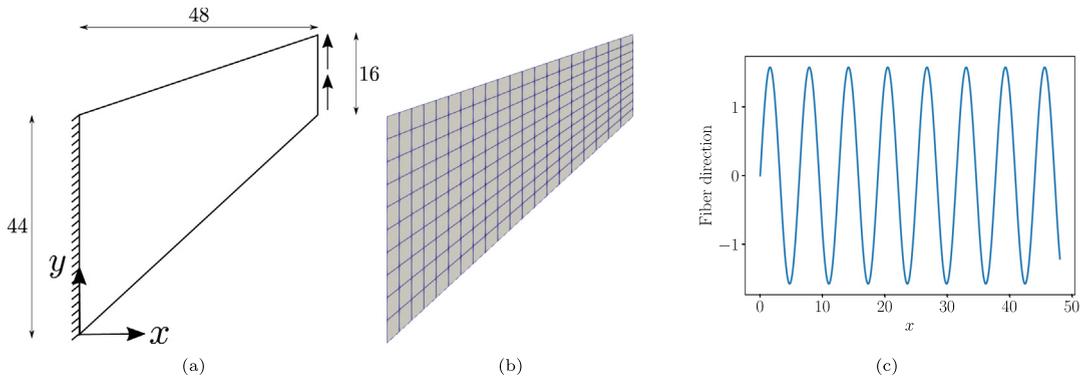
Fig. 10. (a) The geometry of the Cook's membrane and (b) corresponding simulation mesh used. The mesh consists of 231 nodes and 200 4-node quadrilateral elements. (c) The fiber direction depends on the x-coordinate as $\theta(x) = \dfrac{\pi}{2} \sin x$.
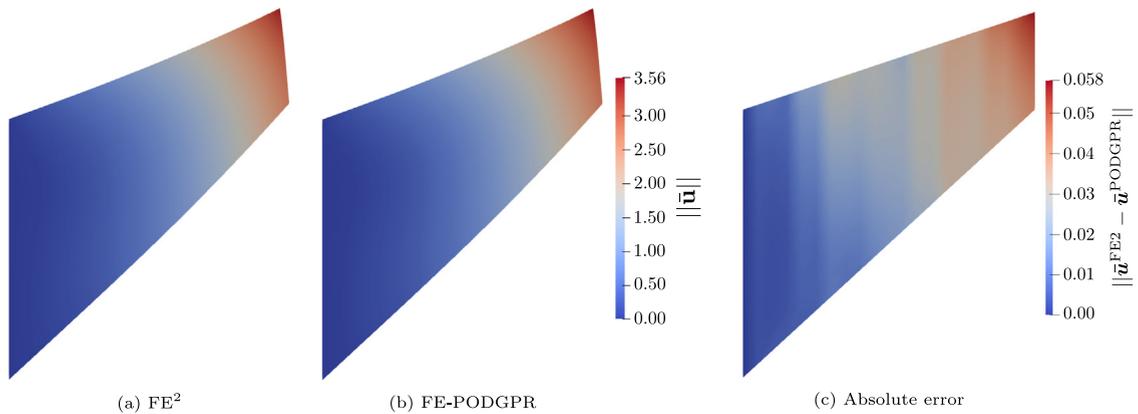
Fig. 11. The norm of the displacement obtained for (a) FE^2 and (b) FE with PODGPR. The displacement fields are nearly identical. (c) The magnitude of the absolute error between both solutions. The error is around two orders of magnitude lower than the displacement values and increases from left to right. Vertical stripes are visible, corresponding to the rapidly varying fiber orientations, showing that some angles are better or worse approximated.

training snapshots is used. The obtained displacement fields shown in Figs. 11(a) and 11(b) are almost identical. To further quantify the error, the magnitude of the difference between both displacement solutions is shown in Fig. 11(d). Here we observe that the highest absolute error is at the top right corner and the error increases from left to right. Comparing the error and the actual value at the top right corner, the relative error corresponds to $0.058/3.56 = 1.6\%$. Furthermore, vertical stripes with similar errors can be seen, which correspond to the quickly varying fiber directions in the x-coordinate, meaning that some angles are better or worse approximated. The quality of approximation for different angles depends on the sampling of the training snapshots. Moreover, the compliance f_c, defined as $f_c := \mathbf{f}_{\text{ext}}^T \bar{\mathbf{u}}$ where $\mathbf{f}_{\text{ext}}$ corresponds to the externally applied vertical traction, is computed for both methods, yielding $f_c^{\text{FE2}} = 2.227$ and $f_c^{\text{PODGPR}} = 2.202$, resulting in a relative error of 1.1%. This is an important quantity, often used in optimization problems.

The execution times[2] for both cases are reported in Table 5. For the construction of PODGPR, 1000 snapshots are generated, involving each time an auxiliary problem and a full simulation to be solved, taking roughly 4 h. With the data available, the construction of PODGPR with 20 basis functions takes around 10 min. After this offline computation, the online speed up is on the order of 1000 as compared to the full two-scale simulation.

---

[2] All operations were executed using four subprocesses on an Intel Core i7-8750H.

Table 5

Run times. The offline stage for constructing PODGPR takes slightly over 4 h. Then, the Cook's membrane problem can be solved within 0.5 min, while at the same time the full FE^2 simulation takes around 1080 min for a single forward solution.

	FE^2	FE-PODGPR
Offline	n.a	Auxiliary Problem: ≈ 10 min
		Snapshot Generation: ≈ 240 min
		PODGPR: ≈ 10 min
Online	≈ 1080 min	≈ 0.5 min

5. Conclusions

In order to successfully find optimal microstructures for complex engineering systems in a reasonable amount of time, it is necessary to find an accurate and fast-to-evaluate approximation of the microscopic simulation, which can handle a large design space of shapes and geometrical variations. In this work, we proposed a PDE-based transformation method for the treatment of geometrical parameters. By combining the transformation with the proper orthogonal decomposition and Gaussian process regression, we have developed a non-intrusive effective constitutive model that makes predictions which automatically fulfill the underlying microscopic governing equations for a wide variety of geometries. For the two considered microstructures, each described by several geometrical parameters, the surrogate model captures the local stress fields accurately with an average error less than 1%. The effective model is embedded in a two-scale problem, where a high variation in parameters throughout the domain is considered, and accelerates the simulation by a factor of 1000 as compared to the full FE^2 simulation, while maintaining high accuracy. Sensitivities with respect to microstructural parameters are available, which can be utilized in two-scale shape optimization problems or solution of inverse problems, and the methodology can also be directly applied to 3D problems.

Although this method is powerful, several limitations exist. For geometrical parameters leading to very severe geometrical variations, the proposed transformation leads to highly irregular and distorted meshes. A possible remedy to resolve this issue is to correct the distorted elements by mesh refinement, or to use multiple parent geometries with local surrogate models. In the online stage, the closest cluster could be chosen to evaluate the surrogate model. Another challenge is the requirement of data. Even though the methodology proved to be more data efficient than feed-forward neural networks in our examples, still a rather large amount of training data is needed. Possible solutions are multi-fidelity methods or adaptive sampling schemes.

Declaration of competing interest

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

Data availability

Data will be made available on request.

Acknowledgments

Appendix A. Proof of Eq. (38)

With $\boldsymbol{B}_n$ the nth basis function of the weighted stress, cf. Eq. (27), we want to prove that

$$\int_{\Omega^p} \boldsymbol{B}_n(\boldsymbol{X}^p)\boldsymbol{F}_{\boldsymbol{\mu}}^T d\boldsymbol{X}^p \overset{!}{=} \int_{\Omega^p} \boldsymbol{B}_n(\boldsymbol{X}^p)d\boldsymbol{X}^p \tag{A.1}$$

for all $\boldsymbol{\mu}$ and all n. First of all, $\boldsymbol{F}_\mu$ can be written as

$$\boldsymbol{F}_\mu = \mathbf{I} + \frac{\partial \boldsymbol{d}_\mu}{\partial \boldsymbol{X}^p} \tag{A.2}$$

with $\boldsymbol{d}_\mu$ the transformation displacement, so the left-hand side of Eq. (A.1) splits into

$$\int_{\Omega^p} \boldsymbol{B}_n(\boldsymbol{X}^p) \boldsymbol{F}_\mu^T d\boldsymbol{X}^p = \int_{\Omega^p} \boldsymbol{B}_n(\boldsymbol{X}^p) d\boldsymbol{X}^p + \int_{\Omega^p} \boldsymbol{B}_n(\boldsymbol{X}^p) \left(\frac{\partial \boldsymbol{d}_\mu}{\partial \boldsymbol{X}^p} \right)^T d\boldsymbol{X}^p. \tag{A.3}$$

To prove Eq. (A.1), we thus need to show that

$$\int_{\Omega^p} \boldsymbol{B}_n(\boldsymbol{X}^p) \left(\frac{\partial \boldsymbol{d}_\mu}{\partial \boldsymbol{X}^p} \right)^T d\boldsymbol{X}^p \overset{!}{=} \boldsymbol{0}. \tag{A.4}$$

Without loss of generality, assume only a single training snapshot, obtained for parameters $(\bar{\mathbf{U}}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ on a domain Ω^{μ^*}, and a transformation map $\boldsymbol{\Phi}_{\mu^*} : \Omega^p \to \Omega^{\mu^*}$, $\boldsymbol{X}^{\mu^*} = \boldsymbol{\Phi}_{\mu^*}(\boldsymbol{X}^p)$. Thus, there is only a single basis function,

$$\boldsymbol{B}_1(\boldsymbol{X}^p) = \boldsymbol{P}(\boldsymbol{\Phi}_{\mu^*}(\boldsymbol{X}^p); \bar{\mathbf{U}}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \boldsymbol{F}_{\mu^*}^{-T} \left| \det \boldsymbol{F}_{\mu^*} \right|, \tag{A.5}$$

and Eq. (A.4) becomes

$$\int_{\Omega^p} \boldsymbol{P}(\boldsymbol{\Phi}_{\mu^*}(\boldsymbol{X}^p); \bar{\mathbf{U}}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \boldsymbol{F}_{\mu^*}^{-T} \left| \det \boldsymbol{F}_{\mu^*} \right| \left(\frac{\partial \boldsymbol{d}_\mu}{\partial \boldsymbol{X}^p} \right)^T d\boldsymbol{X}^p. \tag{A.6}$$

We would now like to push the integral in Eq. (A.6) forward onto the domain Ω^{μ^*}. By introducing the inverse mapping,

$$\boldsymbol{\Phi}_{\mu^*}^{-1} : \Omega^{\mu^*} \to \Omega^p, \quad \boldsymbol{X}^p = \boldsymbol{\Phi}_{\mu^*}^{-1}(\boldsymbol{X}^{\mu^*}), \quad d\boldsymbol{X}^{\mu^*} = \left| \det \boldsymbol{F}_{\mu^*} \right| d\boldsymbol{X}^p, \tag{A.7}$$

$\boldsymbol{F}_{\mu^*}$ is transformed with

$$\boldsymbol{F}_{\mu^*} = \frac{\partial \boldsymbol{\Phi}_{\mu^*}(\boldsymbol{X}^p)}{\partial \boldsymbol{X}^p} = \frac{\partial \boldsymbol{\Phi}_{\mu^*}(\boldsymbol{\Phi}_{\mu^*}^{-1}(\boldsymbol{X}^{\mu^*}))}{\partial \boldsymbol{X}^{\mu^*}} \left(\frac{\partial \boldsymbol{X}^p}{\partial \boldsymbol{X}^{\mu^*}} \right)^{-1} = \left(\frac{\partial \boldsymbol{X}^p}{\partial \boldsymbol{X}^{\mu^*}} \right)^{-1}, \tag{A.8}$$

from which it follows that

$$\boldsymbol{F}_{\mu^*}^{-T} = \left(\frac{\partial \boldsymbol{X}^p}{\partial \boldsymbol{X}^{\mu^*}} \right)^T; \tag{A.9}$$

furthermore, the partial derivative $\dfrac{\partial \boldsymbol{d}_\mu(\boldsymbol{X}^p)}{\partial \boldsymbol{X}^p}$ can be expressed as

$$\frac{\partial \boldsymbol{d}_\mu(\boldsymbol{X}^p)}{\partial \boldsymbol{X}^p} = \frac{\partial \boldsymbol{d}_\mu(\boldsymbol{\Phi}_{\mu^*}^{-1}(\boldsymbol{X}^{\mu^*}))}{\partial \boldsymbol{X}^{\mu^*}} \left(\frac{\partial \boldsymbol{X}^p}{\partial \boldsymbol{X}^{\mu^*}} \right)^{-1}, \tag{A.10}$$

or

$$\left(\frac{\partial \boldsymbol{d}_\mu}{\partial \boldsymbol{X}^p} \right)^T = \left(\frac{\partial \boldsymbol{X}^p}{\partial \boldsymbol{X}^{\mu^*}} \right)^{-T} \left(\frac{\partial \boldsymbol{d}_\mu(\boldsymbol{\Phi}_{\mu^*}^{-1}(\boldsymbol{X}^{\mu^*}))}{\partial \boldsymbol{X}^{\mu^*}} \right)^T \tag{A.11}$$

by transposing both sides of Eq. (A.10). With Eqs. (A.7), (A.11), (A.9) and (A.6) becomes

$$\int_{\Omega^{\mu^*}} \boldsymbol{P}(\boldsymbol{X}^{\mu^*}; \bar{\mathbf{U}}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \left(\frac{\partial \boldsymbol{X}^p}{\partial \boldsymbol{X}^{\mu^*}} \right)^T \left(\frac{\partial \boldsymbol{X}^p}{\partial \boldsymbol{X}^{\mu^*}} \right)^{-T} \left(\frac{\partial \boldsymbol{d}_\mu(\boldsymbol{\Phi}_{\mu^*}^{-1}(\boldsymbol{X}^{\mu^*}))}{\partial \boldsymbol{X}^{\mu^*}} \right)^T d\boldsymbol{X}^{\mu^*} \tag{A.12}$$

$$= \int_{\Omega^{\mu^*}} \boldsymbol{P}(\boldsymbol{X}^{\mu^*}; \bar{\mathbf{U}}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \left(\frac{\partial \boldsymbol{d}_\mu(\boldsymbol{\Phi}_{\mu^*}^{-1}(\boldsymbol{X}^{\mu^*}))}{\partial \boldsymbol{X}^{\mu^*}} \right)^T d\boldsymbol{X}^{\mu^*}. \tag{A.13}$$

Utilizing the divergence theorem, Eq. (A.13) can be rewritten as

$$\int_{\partial \Omega^{\mu^*}} \boldsymbol{t} \otimes \boldsymbol{d}_\mu(\boldsymbol{\Phi}_{\mu^*}^{-1}(\boldsymbol{X}^{\mu^*})) \, ds - \int_{\Omega^{\mu^*}} \boldsymbol{d}_\mu(\boldsymbol{\Phi}_{\mu^*}^{-1}(\boldsymbol{X}^{\mu^*})) \otimes \mathrm{Div}\, \boldsymbol{P}(\boldsymbol{X}^{\mu^*}; \bar{\mathbf{U}}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) d\boldsymbol{X}^{\mu^*}, \tag{A.14}$$

where $\boldsymbol{t} := \boldsymbol{P}(X^{\boldsymbol{\mu}^*}; \bar{\mathbf{U}}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)\boldsymbol{n}$ is the traction vector with $\boldsymbol{n}$ the outer unit normal along the boundary $\partial \Omega^{\boldsymbol{\mu}^*}$ and ds is an infinitesimal boundary element. Using the fact that the training snapshot fulfills the linear momentum balance Div $\boldsymbol{P}(X^{\boldsymbol{\mu}^*}; \bar{\mathbf{U}}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) = \boldsymbol{0}$ on the domain $\Omega^{\boldsymbol{\mu}^*}$, the latter part of Eq. (A.14) becomes $\boldsymbol{0}$, and therefore,

$$\int_{\Omega^{\boldsymbol{\mu}^*}} \boldsymbol{P}(X^{\boldsymbol{\mu}^*}; \bar{\mathbf{U}}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \left(\frac{\partial \boldsymbol{d}_{\boldsymbol{\mu}}(\boldsymbol{\Phi}_{\boldsymbol{\mu}^*}^{-1}(X^{\boldsymbol{\mu}^*}))}{\partial X^{\boldsymbol{\mu}^*}} \right)^T dX^{\boldsymbol{\mu}^*} = \int_{\partial \Omega^{\boldsymbol{\mu}^*}} \boldsymbol{t} \otimes \boldsymbol{d}_{\boldsymbol{\mu}}(\boldsymbol{\Phi}_{\boldsymbol{\mu}^*}^{-1}(X^{\boldsymbol{\mu}^*}))\ ds. \tag{A.15}$$

Due to the definition of the auxiliary problem in Eq. (18), $\boldsymbol{d}_{\boldsymbol{\mu}}(\boldsymbol{\Phi}_{\boldsymbol{\mu}^*}^{-1}(X^{\boldsymbol{\mu}^*}))$ is $\boldsymbol{0}$ on the boundary $\partial \Omega^{\boldsymbol{\mu}^*}$ and therefore the boundary integral in Eq. (A.15) always results in the zero tensor, meaning that the integral in Eq. (A.6) vanishes for all $\boldsymbol{\mu}$.

Since each basis function $\boldsymbol{B}_n$ is a linear combination of converged weighted stress fields, from the linearity, the integral on the left hand side of Eq. (A.4) vanishes as well, which is what we wanted to prove and Eq. (A.4) holds.

References

[1] M.G.D. Geers, V.G. Kouznetsova, W.A.M. Brekelmans, Multi-scale computational homogenization: Trends and challenges, J. Comput. Appl. Math. 234 (7) (2010) 2175–2182, http://dx.doi.org/10.1016/j.cam.2009.08.077, URL https://linkinghub.elsevier.com/retrieve/pii/S0377042709005536.

[2] K. Matouš, M.G.D. Geers, V.G. Kouznetsova, A. Gillman, A review of predictive nonlinear theories for multiscale modeling of heterogeneous materials, J. Comput. Phys. (2017) http://dx.doi.org/10.1016/j.jcp.2016.10.070.

[3] F. Feyel, Multiscale FE2 elastoviscoplastic analysis of composite structures, Comput. Mater. Sci. 16 (1–4) (1999) 344–354, http://dx.doi.org/10.1016/s0927-0256(99)00077-4.

[4] C. Miehe, A. Koch, Computational micro-to-macro transitions of discretized microstructures undergoing small strains, Arch. Appl. Mech. 72 (4–5) (2002) 300–317, http://dx.doi.org/10.1007/s00419-002-0212-2.

[5] V.G. Kouznetsova, Computational Homogenization for the Multi-Scale Analysis of Multi-Phase Materials, (2002) 2002, pp. 1–134, http://dx.doi.org/10.6100/IR560009, URL http://www.narcis.nl/publication/RecordID/oai:library.tue.nl:560009.

[6] H. Moulinec, P. Suquet, A numerical method for computing the overall response of nonlinear composites with complex microstructure, Comput. Methods Appl. Mech. Engrg. 157 (1–2) (1998) 69–94, http://dx.doi.org/10.1016/S0045-7825(97)00218-1.

[7] N. Mishra, J. Vondřejc, J. Zeman, A comparative study on low-memory iterative solvers for FFT-based homogenization of periodic media, J. Comput. Phys. (2016) http://dx.doi.org/10.1016/j.jcp.2016.05.041.

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

[9] T. Kirchdoerfer, M. Ortiz, Data driven computing with noisy material data sets, Comput. Methods Appl. Mech. Engrg. 326 (2017) 622–641, http://dx.doi.org/10.1016/j.cma.2017.07.039.

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

[11] 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. Engrg. 369 (2020) 113234, http://dx.doi.org/10.1016/j.cma.2020.113234.

[12] D.W. Abueidda, S. Koric, N.A. Sobh, H. Sehitoglu, Deep learning for plasticity and thermo-viscoplasticity, Int. J. Plast. 136 (August 2020) (2021) 102852, http://dx.doi.org/10.1016/j.ijplas.2020.102852.

[13] 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, http://dx.doi.org/10.1073/pnas.1911815116.

[14] 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, http://dx.doi.org/10.1002/nme.4953.

[15] S. Kumar, S. Tan, L. Zheng, D.M. Kochmann, Inverse-designed spinodoid metamaterials, NPJ Comput. Mater. 6 (1) (2020) 1–10, http://dx.doi.org/10.1038/s41524-020-0341-6.

[16] M. Fernández, M. Jamshidian, T. Böhlke, K. Kersting, O. Weeger, Anisotropic hyperelastic constitutive models for finite deformations combining material theory and data-driven approaches with application to cubic lattice metamaterials, Comput. Mech. (2021) http://dx.doi.org/10.1007/s00466-020-01954-7.

[17] K. Linka, M. Hillgärtner, K.P. Abdolazizi, R.C. Aydin, M. Itskov, C.J. Cyron, Constitutive artificial neural networks: A fast and general approach to predictive data-driven constitutive modeling by deep learning, J. Comput. Phys. 429 (2021) 110010, http://dx.doi.org/10.1016/j.jcp.2020.110010.

[18] M. Flaschel, S. Kumar, L. De Lorenzis, Unsupervised discovery of interpretable hyperelastic constitutive laws, Comput. Methods Appl. Mech. Engrg. 381 (2018) (2021) 113852, http://dx.doi.org/10.1016/j.cma.2021.113852, URL http://arxiv.org/abs/2010.13496. https://linkinghub.elsevier.com/retrieve/pii/S0045782521001894.

[19] G.J. Dvorak, Transformation field analysis of inelastic composite materials, Proc. R. Soc. A 437 (1900) (1992) 311–327, http://dx.doi.org/10.1098/rspa.1992.0063.

[20] J.C. Michel, P. Suquet, Nonuniform transformation field analysis, Int. J. Solids Struct. 40 (25) (2003) 6937–6955, http://dx.doi.org/10.1016/S0020-7683(03)00346-9, URL https://linkinghub.elsevier.com/retrieve/pii/S0020768303003469.

[21] 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, http://dx.doi.org/10.1016/j.cma.2016.04.004.

[22] A. Quarteroni, A. Manzoni, F. Negri, Reduced Basis Methods for Partial Differential Equations, UNITEXT, vol. 92, Springer International Publishing, Cham, 2016, pp. 1–263, http://dx.doi.org/10.1007/978-3-319-15431-2.

[23] J.S. Hesthaven, G. Rozza, B. Stamm, Certified Reduced Basis Methods for Parametrized Partial Differential Equations, SpringerBriefs in Mathematics, Springer International Publishing, Cham, 2016, pp. 1–131, http://dx.doi.org/10.1007/978-3-319-22470-1.

[24] J.A. Hernández, J. Oliver, A.E. Huespe, M.A. Caicedo, J.C. Cante, High-performance model reduction techniques in computational multiscale homogenization, Comput. Methods Appl. Mech. Engrg. 276 (2014) 149–189, http://dx.doi.org/10.1016/j.cma.2014.03.011.

[25] D. Soldner, B. Brands, R. Zabihyan, P. Steinmann, J. Mergheim, A numerical study of different projection-based model reduction techniques applied to computational homogenisation, Comput. Mech. 60 (4) (2017) 613–625, http://dx.doi.org/10.1007/s00466-017-1428-x.

[26] D. Ryckelynck, Hyper-reduction of mechanical models involving internal variables, Internat. J. Numer. Methods Engrg. 77 (1) (2009) 75–89, http://dx.doi.org/10.1002/nme.2406.

[27] A. Radermacher, S. Reese, A comparison of projection-based model reduction concepts in the context of nonlinear biomechanics, Arch. Appl. Mech. 83 (8) (2013) 1193–1213, http://dx.doi.org/10.1007/s00419-013-0742-9.

[28] M. Guo, J.S. Hesthaven, Reduced order modeling for nonlinear structural analysis using Gaussian process regression, Comput. Methods Appl. Mech. Engrg. 341 (2018) 807–826, http://dx.doi.org/10.1016/j.cma.2018.07.017.

[29] J.S. Hesthaven, S. Ubbiali, Non-intrusive reduced order modeling of nonlinear problems using neural networks, J. Comput. Phys. 363 (2018) 55–78, http://dx.doi.org/10.1016/j.jcp.2018.02.037.

[30] R. Swischuk, L. Mainini, B. Peherstorfer, K. Willcox, Projection-based model reduction: Formulations for physics-based machine learning, Comput. & Fluids 179 (2019) 704–717, http://dx.doi.org/10.1016/j.compfluid.2018.07.021.

[31] T. Guo, O. Rokoš, K. Veroy, Learning constitutive models from microstructural simulations via a non-intrusive reduced basis method, Comput. Methods Appl. Mech. Engrg. 384 (2021) 113924, http://dx.doi.org/10.1016/j.cma.2021.113924, URL http://arxiv.org/abs/2104.04451. https://linkinghub.elsevier.com/retrieve/pii/S0045782521002619.

[32] M. Barrault, Y. Maday, N.C. Nguyen, A.T. Patera, An 'empirical interpolation' method: Application to efficient reduced-basis discretization of partial differential equations, C. R. Math. 339 (9) (2004) 667–672, http://dx.doi.org/10.1016/j.crma.2004.08.006.

[33] S. Chaturantabut, D.C. Sorensen, Nonlinear model reduction via discrete empirical interpolation, SIAM J. Sci. Comput. 32 (5) (2010) 2737–2764, http://dx.doi.org/10.1137/090766498.

[34] C.E. Rasmussen, Gaussian processes in machine learning, in: O. Bousquet, U. von Luxburg, G. Rätsch (Eds.), Advanced Lectures on Machine Learning: ML Summer Schools 2003, Canberra, Australia, February 2 - 14, 2003, Tübingen, Germany, August 4 - 16, 2003, Revised Lectures, Springer Berlin Heidelberg, Berlin, Heidelberg, 2004, pp. 63–71, http://dx.doi.org/10.1007/978-3-540-28650-9_4.

[35] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, B. Levy, Polygon Mesh Processing, Vol. 44, (8) A K Peters/CRC Press, 2010, pp. 1689–1699, http://dx.doi.org/10.1201/b10688, URL https://www.taylorfrancis.com/books/9781439865316.

[36] T.W. Sederberg, S.R. Parry, Free-form deformation of solid geometric models, in: Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1986, vol. 20, (4) 1986, pp. 151–160, http://dx.doi.org/10.1145/15922.15903.

[37] M.D. Buhmann, Radial Basis Functions: Theory and Implementations, vol. 12, Cambridge University Press, 2003.

[38] G. Rozza, D.B.P. Huynh, A.T. Patera, Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations: Application to transport and continuum mechanics, Arch. Comput. Methods Eng. 15 (3) (2008) 229–275, http://dx.doi.org/10.1007/s11831-008-9019-9.

[39] K. Veroy, Reduced-basis methods applied to problems in elasticity : Analysis and applications (2000), 2003, URL https://dspace.mit.edu/handle/1721.1/29583.

[40] G. Rozza, M. Hess, G. Stabile, M. Tezzele, F. Ballarin, 1 basic ideas and tools for projection-based model reduction of parametric partial differential equations, in: Snapshot-Based Methods and Algorithms, De Gruyter, 2020, pp. 1–47, http://dx.doi.org/10.1515/9783110671490-001, URL https://www.degruyter.com/document/doi/10.1515/9783110671490-001/html.

[41] F. Negri, A. Manzoni, D. Amsallem, Efficient model reduction of parametrized systems by matrix discrete empirical interpolation, J. Comput. Phys. 303 (2015) 431–454, http://dx.doi.org/10.1016/j.jcp.2015.09.046.

[42] A. Manzoni, A. Quarteroni, G. Rozza, Model reduction techniques for fast blood flow simulation in parametrized geometries, Int. J. Numer. Methods Biomed. Eng. 28 (6–7) (2012) 604–625, http://dx.doi.org/10.1002/cnm.1465, URL https://onlinelibrary.wiley.com/doi/10.1002/cnm.1465.

[43] K. Washabaugh, M.J. Zahr, C. Farhat, On the use of discrete nonlinear reduced-order models for the prediction of steady-state flows past parametrically deformed complex geometries, in: 54th AIAA Aerospace Sciences Meeting, (January) 2016, pp. 1–16, http://dx.doi.org/10.2514/6.2016-1814.

[44] N. Demo, M. Tezzele, G. Gustin, G. Lavini, G. Rozza, Shape optimization by means of proper orthogonal decomposition and dynamic mode decomposition, in: Technology and Science for the Ships of the Future - Proceedings of NAV 2018: 19th International Conference on Ship and Maritime Research, 2018, pp. 212–219, http://dx.doi.org/10.3233/978-1-61499-870-9-212.

[45] G. Stabile, M. Zancanaro, G. Rozza, Efficient geometrical parametrization for finite-volume-based reduced order methods, Internat. J. Numer. Methods Engrg. 121 (12) (2020) 2655–2682, http://dx.doi.org/10.1002/nme.6324.

[46] T.-M. Yao, K.K. Choi, 3-D shape optimal design and automatic finite element regridding, Internat. J. Numer. Methods Engrg. 28 (2) (1989) 369–384, http://dx.doi.org/10.1002/nme.1620280209, URL https://onlinelibrary.wiley.com/doi/10.1002/nme.1620280209.

[47] H. Jasak, Z. Tuković, Automatic mesh motion for the unstructured Finite Volume Method, Trans. Famena 30 (2) (2006) 1–20.

[48] M. Itskov, Tensor Algebra and Tensor Analysis for Engineers, Springer, 2007.

[49] V.G. Kouznetsova, W.A.M. Brekelmans, F.P.T. Baaijens, Approach to micro–macro modeling of heterogeneous materials, Comput. Mech. 27 (1) (2001) 37–48, http://dx.doi.org/10.1007/s004660000212.

[50] S. Saeb, P. Steinmann, A. Javili, Aspects of computational homogenization at finite deformations: A unifying review from Reuss' to Voigt's Bound, Appl. Mech. Rev. 68 (5) (2016) http://dx.doi.org/10.1115/1.4034024.

[51] I. Temizer, P. Wriggers, On the computation of the macroscopic tangent for multiscale volumetric homogenization problems, Comput. Methods Appl. Mech. Engrg. 198 (3–4) (2008) 495–510, http://dx.doi.org/10.1016/j.cma.2008.08.018.

[52] N. Sarna, P. Benner, Data-driven model order reduction for problems with parameter-dependent jump-discontinuities, Comput. Methods Appl. Mech. Engrg. 387 (2021) 114168, http://dx.doi.org/10.1016/j.cma.2021.114168.

[53] M. Guénot, I. Lepot, C. Sainvitu, J. Goblet, R.F. Coelho, Adaptive sampling strategies for non-intrusive POD-based surrogates, Eng. Comput. 30 (4) (2013) 521–547, http://dx.doi.org/10.1108/02644401311329352.

[54] M. Kast, M. Guo, J.S. Hesthaven, A non-intrusive multifidelity method for the reduced order modeling of nonlinear problems, Comput. Methods Appl. Mech. Engrg. 364 (2020) 112947, http://dx.doi.org/10.1016/j.cma.2020.112947.

[55] A. Bērziņš, J. Helmig, F. Key, S. Elgeti, Standardized non-intrusive reduced order modeling using different regression models with application to complex flow problems, arXiv, 2020.

[56] M. Yang, Z. Xiao, POD-based surrogate modeling of transitional flows using an adaptive sampling in Gaussian process, Int. J. Heat Fluid Flow 84 (December 2019) (2020) 108596, http://dx.doi.org/10.1016/j.ijheatfluidflow.2020.108596.

[57] GPy, GPy: A Gaussian process framework in python, 2012, URL https://github.com/SheffieldML/GPy.

[58] C.J. Permann, D.R. Gaston, D. Andrš, R.W. Carlsen, F. Kong, A.D. Lindsay, J.M. Miller, J.W. Peterson, A.E. Slaughter, R.H. Stogner, R.C. Martineau, MOOSE: Enabling massively parallel multiphysics simulation, SoftwareX 11 (2020) 100430, http://dx.doi.org/10.1016/j.softx.2020.100430.

[59] D.P. Kingma, M. Welling, Auto-encoding variational bayes, in: 2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings, (Ml) 2014, pp. 1–14.

[60] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, PyTorch: An imperative style, high-performance deep learning library, in: H. Wallach, H. Larochelle, A. Beygelzimer, F. dÁlché Buc, E. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems 32, Curran Associates, Inc., 2019, pp. 8024–8035, URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[61] I.M. Sobol', On the distribution of points in a cube and the approximate evaluation of integrals, Zh. Vychisl. Mat. Mat. Fiz. 7 (4) (1967) 784–802.

[62] O.R. Bingol, A. Krishnamurthy, NURBS-Python: An open-source object-oriented NURBS modeling framework in Python, SoftwareX 9 (2019) 85–94.

[63] B.D. Reddy, D. van Huyssteen, A virtual element method for transversely isotropic elasticity, Comput. Mech. 64 (4) (2019) 971–988, http://dx.doi.org/10.1007/s00466-019-01690-7.