International Journal of Solids and Structures 252 (2022) 111769

Contents lists available at ScienceDirect

International Journal of Solids and Structures

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

Optimal design of shape changing mechanical metamaterials at finite strains

Tobias Lichti [a,*], Alexander Leichner [a], Heiko Andrä [a], Ralf Müller [b], Franziska Wenz [c,d], Christoph Eberl [c,e], Angela Schwarz [f], Christof Hübner [f]

[a] *Fraunhofer ITWM, Department of Flow and Material Simulation, Kaiserslautern, Germany*
[b] *TU Darmstadt, Division of Continuum Mechanics, Darmstadt, Germany*
[c] *Fraunhofer IWM, Department of Meso- and Micromechanics, Freiburg im Breisgau, Germany*
[d] *Associated researcher @ Cluster of Excellence livMatS, University of Freiburg, Germany*
[e] *Cluster of Excellence livMatS @ FIT – Freiburg Center for Interactive Materials and Bioinspired Technologies, University of Freiburg, Freiburg, Germany*
[f] *Fraunhofer ICT, Department of Polymer Engineering, Pfinztal, Germany*

ARTICLE INFO

Keywords:
Metamaterials
Multiscale problems
Optimization
Homogenization
Surrogate model

ABSTRACT

Programmable metamaterials establish a new subset of metamaterials offering controllable and variable physical properties. As metamaterials, they are artificial materials and exhibit exotic and counter-intuitive material behavior, but are more specifically tailored for engineering purposes. Whereas for metamaterials a mostly homogeneous layout of unit cells is considered, programmable materials are constructed by an individual distribution in order to satisfy custom intentions regarding a specific shape change under given loading conditions. In order to tackle this customization of material response, a computational optimization framework similar to topology or material optimization is proposed. Our work is based on a multiscale and data approach, allowing a broad range of application with different classes of unit cells and target functions under finite strains. In this contribution, we present the complete process chain from a parametrized unit cell to the final model of the programmable material, ready to be manufactured. We show numerical results with different unit cells and compare them to fully resolved simulations. Further, with the development of new generative manufacturing processes, the production of such programmable materials consisting of spatially varying cells has also become possible on an industrial scale. One example of lab-scale production is shown in the paper and compared to simulation results.

1. Introduction

Metamaterials consist of a periodic array of unit cells and are similar in this property to biological organisms, which are also composed of cells. However, metamaterials are often proposed with uniform unit cells, whereas biological cells differ from stem cells. In this way, they adapt to environmental conditions such as wind loads and moisture during growth.

PMs should also possess this basic property of biological structures, but should not have such a complex structure. In contrast to biological tissue, which adapts to changing conditions by remodeling, the metamaterials considered here have a cellular structure that does not change over time and can therefore only execute a fixed program, i.e. cannot be reprogrammed. It is therefore programmed during the production process. One possibility of programming is a change of geometry under given loading conditions.

In particular, PMs are intended to simplify and replace complex systems where the reaction to environmental conditions is implemented with the help of sensors and actors. For instance, mechatronic systems with switchable states can be replaced by a single material “programmed” to work in the same way (Fratzl et al., 2019; Weisheit et al., 2020). Assuming the unit cells can be sufficiently scaled, the final manufactured PM can be considered as continuum material with locally varying properties. They offer the possibility for distributed control, i.e. the reaction to environmental conditions is triggered and performed locally, in contrast to systems which are controlled by external power sources, actuators sensors and control circuits (Aubin et al., 2022). As a consequence, the material with programmed behavior is more unlikely to fail and, hence, more robust in application compared to a complex mechatronic system.

PMs can be applied wherever a tailored reaction to environmental and operating conditions of a material or a component is required. One example are the wings of an aircraft which have to adapt to starting, climbing, cruising and landing. Adaptable wing structures implemented through a mechanical system as proposed in Grant et al.

---

* Corresponding author.
E-mail address: tobias.lichti@itwm.fraunhofer.de (T. Lichti).

https://doi.org/10.1016/j.ijsolstr.2022.111769
Received 3 March 2022; Received in revised form 3 May 2022; Accepted 31 May 2022

(2010) and Andersen et al. (2007) are one potential field of application. By “programming” an intended behavior of these structures through an optimal inner design of the material simplifies their construction and offers advantages in robustness and weight. Different solutions can be found in the literature, e.g., based on bistable laminates (Daynes et al., 2009) or also topology optimization (Stanford and Ifju, 2008). Also in soft robotics (Mark et al., 2016; Aubin et al., 2022) and prosthetics (Awad et al., 2017) a controlled response of components to individual environmental conditions is necessary and suggests them to be replaced by a PM. Further, whenever standard metamaterials are used there is a potential of enhancement by introducing a spatial gradient in the material. One example are light-weight applications where metamaterials are widely used due to their extreme properties (Chen et al., 2016; Maconachie et al., 2019).

For optimizing the properties of a metamaterial with uniform cells, the optimization of a single cell is sufficient. This can be done with topology optimization where the distribution of material in a specific domain is optimized. Homogenization based approaches (Bendsøe and Kikuchi, 1988), where a density distribution is optimized, are widely used. The topic is extensively treated in the text-books of Allaire et al. (1997) and Bendsøe and Sigmund (2004). A different approach are level-set methods where the boundary of a component is moved according to a specific criterion such as the strain energy distribution (Sethian and Wiegmann, 2000). Another aspect is optimization with the aim to find optimal local material properties at every point of a component. Examples are the optimization of fiber orientation distribution (Frei et al., 2013) or the distribution of different unit-cells inside a component (Ferrer et al., 2018). In order to ensure connectivity between the cells and manufacturability, the unit cell design is restricted, e.g., through a parametrization at the expense of optimality (Wu et al., 2021). Then a mapping from the design to the effective mechanical properties of the cells is necessary. This leads to surrogate based optimization, which is a common technique when the evaluation of the target function is too expensive or its derivative cannot be computed (Booker et al., 1999). One example is the space-mapping technique (Bandler et al., 1994) where a surrogate or coarse model is coupled with a fine model. Most function evaluations are conducted on the coarse model while the fine model is used to verify and iteratively improve the coarse model. Recent approaches in material optimization use high-order polynomials (Imediegwu et al., 2019; Wang et al., 2020) and neuronal networks (White et al., 2019; Zheng et al., 2021) for the mapping of cell design to its mechanical properties.

We aim to establish a target shape under given boundary conditions in an array of cells undergoing large deformations. This is done by optimizing the distribution of design parameters, e.g., a beam thickness or an angle inside an array of cells. In contrast to topology optimization no cells inside the design domain are removed, taking into account robustness of the material and manufacturability. Therefore, several challenges have to be tackled. A boundary value problem (BVP) with a strong coupling between macro- and micro-scale has to be solved. Various cells with highly nonlinear, load case and design dependent behavior have to be considered. The size of these arrays must be sufficiently large, such that it is perceived as a material and to be able to resolve gradients in the design parameters. Further, each cell may have multiple design parameters opening a huge design space. In contrast to recent publications we aim to handle large strains, which poses additional challenges especially for the surrogate model and the derivation of the adjoint problem.

We obtained the results in this paper by combining a surrogate model with a gradient-based optimization method as it was recently done in multiscale topology optimization (Imediegwu et al., 2019; Wang et al., 2020; White et al., 2019; Zheng et al., 2021). Because of the large amount of degrees of freedom in the design space, we consider the adjoint approach. Extensive treatment of this optimization method can be found in Hinze et al. (2009). The domain, which we consider to be programmed, is divided into subdomains, each of which represents a unit cell of the same type, but with different design parameters. In Ferrer et al. (2018) a similar way is presented, but in each subdomain a different unit cell can be placed. In this work, however, the emphasis lies in parametrized unit cells. The total amount of degrees of freedom is then the product of number of design parameters and number of subdomains. Because of the presence of many design possibilities, we follow the common procedure, known from topology optimization, and apply the adjoint sensitivity approach. In addition to the multiscale BVP, this approach requires the solution of a dual BVP.

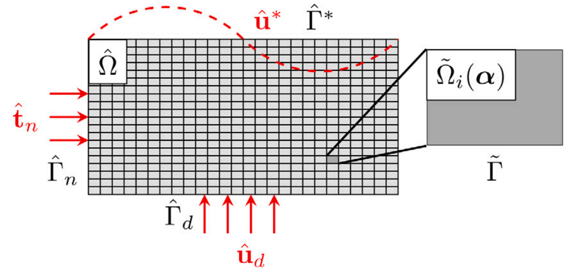
Within the scope of programming materials we couple the design parameters and the final mechanical properties through upscaling. Due to the complex relation between those we follow a data-driven approach in order to handle individually designed unit cells. The surrogate model must provide a fast representation of the data, regardless of the number of dimensions, be able to capture the high nonlinearities and provide smooth derivatives w.r.t. the strain and the design. For that purpose we establish for each unit cell a database filled with homogenized stresses, stored as multidimensional tensor as in Yvonnet et al. (2009). In contrast to the work in that reference, we extend the order of the data tensor by the number of design variables. By the tensor rank decomposition (Tamara and Kolda, 2009) we compress the data size and are further able to interpolate within the tensor entries (Le et al., 2015). In our contribution we use the canonical one-rank decomposition, known as PARAFAC or CANDECOMP. More advanced decomposition methods are the *Tucker* decomposition (Tamara and Kolda, 2009) or *Tensor Train* (Oseledets, 2011; Savostyanov and Oseledets, 2011), which may be applied in our optimization framework, too. Concerning data decomposition and reconstruction, the proper generalized decomposition (Chinesta et al., 2013) is also worth mentioning. It considers the same approach by separated variable interpolation as in our work. In Hirschler et al. (2021) another promising approach via look-up tables of effective properties can be considered for optimization problems, too. The authors in Orlik et al. (2016) show, that an analytical approach for textile-like microstructures is also possible: Instead of creating a database with precomputed stiffnesses and stresses, the necessary gradients for the optimization are obtained through symbolic differentiation.

The paper is structured as follows: In Section 2 we give an overview over the methods and approaches which are used in our optimization framework. First, we start with the concept of unit cell composition and parameters and their connection to the macroscopic properties. After that, the multiscale optimization problem is presented, more precisely as optimization task with an equilibrium equation as constraint. The target is to establish a desired shape under given boundary conditions through an optimal distribution of design parameters. Our solution approach uses the adjoint sensitivity, which is the most convenient way for our optimal control problem. Since this kind of approach relies on differentiation, the need for differentiable functions, describing material behavior, is evident. Therefore, we briefly introduce the tensor rank decomposition and show how to reconstruct and interpolate data for the adjoint sensitivity approach and the gradient descent. For computing the data we consider standard homogenization methods by volume averaging.

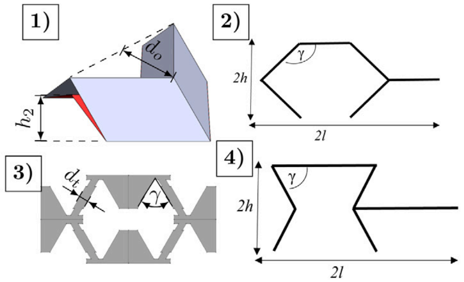
In Section 3 we present examples from our optimization and validate the results. The solutions using the surrogate model are compared to fully resolved simulations and experimental results, confirming the accuracy of our method. Further different unit cells are compared for a programming task. A large scale example demonstrates the efficiency of the presented framework.

2. Mathematical formulation and solution algorithm for optimal design of PMs

In this section we give an overview over PMs and present our ideas in context of implementing certain material behaviors. First, we explain



(a) Multiscale optimization problem: The macroscopic domain $\hat{\Omega}$ is a union of unit cells $\tilde{\Omega}_i$ parameterized by geometric design variables $\boldsymbol{\alpha}$. The goal is to find a distribution $\boldsymbol{\alpha}(\mathbf{X})$ such that a target displacement $\hat{\mathbf{u}}^*$ on the boundary $\hat{\Gamma}^*$ is obtained.



(b) Examples for unit cells $\tilde{\Omega}$ with possible design parameters. **1)** Stacked Miura-Ori cell (see Section 3.1). **2)** and **4)** Honeycomb cell (see Wenz et al. (2021) and Section 3.2) with positive and negative Poisson's ratio through design parameter γ. **3)** Cell with switching Poisson's ratio and inner contact, d_t: diameter of the joints, γ: contact angle (see Wenz et al. (2021))

Fig. 1. Multiscale optimization problem and examples of unit cells from our work and the literature.

the geometrical framework, we currently consider for PMs and explain the issues with these. After that, we show how we tackle those and emphasize the important relation between unit cell parameters and the programming itself.

2.1. Composition of unit cells and the relation to design

In Section 1 we introduced our notion of PMs and explained the difference to classical metamaterials. Here, we want to describe the geometrical conditions which are important to our work: A PM is considered as a domain $\hat{\Omega}$ which is the union of finitely many unit cells $\tilde{\Omega}_i$, for $i = 1, \dots, N$:

$$\hat{\Omega} = \bigcup_{i=1}^{N} \tilde{\Omega}_i. \tag{1}$$

An example is sketched in Fig. 1(a). There, each box represents a unit cell as a subdomain, which we denote with $\tilde{\Omega}_i$ (or $\tilde{\Omega}$). If we consider a certain sub- or micro-domain with emphasis on its set of design parameter, we use $\tilde{\Omega}_i(\boldsymbol{\alpha})$ as notation. The composition of all unit cells is regarded as computational mesh in this work. Each unit cell is equipped with its own set of design parameters $\boldsymbol{\alpha}$. In this contribution, design variables mostly describe geometric properties, e.g. angles, beam thicknesses or lengths which significantly influence mechanical properties of unit cells. Possible unit cells $\tilde{\Omega}$ from our work and from the literature that can be used within an array $\hat{\Omega}$ are shown in Fig. 1(b). In this paper we focus on the stacked Miura-Ori cell (**(1)**) and the honeycomb cell (**(2)** and **(4)**) where the parametrization is chosen in a way to influence their transversal contraction, meeting the requirement to optimize the outer shape of a material under load. Figs. 3(b) and 3(c) the influence of the parametrization of the Miura-Ori cell. The consideration of many other types of unit cells and parametrizations would be possible, such as the example from Wenz et al. (2021) in Fig. 1(b), **(3)**, where a changing transversal contraction is combined with an adjustable stiffness and inner contact.

We assume, that the dimensional scaling is sufficiently large, such that the macroscopic domain $\hat{\Omega}$ can be considered as "homogeneous" material, i.e. the deformation gradient is sufficiently resolved in such a way, that a further miniaturization of the cells will not change the resulting behavior. Note that due to current manufacturing limitations this might not apply in reality, yet. We call a composition of unit cells an *array*.

The macroscopic behavior of a PM is mainly determined by the distribution of design variables inside the array. Therefore, the meaning of the design parameters $\boldsymbol{\alpha}$ in each cell $\tilde{\Omega}_i$ for $i = 1, \dots, N$ is substantial: They are the actual control and allow many different design cases. More precisely, each unit cell in an array may have different parameters. Due to the chosen discretization, the parameters are constant in each $\tilde{\Omega}_i$, i.e. $\boldsymbol{\alpha}$ is a piece-wise constant function over $\hat{\Omega}$

$$\boldsymbol{\alpha}_{|\tilde{\Omega}_i} = \begin{bmatrix} \boldsymbol{\alpha}_1 \\ \vdots \\ \boldsymbol{\alpha}_{M_\alpha} \end{bmatrix} \in \mathbb{R}^{M_\alpha}, \tag{2}$$

where $\boldsymbol{\alpha}_j$ are constants for $j = 1, \dots, M_{\boldsymbol{\alpha}}$. Through this choice of parametrical distribution the assembly of the final design as a resolved array of cells is straight forward. The goal of implementing material behavior is to induce desired mechanical properties by manipulating design parameters. Hence, it is only natural that the programming task is defined as an optimization problem where a mechanical BVP is considered to be the constraints. Therefore, the aim of the optimization is to find a suitable, piece-wise constant parameter distribution $\boldsymbol{\alpha}$ over $\hat{\Omega}$. This optimization task can be considered as equivalent to programming a material behavior.

Due to the complex inner structure of unit cells, they reveal unconventional deformation behavior when subjected to large strains. In general, the design-dependent strain–stress relation cannot be given explicitly and an analytical description is only possible for limited and simple cases. Especially the link between deformation of the array and the design $\boldsymbol{\alpha}$ is not trivial and will be of special interest for this work. The following sections deal with this difficulty. Further, we define the following set of admissible designs or parameter distribution:

$$\mathcal{D} := \left\{ \rho \in \left[H^1(\hat{\Omega})\right]^{M_\alpha} \;\middle|\; c_j \le \rho_j \le C_j,\ j = 1, \dots, M_\alpha \right\} \tag{3}$$

where c_j and C_j are constants and can be understood as production restrictions.

2.2. Multiscale optimization problem

In Section 2.1 we explain the concept of PMs as arrays of parametrized unit cells. In this section the focus is on the "programming" aspect of these arrays.

The multiscale coupling originates from the decomposition of the domain $\hat{\Omega}$ into unit cell domains $\tilde{\Omega}_i$, $i = 1, \dots, N$, see definition in (1). Therefore, the (macroscopic) displacements $\hat{\mathbf{u}}$ are defined as function of the (microscopic) design in this work:

$$\hat{\mathbf{u}} = \hat{\mathbf{u}}(\boldsymbol{\alpha}). \tag{4}$$

By adjusting the distribution of design parameters $\boldsymbol{\alpha}$ on the micro-scale, one can perform the "programming" at macro-scale. The macroscopic deformation behavior has to be considered, too, hence it is subject to the equivalent optimization problem, here for a given cost functional $\mathcal{J}$ as a function of the displacements and design variables:

$$\min_{\boldsymbol{\alpha}\in D}\ \mathcal{J}(\hat{\mathbf{u}}(\boldsymbol{\alpha}),\boldsymbol{\alpha})$$

s.t.

$$\begin{aligned}\mathrm{Div}(\hat{\mathbf{F}}\,\hat{\mathbf{S}})=0 &\quad \text{in}\quad \hat{\Omega},\\ \hat{\mathbf{u}}=\hat{\mathbf{u}}_d &\quad \text{on}\quad \hat{\Gamma}_d,\\ \hat{\mathbf{F}}\,\hat{\mathbf{S}}\,\hat{\mathbf{N}}=\hat{\mathbf{t}}_n &\quad \text{on}\quad \hat{\Gamma}_n,\end{aligned}\tag{5}$$

where $\hat{\mathbf{S}}$ is the second Piola–Kirchhoff stress tensor on the macro-scale, $\hat{\mathbf{N}}$ is the outward normal on $\hat{\Gamma}_n$ and $\mathrm{Div}(\bullet)$ is the divergence w.r.t. the reference configuration. Further, $\hat{\mathbf{F}}$ is the macroscopic deformation gradient defined by

$$\hat{\mathbf{F}}=\nabla_{\mathbf{X}}\hat{\mathbf{u}}+\mathbf{1}.\tag{6}$$

Dirichlet and Neumann boundary conditions are applied on $\hat{\Gamma}_d$ and $\hat{\Gamma}_n$ by $\hat{\mathbf{u}}_d$ and $\hat{\mathbf{t}}_n$, respectively. By assuming a sufficiently large array of cells we can use an hyperelastic constitutive law on the macro-scale which describes the stresses and the tangential stiffness $\hat{\mathbb{C}}$ depending on the strain state $\hat{\mathbf{E}}$ and the design $\boldsymbol{\alpha}$:

$$\hat{\mathbf{S}}=\hat{\mathbf{S}}(\hat{\mathbf{E}},\boldsymbol{\alpha}),\quad \hat{\mathbb{C}}=\frac{\partial\hat{\mathbf{S}}(\hat{\mathbf{E}},\boldsymbol{\alpha})}{\partial\hat{\mathbf{E}}}.\tag{7}$$

Note that $\hat{\mathbb{C}}$ and its derivative w.r.t. $\boldsymbol{\alpha}$ are needed later within the optimization (see Section 2.3). Here, $\hat{\mathbf{E}}$ is the macroscopic Green–Lagrange strain tensor which is defined by

$$\hat{\mathbf{E}}=\frac{1}{2}(\hat{\mathbf{F}}^T\hat{\mathbf{F}}-\mathbf{1})=\frac{1}{2}(\hat{\mathbf{C}}-\mathbf{1}),\tag{8}$$

where $\hat{\mathbf{C}}$ is the macroscopic right Cauchy–Green tensor. The macroscopic stress in (7) is calculated by homogenizing the stress obtained from the solution of a corresponding microscopic BVP:

$$\begin{aligned}\mathrm{Div}(\tilde{\mathbf{F}}\,\tilde{\mathbf{S}})=\mathbf{0} &\quad \text{in}\quad \tilde{\Omega}_i(\boldsymbol{\alpha})\subset\hat{\Omega},\\ \tilde{\mathbf{u}}=(\hat{\mathbf{C}}^{1/2}-\mathbf{1})\tilde{\mathbf{X}}+\tilde{\mathbf{u}}_p &\quad \text{on}\quad \tilde{\Gamma},\\ \tilde{\mathbf{F}}\,\tilde{\mathbf{S}}\,\tilde{\mathbf{N}}\quad \text{anti-periodic} &\quad \text{on}\quad \tilde{\Gamma},\end{aligned}\tag{9}$$

where $\tilde{\mathbf{F}}$ and $\tilde{\mathbf{S}}$ are the microscopic deformation gradient and second Piola–Kirchhoff stress tensor, respectively. The transition from macro- to micro-scale is obtained in terms of the design $\boldsymbol{\alpha}$ and by applying the displacements on the boundary $\tilde{\Gamma}$ of a cell through the macroscopic strain. Note that when calculating $\mathbf{C}^{1/2}$ we assume the absence of rotations, which is admissible as the constitutive law (7) is independent from rigid body rotations. Further, $\tilde{\mathbf{u}}_p$ is a periodic function over $\tilde{\Gamma}$ to ensure a periodic deformation of the cell. Consequently, tractions are anti-periodic over $\tilde{\Gamma}$. The constitutive law on the micro-scale depends on the unit cell's base material:

$$\tilde{\mathbf{S}}=\frac{\partial\tilde{W}(\tilde{\mathbf{E}},\boldsymbol{\beta})}{\partial\tilde{\mathbf{E}}}\tag{10}$$

where $\boldsymbol{\beta}$ are material constants of the microscopic unit cell. In the scope of this work we choose an hyperelastic material law for both, macro- and microscale as all deformations should be fully elastic. Hyperelasticity assumes the existence of an elastic potential (Ogden, 1984) which is suitable for our application.

Note that through the multiscale coupling defined in (4) the cost functional can be simplified, such that $\check{\mathcal{J}}(\boldsymbol{\alpha})=\mathcal{J}(\hat{\mathbf{u}}(\boldsymbol{\alpha}),\boldsymbol{\alpha})$ is a function only of the design $\boldsymbol{\alpha}$. In this work, the cost functional is the difference to a desired state such as a target displacement $\hat{\mathbf{u}}^*$ on the boundary $\hat{\Gamma}^*$ of the array, see Fig. 1(a):

$$\check{\mathcal{J}}(\boldsymbol{\alpha})=\mathcal{J}(\hat{\mathbf{u}}(\boldsymbol{\alpha}),\boldsymbol{\alpha})=\frac{1}{2}\int_{\hat{\Gamma}^*}\|\hat{\mathbf{u}}(\boldsymbol{\alpha})-\hat{\mathbf{u}}^*\|^2\,\mathrm{d}\gamma+\frac{\kappa}{2}\int_{\hat{\Omega}}\|\boldsymbol{\alpha}\|^2+\nabla\boldsymbol{\alpha}\cdot\nabla\boldsymbol{\alpha}\,\mathrm{d}\mathbf{X}.\tag{11}$$

The first integral emphasizes that we consider inverse problems in our work: We track certain distortions or displacements $\hat{\mathbf{u}}^*$ on chosen boundary sides $\hat{\Gamma}^*$ of the macroscopic geometry $\hat{\Omega}$. The second integral is added as Tikhonov based smoothing regulator (Hinze et al., 2009; Frei et al., 2013). The parameter $\kappa\geq 0$ is a penalization factor and can be chosen arbitrarily. A smooth distribution of design parameters is necessary such that the metamaterial shows the intended mechanical behavior and also due to manufacturing restrictions. This can be achieved by using Tikhonov regularization which penalizes the H^1-norm of the design parameters in the cost functional in (11). Beside inverse problems and in general, the cost functional can be a function of stresses, strains, displacements or any other mechanical property. Recall the (discrete) piece-wise constant description of the design variables in (2), such that we may consider parameter functions $\boldsymbol{\alpha}\in\mathcal{D}$ for the cost functional in (11).

The constraints in (5) are formulated as BVP in strong form. For the solution method, these equations are written as the residual of the mechanical virtual work or the weak form of the BVP:

$$\begin{aligned}\langle\mathcal{R}(\hat{\mathbf{u}},\boldsymbol{\alpha}),\hat{\mathbf{v}}\rangle &:= \langle\mathcal{C}(\hat{\mathbf{u}},\boldsymbol{\alpha}),\hat{\mathbf{v}}\rangle-\int_{\hat{\Gamma}_n}\hat{\mathbf{t}}_n\cdot\hat{\mathbf{v}}\,\mathrm{d}\gamma,\\ \langle\mathcal{C}(\hat{\mathbf{u}},\boldsymbol{\alpha}),\hat{\mathbf{v}}\rangle &:= \int_{\hat{\Omega}}\hat{\mathbf{S}}(\hat{\mathbf{E}},\boldsymbol{\alpha})\cdot\delta\hat{\mathbf{E}}\,\mathrm{d}\mathbf{X},\end{aligned}\tag{12}$$

where $\delta\hat{\mathbf{E}}=(\hat{\mathbf{F}}^T\nabla\hat{\mathbf{v}}+\nabla\hat{\mathbf{v}}^T\hat{\mathbf{F}})/2$ is the variational strain tensor. Here, the functional $\mathcal{C}(\hat{\mathbf{u}},\boldsymbol{\alpha})$ takes into account the internal mechanical work. Hence the constraint in (5) can be equivalently written in variational or weak sense:

Find $\hat{\mathbf{u}}\in\hat{\mathbf{u}}_d+V$, such that

$$\langle\mathcal{R}(\hat{\mathbf{u}},\hat{\boldsymbol{\alpha}}),\hat{\mathbf{v}}\rangle=0\quad\forall\hat{\mathbf{v}}\in V,\tag{13}$$

where $V=\{\mathbf{v}\in H^1(\hat{\Omega})\mid\mathbf{v}=0\text{ on }\hat{\Gamma}_d\}$ is the trial or solution space. Dirichlet boundary conditions on $\hat{\Gamma}_d$ are often enforced by considering the Lagrangian or dual formulation. A similar adjoint or dual approach is also used for the optimization problem in (5).

2.3. Optimization method

In this subsection we focus on a solution method for the optimization problem (5). For our work we use the standard gradient descent, which presumes the existence of a sufficiently smooth (Frechét) gradient or derivative of $\check{\mathcal{J}}(\boldsymbol{\alpha})$ w.r.t. the design variable $\boldsymbol{\alpha}$. In general, the resulting solution procedure can be written in iterative form:

$$\boldsymbol{\alpha}^{(j+1)}=\boldsymbol{\alpha}^{(j)}-\theta D_{\boldsymbol{\alpha}}\check{\mathcal{J}}\left(\boldsymbol{\alpha}^{(j)}\right),\tag{14}$$

where $j=1,2,3,\ldots$. The parameter θ is the step size which can be adjusted by some scaling criteria.

In Appendix A we deduce an optimization method, for which we replaced $\mathcal{C}$ in (12) with $\bar{\mathcal{C}}$ from (A.2), which is assumed to be linear. As consequence, the constraint $\mathcal{R}$ in (13) is replaced by a linear alternative $\bar{\mathcal{R}}$. Hence, we call this optimization approach also *linear*. The idea behind our optimization method is to apply this linear optimization approach to our nonlinear multiscale optimization problem (5).

This means that we replace the self-adjoint, linear operator $\bar{\mathcal{C}}(\hat{\mathbf{u}},\boldsymbol{\alpha})$ in the adjoint problem (A.5) with the original $\mathcal{C}(\hat{\mathbf{u}},\boldsymbol{\alpha})$ from (12) and, considering the definition of the cost functional in (11), obtain the adjoint residual $\mathcal{A}$:

$$\begin{aligned}\left\langle\mathcal{A}(\hat{\mathbf{u}},\hat{\lambda},\boldsymbol{\alpha}),\hat{\mathbf{v}}\right\rangle &:= \langle\partial_{\hat{\mathbf{u}}}\mathcal{J}(\hat{\mathbf{u}},\boldsymbol{\alpha}),\hat{\mathbf{v}}\rangle_{V'}+\left\langle\mathcal{C}(\hat{\lambda},\boldsymbol{\alpha}),\hat{\mathbf{v}}\right\rangle\\ &= \int_{\hat{\Gamma}^*}(\hat{\mathbf{u}}-\hat{\mathbf{u}}^*)\cdot\hat{\mathbf{v}}\,\mathrm{d}\gamma+\int_{\hat{\Omega}}\hat{\mathbf{S}}(\hat{\boldsymbol{\Xi}},\boldsymbol{\alpha})\cdot\delta\hat{\boldsymbol{\Xi}}\,\mathrm{d}\mathbf{X},\end{aligned}\tag{15}$$

where analogously to (8) the dual strain tensor $\hat{\boldsymbol{\Xi}}$ and dual deformation gradient $\hat{\boldsymbol{\Upsilon}}$ is defined as:

$$\begin{aligned}\hat{\boldsymbol{\Xi}} &= \frac{1}{2}(\hat{\boldsymbol{\Upsilon}}^T\hat{\boldsymbol{\Upsilon}}-\mathbf{1}),\\ \hat{\boldsymbol{\Upsilon}} &= \nabla\hat{\lambda}+\mathbf{1},\end{aligned}\tag{16}$$

such that $\delta\hat{\boldsymbol{\Xi}} = (\hat{\boldsymbol{\Upsilon}}^T\nabla\hat{\mathbf{v}} + \nabla\hat{\mathbf{v}}^T\hat{\boldsymbol{\Upsilon}})/2$ (compare with weak form in (12)). The variational problem for the total gradient in (A.6) can be obtained analogously to the adjoint problem in (15). The resulting weak form can be written like this:

$$\langle \mathcal{S}, \rho\rangle_{\mathcal{D}} = \langle \partial_{\boldsymbol{\alpha}}\mathcal{J}(\hat{\mathbf{u}}, \boldsymbol{\alpha}), \rho\rangle_{\mathcal{D}} + \left\langle \partial_{\boldsymbol{\alpha}}\mathcal{C}(\hat{\mathbf{u}}, \hat{\lambda}, \boldsymbol{\alpha}), \rho\right\rangle \text{ for } \rho \in \left[H^1(\hat{\Omega})\right]^{M_{\alpha}}, \tag{17}$$

where we used the abbreviation $\mathcal{S} = D_{\boldsymbol{\alpha}}\check{\mathcal{J}}(\boldsymbol{\alpha})$ and the other terms are defined as:

$$\begin{aligned}
\left\langle D_{\boldsymbol{\alpha}}\check{\mathcal{J}}(\boldsymbol{\alpha}), \rho\right\rangle_{\mathcal{D}} &:= \int_{\hat{\Omega}} \mathcal{S}\cdot\rho + \nabla\mathcal{S}\cdot\nabla\rho \, \mathrm{d}\mathbf{X}, \\
\langle \partial_{\boldsymbol{\alpha}}\mathcal{J}(\hat{\mathbf{u}}, \boldsymbol{\alpha}), \rho\rangle_{\mathcal{D}} &:= \kappa \int_{\hat{\Omega}} \boldsymbol{\alpha}\cdot\rho + \nabla\boldsymbol{\alpha}\cdot\nabla\rho \, \mathrm{d}\mathbf{X}, \\
\left\langle \partial_{\boldsymbol{\alpha}}\mathcal{C}(\hat{\mathbf{u}}, \hat{\lambda}, \boldsymbol{\alpha}), \rho\right\rangle &:= \frac{1}{2}\int_{\hat{\Omega}} \left[\partial_{\boldsymbol{\alpha}}\hat{\mathbf{S}}(\hat{\boldsymbol{\Xi}}, \boldsymbol{\alpha})\cdot(\hat{\boldsymbol{\Upsilon}}^T\nabla\hat{\mathbf{u}} + \nabla\hat{\mathbf{u}}^T\hat{\boldsymbol{\Upsilon}})\right] \rho \, \mathrm{d}\mathbf{X}.
\end{aligned} \tag{18}$$

Because of the complicated form of the last integral in (18) we replace $\mathcal{C}$ with an operator $\mathring{\mathcal{C}}$ similar to $\bar{\mathcal{C}}$ from (A.2). In the end, we choose for the gradient w.r.t. $\boldsymbol{\alpha}$ the derivative $\partial_{\boldsymbol{\alpha}}\mathring{\mathcal{C}}$ defined as:

$$\left\langle \partial_{\boldsymbol{\alpha}}\mathring{\mathcal{C}}(\hat{\mathbf{u}}, \hat{\lambda}, \boldsymbol{\alpha}), \rho\right\rangle := \int_{\hat{\Omega}} \left[\nabla\hat{\lambda}\cdot\partial_{\boldsymbol{\alpha}}\hat{\mathbb{C}}(\hat{\mathbf{E}}, \boldsymbol{\alpha})\cdot\nabla\hat{\mathbf{u}}\right] \rho \, \mathrm{d}\mathbf{X}. \tag{19}$$

Note that $\partial_{\boldsymbol{\alpha}}\hat{\mathbb{C}}(\hat{\boldsymbol{\Xi}}, \boldsymbol{\alpha})$ is also a possible choice, but with $\partial_{\boldsymbol{\alpha}}\hat{\mathbb{C}}(\hat{\mathbf{E}}, \boldsymbol{\alpha})$ we observed better numerical results. The following iterative method results from the gradient descent (14) applied to the multiscale optimization task from (5). Because of the derivation from previous section, the method consists of three problems, including the primal and adjoint weak forms in (12) and (15), respectively, as well as the design gradient in (18) and (19). Then, given a feasible $\boldsymbol{\alpha} \in \mathcal{D}$:

1. Solve the *basis* or *primal* problem, i.e. find a primal solution satisfying the constraints in the optimization problem (5) for the given design $\boldsymbol{\alpha}$. The following variational problem corresponds to the second equation of the saddle point problem (A.3) based on the weak forms in (12):
   Find $\hat{\mathbf{u}}_s \in \hat{\mathbf{u}}_d + V$, such that
   $$\langle \mathcal{R}(\hat{\mathbf{u}}_s, \boldsymbol{\alpha}), \hat{\mathbf{v}}\rangle = 0 \quad \forall\hat{\mathbf{v}} \in V. \tag{20}$$
2. Solve the *adjoint* or *dual* problem (A.5). The following variational problem corresponds to the first equation of the saddle point problem (A.3) based on the weak forms in (15):
   Find $\hat{\lambda}_s \in V$, such that
   $$\left\langle \mathcal{A}(\hat{\mathbf{u}}_s, \hat{\lambda}_s, \boldsymbol{\alpha}), \hat{\mathbf{v}}\right\rangle = 0 \quad \forall\hat{\mathbf{v}} \in V. \tag{21}$$
3. Solve the variational problem (A.6) using the expressions from (18) and (19) in order to compute the gradient and update the design:
   For $i = 1, \ldots, M_{\alpha}$ find $\mathcal{S}_i \in H^1(\hat{\Omega})$, such that:
   $$\begin{aligned}
   \int_{\hat{\Omega}} \mathcal{S}_i\rho + \nabla\mathcal{S}_i\cdot\nabla\rho \, \mathrm{d}\mathbf{X} = \kappa_i \int_{\hat{\Omega}} &\boldsymbol{\alpha}_i\rho + \nabla\boldsymbol{\alpha}_i\cdot\nabla\rho \, \mathrm{d}\mathbf{X} \\
   &+ \left\langle \partial_{\boldsymbol{\alpha}_i}\mathring{\mathcal{C}}(\hat{\mathbf{u}}_s, \hat{\lambda}_s, \boldsymbol{\alpha}), \rho\right\rangle \quad \forall\rho \in H^1(\hat{\Omega}), \\
   \boldsymbol{\alpha}_i \leftarrow \boldsymbol{\alpha}_i - \theta\frac{\mathcal{S}_i}{\|\mathcal{S}_i\|}&.
   \end{aligned} \tag{22}$$

We repeat these three steps until convergence has been reached. Note that we update the design variables $\boldsymbol{\alpha}$ entry-wise. Furthermore, we extend the (general) formulation of the Tikhonov regularization in (11) and assign each individual design variable $\boldsymbol{\alpha}_i$ its own penalization factor κ_i. As simplification, we may reduce the regularity of the derivative, i.e. $\mathcal{S}_i \in L^2(\hat{\Omega})$. Then, all differentiations of ρ vanish in the first equation in (22) (since also $\rho \in L^2(\hat{\Omega})$) and we obtain a possibly less smoother solution to $D_{\boldsymbol{\alpha}}\check{\mathcal{J}}(\boldsymbol{\alpha})$ resp. $\mathcal{S}$. In order to ensure that $\boldsymbol{\alpha} \in \mathcal{D}$, we simply project $\boldsymbol{\alpha}_j$ onto c_j and C_j for $j = 1, \ldots, M_{\alpha}$, if they are smaller or larger than the corresponding threshold (see definition of $\mathcal{D}$ in (3)).

From those three steps two critical problems, which are mainly relevant to our work, arise: The evaluation of the stress tensor $\hat{\mathbf{S}}(\hat{\mathbf{E}}, \boldsymbol{\alpha})$ and the stiffness tensor $\hat{\mathbb{C}}(\hat{\mathbf{E}}, \boldsymbol{\alpha})$ as well as the computation of its derivative $\partial_{\boldsymbol{\alpha}}\hat{\mathbb{C}}(\hat{\mathbf{E}}, \boldsymbol{\alpha})$ as functions of the design $\boldsymbol{\alpha}$. For the weak form of the adjoint problem (15) we need to evaluate $\hat{\mathbf{S}}(\hat{\boldsymbol{\Xi}}, \boldsymbol{\alpha})$, but this can be done analogously to $\hat{\mathbf{S}}(\hat{\mathbf{E}}, \boldsymbol{\alpha})$. In the following, we give some comments on that.

The evaluation of the stresses is necessary since we apply linear solvers for the numerical solution to the BVP in the multiscale optimization problem (5). Hence, the residual equation in (20) is linearized, for which the material tangent $\mathbb{C}$ is necessary. Because we consider arrays of freely designed unit cells, analytical expressions for stress–strain-relations are not at hand. The computation and final assembly of $\partial_{\boldsymbol{\alpha}}\hat{\mathbb{C}}$ is also not trivial, but very important for our work. As mentioned, for unit cells there are no general explicit relations between design $\boldsymbol{\alpha}$ and $\hat{\mathbf{u}}$ on the one hand and the stresses $\hat{\mathbf{S}}$ and stiffness $\mathbb{C}$ on the other, hence those derivatives are not available, too.

Our approach is to approximate the material law by a data based surrogate model, which is easy to evaluate. For that purpose we construct a multidimensional, smooth interpolation between different cases of designs and loadcases. The resulting virtual material model allows the evaluation of both the material tangent and its design derivative. In addition, fully resolved computational meshes can be avoided when using surrogate models and the final computational mesh can be reduced to the structured grid we defined in (1). Obviously, this reduces the computational time tremendously, which we demonstrate in our numerical experiments.

2.4. Surrogate material model

To compute the derivatives for the gradient descent algorithm, a smooth and differentiable function of the nonlinear mechanical behavior of unit cells is needed. Especially the dependence of the mechanical behavior on the design $\boldsymbol{\alpha}$ can be highly nonlinear. In addition, the nonlinear influence of the strain state has to be considered as well. Especially when dealing with arbitrary unit cells, closed analytic expressions are difficult to find. Therefore, numerical homogenization is used to establish a database for surrogate models describing them. In contrast to typical homogenization applications with microstructures of composite materials, the separation of scales is smaller in our application. Therefore the accuracy of the macroscopic computation depends on the number of unit cells and the gradients of $\boldsymbol{\alpha}$ in the final array. As mentioned in Section 1, our data space is high dimensional as we interpolate in both, the strain and the design space. Further due to the optimization context, the smoothness of the derivatives is crucial. We therefore chose to store the data in a structured tensor and use tensor decomposition and spline reconstruction for our surrogate model.

2.4.1. Homogenization and data sampling

In order to establish a database for the surrogate model, the mechanical response of a unit cell $\tilde{\Omega}$ (recall set-up in Fig. 1(a)) is computed for all relevant design and strain possibilities (in the following we skip the subscription for the subdomain $\tilde{\Omega}$ of the unit cell). This data sampling can be considered as precomputing part of the material programming and has to be done only once.

The purpose of the database and the resulting surrogate model is to replace the macroscopic constitutive law (7) by providing an accurate mapping from design and macroscopic strain values to the stress response within an array of unit cells. Instead of solving the microscopic BVP for each cell (see (9)) during the optimization, a surrogate model is evaluated which provides the macroscopic stress $\hat{\mathbf{S}} = \hat{\mathbf{S}}(\hat{\mathbf{E}}, \boldsymbol{\alpha})$ which is a function of the strain state and the design.

During the sampling process, the macroscopic stress is obtained by solving the microscopic BVP (see (9)) for given $\hat{\mathbf{E}}$ and $\boldsymbol{\alpha}$ and volume averaging over the unit cell:

$$\begin{aligned}
&\hat{\mathbf{S}} = \hat{\mathbf{F}}^{-1}\left\langle\tilde{\mathbf{P}}\right\rangle \\
\text{with} \quad &\left\langle\tilde{\mathbf{P}}\right\rangle = \frac{1}{|\tilde{\Omega}|}\int_{\tilde{\Omega}} \tilde{\mathbf{P}} \, \mathrm{d}\mathbf{X}
\end{aligned} \tag{23}$$

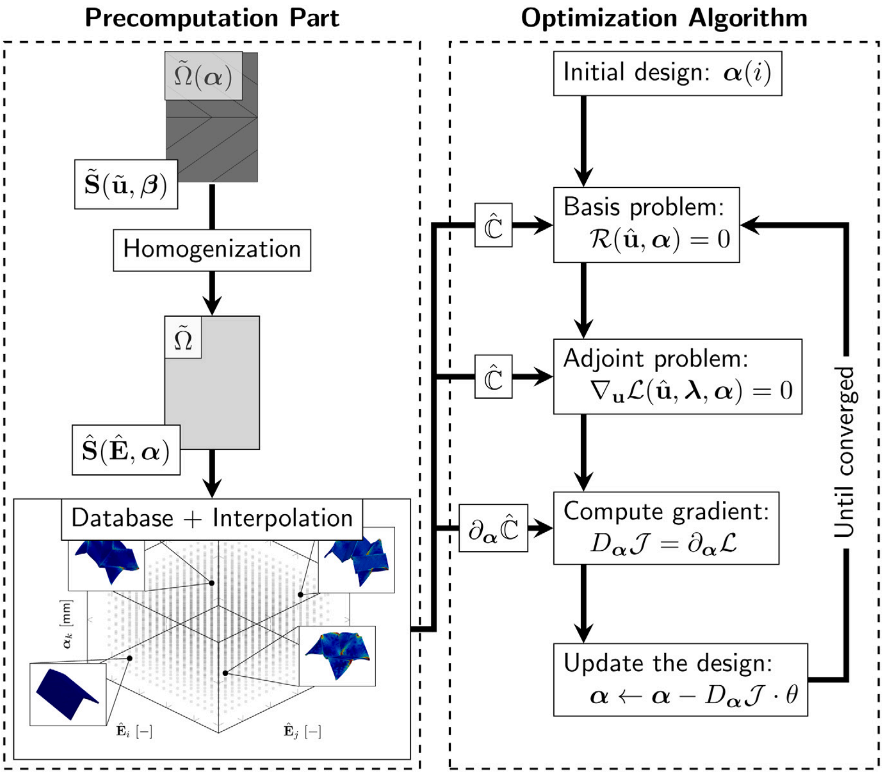

Precomputation Part
Optimization Algorithm
$\tilde{\Omega}(\boldsymbol{\alpha})$
$\tilde{\mathbf{S}}(\tilde{\mathbf{u}}, \boldsymbol{\beta})$
Homogenization
$\hat{\Omega}$
$\hat{\mathbf{S}}(\hat{\mathbf{E}}, \boldsymbol{\alpha})$
Database + Interpolation
α_k [mm]
$\hat{\mathbf{E}}_i$ [−]
$\hat{\mathbf{E}}_j$ [−]
Initial design: $\boldsymbol{\alpha}(i)$
$\hat{\mathbb{C}}$
Basis problem: $\mathcal{R}(\hat{\mathbf{u}}, \boldsymbol{\alpha}) = 0$
$\hat{\mathbb{C}}$
Adjoint problem: $\nabla_{\mathbf{u}}\mathcal{L}(\hat{\mathbf{u}}, \boldsymbol{\lambda}, \boldsymbol{\alpha}) = 0$
$\partial_{\boldsymbol{\alpha}}\hat{\mathbb{C}}$
Compute gradient: $D_{\boldsymbol{\alpha}}\mathcal{J} = \partial_{\boldsymbol{\alpha}}\mathcal{L}$
Update the design: $\boldsymbol{\alpha} \leftarrow \boldsymbol{\alpha} - D_{\boldsymbol{\alpha}}\mathcal{J} \cdot \theta$
Until converged

Fig. 2. Multiscale optimization algorithm: Precomputation part with generation of the database through numerical homogenization and adjoint optimization algorithm based on the data based surrogate model.

where $|\tilde{\Omega}|$ is the volume of the periodicity cell $\tilde{\Omega} = \tilde{\Omega}(\boldsymbol{\alpha})$. Note that the volume of the periodicity cell is independent from the design $\boldsymbol{\alpha}$ in order to allow an arrangement of cells on a Cartesian grid. Further, $\hat{\mathbf{F}}$ is known from the BVP (see (9)). By making use of the divergence theorem and the balance of momentum the volume integral in (23) can be rewritten as an integral over the cell's boundaries (Hill, 1972):

$$\langle \tilde{\mathbf{P}} \rangle = \frac{1}{|\tilde{\Omega}|} \int_{\tilde{\Gamma}} t \otimes \mathbf{X} \, \mathrm{d}\gamma \tag{24}$$

where $\otimes$ denotes the dyadic product.

For the construction of the database, we traverse all design and strain cases in a canonical way, and solve the microscopic BVP in (9) in order to obtain the macroscopic stress tensor $\hat{\mathbf{S}}$ for all of those cases. These stress data are then understood as sample points and are ordered on a Cartesian grid, see Fig. 2. Due to the nonlinear behavior of the unit cells, the microscopic BVP in (9) has to be solved iteratively. To accelerate the sampling procedure, we adjust the iteration steps to lie on sampling points for our database, which can be stored at the same time. After sampling, the collected data points can be represented by a high-dimensional hypercube, whose dimensions correspond both to a design variables $\boldsymbol{\alpha}_j$ where $j = 1, \dots, M_{\boldsymbol{\alpha}}$ and strain variables $\hat{\mathbf{E}}_i$ where $i = 1, \dots, M_{\mathbf{E}}$ (in Voigt notation) where $M_{\boldsymbol{\alpha}}$ and $M_{\mathbf{E}}$ are the number of design variables and strain components, respectively. Mathematically, this hypercube can be written as multidimensional tensor:

$$\mathbf{D} = (\mathbf{D}_I)_{I \in \mathcal{I}} = (\mathbf{D}_{i_1,\dots,i_M})_{(i_1,\dots,i_M) \in \mathcal{I}} \in \mathbb{R}^{m_1 \times \cdots \times m_M} \tag{25}$$

where M is the order of the tensor and the multi index $I = (i_1, \dots, i_M)$ describes the expansions of the variable space. Each index i_k for $k = 1, \dots, M$ runs within its specific interval in $\mathcal{I}_k$ resulting in a multidimensional hypercube represented as Cartesian product of the design and strain possibilities. More precisely, we consider the canonical product $\mathcal{I} = \mathcal{I}_1 \times \cdots \times \mathcal{I}_M$ such that $i_k \in \mathcal{I}_k = \{1, \dots, m_k\}$ where m_k is the number of sampling points of the k-th design or strain variable. Each entry $\mathbf{D}_I$ stores an entry $\hat{\mathbf{S}}_j$ of the homogenized stress tensor $\hat{\mathbf{S}}$ for different design and strain cases, hence we need between three and six different data tensors in total (depending on the spatial dimension).

For later purposes it is necessary to distinguish between the design and strain variables. Therefore, we sort the index space $\mathcal{I}$ accordingly:

$$\begin{aligned} \mathcal{I} &= \mathcal{I}_1 \times \cdots \times \mathcal{I}_{M_{\boldsymbol{\alpha}}} \times \mathcal{I}_{M_{\boldsymbol{\alpha}}+1} \times \cdots \times \mathcal{I}_{M_{\boldsymbol{\alpha}}+M_{\mathbf{E}}} \\ &:= \mathcal{I}_1 \times \cdots \times \mathcal{I}_{M_{\boldsymbol{\alpha}}} \times \mathcal{J}_1 \times \cdots \times \mathcal{J}_{M_{\mathbf{E}}} \end{aligned} \tag{26}$$

such that we have $M_{\boldsymbol{\alpha}}$ design variables and $M_{\mathbf{E}}$ strain variables, hence $M = M_{\boldsymbol{\alpha}} + M_{\mathbf{E}}$. For a more convenient notation we write $\mathcal{J}_j := \mathcal{I}_{M_{\boldsymbol{\alpha}}+j}$ for $j = 1, \dots, M_{\mathbf{E}}$. In general, the entry $\mathbf{D}_{i_1,\dots,i_M}$ of the data tensor represents the solution to the microscopic BVP in (9) with the *configuration*:

$$(\boldsymbol{\alpha}_{i_1}, \dots, \boldsymbol{\alpha}_{i_{M_{\boldsymbol{\alpha}}}}, \hat{\mathbf{E}}_{j_1}, \dots, \hat{\mathbf{E}}_{j_{M_{\mathbf{E}}}}), \tag{27}$$

where $(i_1, \dots, i_{M_{\boldsymbol{\alpha}}}) \in \mathcal{I}_1 \times \cdots \times \mathcal{I}_{M_{\boldsymbol{\alpha}}}$ and $(j_1, \dots, j_{M_{\mathbf{E}}}) \in \mathcal{J}_1 \times \cdots \times \mathcal{J}_{M_{\mathbf{E}}}$. The term configuration refers to a feasible design and loadcase scenario for the microscopic BVP in (9). According to the data sampling for $\mathbf{D}$, we have to solve the microscopic BVP for $m_1 \cdots m_M$ different configurations.

Retrieving the data points $\mathbf{D}_I$ is partly done by solving the microscopic BVP in (9) and partly through reconstruction of (possibly inaccessible) data points: As the database is structured as a hypercube, it is possible that inaccessible points are required for which the microscopic BVP in (9) cannot be solved. Further, the high number of dimensions (1 or more in the design space and 3 or 6 in the strain space) leads to a high number of data points that has to be sampled. Therefore, a possibility to reconstruct missing data points is needed. Here we use the idea presented in Yvonnet et al. (2013). At locations of missing data points we discretize the Laplacian with finite differences (FD). In this way we construct an overdetermined system of equations which can be solved iteratively. In contrast to Yvonnet et al. (2013) we use higher order FD which yields smoother reconstruction and allows to compute only a small part of the required data points.

2.4.2. Tensor rank decomposition and interpolation

Recall from previous sections that the knowledge about the stress–strain relation of a unit cell array is generally not on hand. Since analytical or approximative methods are not available (yet), we consider data based material models. In the previous subsection we explain how to establish the database and retrieve the stress data. Now we continue with the processing of this database.

For the final optimization method, we seek a more convenient representation of the data tensor $\mathbf{D}$. To this end, we follow the approach from Yvonnet et al. (2009) and introduce the tensor rank decomposition (TRD) as least squares problem:

$$\inf_{D_i^r} \left\| \mathbf{D} - \sum_{r=1}^{R} \bigotimes_{i=1}^{M} D_i^r \right\|^2, \tag{28}$$

where R is a given integer and is called rank. If the quadratic error in (28) is exactly zero, then R is the rank of the tensor $\mathbf{D}$ and is defined as the minimal integer for that equation to hold (Tamara and Kolda, 2009), assuming such a decomposition exists. Note that $D_k^r \in \mathbb{R}^{m_k}$ are one-rank tensors (or ordinary vectors). Hence, the data tensor is approximated by a series of outer products of vectors, denoted by "⊗". We rewrite the least squares problem in (28) by means of the split index space from (26):

$$\begin{aligned} \mathbf{D} &\approx \sum_{r=1}^{R} D_1^r \otimes \cdots \otimes D_{M_{\boldsymbol{\alpha}}}^r \otimes D_{M_{\boldsymbol{\alpha}}+1}^r \otimes \cdots \otimes D_{M_{\boldsymbol{\alpha}}+M_{\mathbf{E}}}^r \\ &:= \sum_{r=1}^{R} D_1^r \otimes \cdots \otimes D_{M_{\boldsymbol{\alpha}}}^r \otimes C_1^r \otimes \cdots \otimes C_{M_{\mathbf{E}}}^r \end{aligned} \tag{29}$$

where we write for convenience $C_j^r := D_{M_{\boldsymbol{\alpha}}+j}^r$ for $j = 1, \dots, M_{\mathbf{E}}$. This representation yields the component-wise expression according to (26)

$$\mathbf{D}_{i_1,\dots,i_M} \approx \sum_{r=1}^{R} \left(\prod_{k=1}^{M_{\boldsymbol{\alpha}}} \left(D_k^r\right)_{i_k} \prod_{l=1}^{M_{\mathbf{E}}} \left(C_l^r\right)_{j_l} \right), \tag{30}$$

where $\left(D_k^r\right)_{i_k}$ is the i_kth component of the vector D_k^r and $\left(C_l^r\right)_{j_l}$ is the j_lth component of the vector C_l^r, respectively. Through that distinction between designs and strains the construction of the design gradient becomes easier, which is necessary for the optimization method described by (20)–(22). Because of the BVP in the optimization problem (5), the evaluation of the stress tensor between different configurations (see definition in (27)) is crucial: Given any design parameters $(\boldsymbol{\alpha}_1, \dots, \boldsymbol{\alpha}_{M_{\boldsymbol{\alpha}}})$, we pursue the tensor $\mathbf{S}$ at the (arbitrary) load case $(\hat{\mathbf{E}}_1, \dots, \hat{\mathbf{E}}_{M_{\mathbf{E}}})$ for which there is no data entry in $\mathbf{D}$. In this case, we canonically interpolate within the nearest data interval around the considered configuration. Note we consider only design and loading configurations within the interval product I, i.e. $\min\{\boldsymbol{\alpha}_{i_k}\}_{i_k \in \mathcal{I}_k} \le \boldsymbol{\alpha}_k \le \max\{\boldsymbol{\alpha}_{i_k}\}_{i_k \in \mathcal{I}_k}$, for $k = 1, \dots, M_{\boldsymbol{\alpha}}$. Analogous for $\hat{\mathbf{E}}_l$ and $l = 1, \dots, M_{\mathbf{E}}$ such that $\min\{\hat{\mathbf{E}}_{j_l}\}_{j_l \in \mathcal{J}_l} \le \hat{\mathbf{E}}_l \le \max\{\hat{\mathbf{E}}_{j_l}\}_{j_l \in \mathcal{J}_l}$.

For the interpolation we use the vectors $D_k^r \in \mathbb{R}^{m_k}$ as interpolation nodes for $k = 1, \dots, M_{\boldsymbol{\alpha}}$. Then the interpolating function is defined as:

$$\bar{D}_k^r(\boldsymbol{\alpha}_k) := \sum_{n=1}^{m_k} N_n(\boldsymbol{\alpha}_k)\, (D_k^r)_n, \tag{31}$$

where $m_k = |\mathcal{I}_k|$ is the number of indices. Analogously we define the interpolating function $\bar{C}_l^r(\hat{\mathbf{E}}_l)$ for the vectors C_l^r and $l = 1, \dots, M_{\mathbf{E}}$. For the basis interpolating functions N_n we choose B-Splines (de Boor, 1978) due to their smoothing properties and the local influence of data points, but any other choice works here, too. The final interpolating function is defined similarly to the component representation (30):

$$\bar{\mathbf{D}}(\boldsymbol{\alpha}_1, \dots, \boldsymbol{\alpha}_{M_{\boldsymbol{\alpha}}}, \hat{\mathbf{E}}_1, \dots, \hat{\mathbf{E}}_{M_{\mathbf{E}}}) = \sum_{r=1}^{R} \left(\prod_{k=1}^{M_{\boldsymbol{\alpha}}} \bar{D}_k^r(\boldsymbol{\alpha}_k) \prod_{l=1}^{M_{\mathbf{E}}} \bar{C}_l^r(\hat{\mathbf{E}}_l) \right). \tag{32}$$

The interpolating function $\bar{\mathbf{D}}$ approximates the data tensor $\mathbf{D}$ with the following property:

$$\bar{\mathbf{D}}(\boldsymbol{\alpha}_{i_1}, \dots, \boldsymbol{\alpha}_{i_{M_{\boldsymbol{\alpha}}}}, \hat{\mathbf{E}}_{j_1}, \dots, \hat{\mathbf{E}}_{j_{M_{\mathbf{E}}}}) = \mathbf{D}_{i_1,\dots,i_M} + \delta_{i_1,\dots,i_M} \tag{33}$$

where $\delta_{i_1,\dots,i_M}$ is the error originating from interpolating at the given configuration and from the TRD of the least squares problem (28). In Yvonnet et al. (2009) the authors call the idea in (32) the separated variables interpolation (SVI) which describes the convenient property of this approach well: This variable separation allows now a very simple differentiation w.r.t. to the design, i.e. only $\bar{D}_k^r$ in (32) has to be differentiated for $r = 1, \dots, R$ and $k = 1, \dots M_{\boldsymbol{\alpha}}$. Since we use smooth B-Splines, the evaluation of both the interpolating functions N_n and their derivatives are based on the same efficient algorithm. The differentiation of $\bar{\mathbf{D}}$ w.r.t. the strain variables $\hat{\mathbf{E}}_l$ can be done analogously for $l = 1, \dots, M_{\mathbf{E}}$, but now we have to consider $\bar{C}_l^r$. Since $\mathbf{D}$ stores (homogenized) stress data, the differentiation of $\bar{\mathbf{D}}$ w.r.t. $\hat{\mathbf{E}}_l$ yields the tangential stiffness $\hat{\mathbb{C}}$.

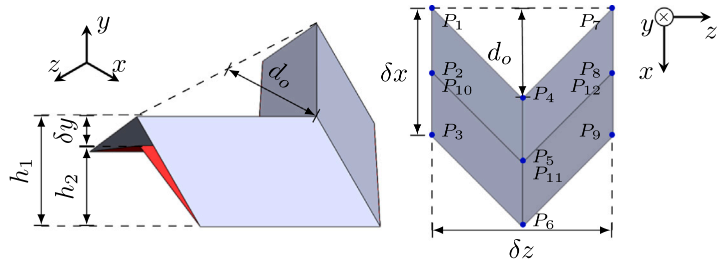
Obviously, the amount of components in $\mathbf{D}$ is equal to $\prod_{n=1}^{M} m_n = \prod_{k=1}^{M_{\boldsymbol{\alpha}}} |\mathcal{I}_k| \prod_{l=1}^{M_{\mathbf{E}}} |\mathcal{J}_l|$. Hence, unit cells with several design parameters, i.e. a high number $M_{\boldsymbol{\alpha}}$, become an issue: The data to the $M_{\mathbf{E}}$ strain cases are then multiplied with the number of design parameters. Before collecting all data (we explain this in Section 2.4.1) one has to think about the amount of data points for the strain cases. Considering the multiplicative character of the workload, one has to decide carefully about the number of design cases for each design parameter. Nevertheless, it is inevitable, that the data tensor $\mathbf{D}$ might become very large. Through the TRD, however, it is possible to approximate a large amount of data with one-rank tensors, such that the total data amount can be reduced to $R \sum_{k=1}^{M} m_k$. Note that in most cases the rank R is so small, that this data compression leads indeed to a smaller data storage. In our case, the amount of storage typically reduces to approx. 1×10^{-3} times the original storage requirement $\prod_{k=1}^{M} m_k$.

In Appendix B we give some more insights of how the tensor rank decomposition can be used for our purposes. However, we skipped so far the important question whether a TRD exists for all data tensors $\mathbf{D}$ in (28), or if there are always vectors D_i^r such that the quadratic error in (28) can be sufficiently minimized. Since we intend to only show how the TRD can be applied to our use case, we refer to Tamara and Kolda (2009) and Savostyanov and Oseledets (2011), Ballester-Ripoll et al. (2016) for some comments about that issue.

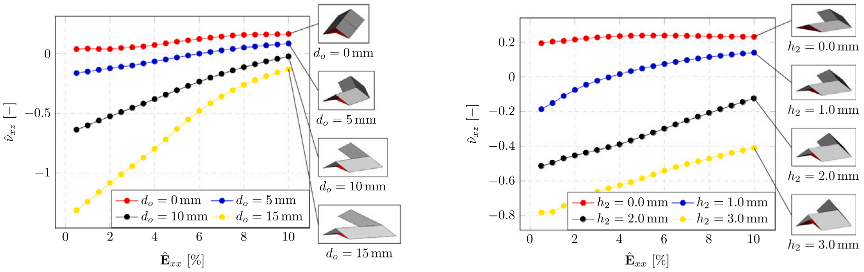
2.5. Algorithm and code

As conclusion to this section we summarize all presented approaches into an algorithm, which we use in our software *ProgMatSim* (ProgMatSim, 2022) in order to obtain our numerical results. The actual algorithm is divided into a precomputing part (see Sections 2.4.1 and 2.4.2) and an optimization part (see Sections 2.2 and 2.3). In Fig. 2 a flow chart presents the two main items of the complete algorithm and how they depend on each other.

All algorithms in pseudocode are presented in Appendix C: The precomputing part is sketched in Algorithm 2, where we apply the methods from the python package *TensorLy* for the TRD. Note that we precompute the database only once for every unit cell. In Algorithm 3 we show how the SVI can be implemented. As interpolating basis functions we use B-Splines from the python package *scipy*. The actual "material programming" is presented in Algorithm 1, where we describe the usage of the interpolating function $\bar{\mathbf{D}}$ as surrogate model. In principle, it is a algorithmic summary of the Eqs. (20)–(22). Note that we skip the numerical treatment of the primal and dual problems in the algorithmic sketch, since it would be out of the scope of this paper. However, we apply the open-source finite element software *CalculiX* (Calculix, 2021) to the microscopic BVPs for generating the database and to both macroscopic BVPs: (20) and its adjoint version (21). The first equation in (22) is solved via the finite difference method, where we transform the weak form into strong form. Furthermore, we pad the macroscopic domain $\hat{\Omega}$ with additional, artificial cells, so called ghost cells, on each boundary face and apply homogeneous boundary conditions on those cells.



(a) Geometry of the stacked Miura-Ori unit cell with design parameters $\boldsymbol{\alpha} = [\alpha_1, \alpha_2]^T = [d_o, h_2]^T$. P_i are the vertices defining the geometry where P_{10}, P_{11} and P_{12} belong to the red layer.



(b) Influence of d_o on Poisson's ratio $\hat{\nu}_{xz}$, $h_2 = 1.5$ mm

(c) Influence of h_2 on Poisson's ratio $\hat{\nu}_{xz}$, $d_o = 7.5$ mm

Fig. 3. Miura-Ori unit cell based on folded sheets.

3. Results

In this section we present some examples for PMs which we computed by means of our in-house software ProgMatCode (Programmable, 2021). In Section 3.1 we consider examples with a cell based on stacked Miura-Ori foils for which we use a data-driven surrogate model. In Section 3.2 we show the applicability of our algorithm to a 2D honeycomb cell described by an analytical model. Comparison to fully resolved models and, in case of the Miura-Ori cell, to manufactured samples are provided as well.

The geometries in all examples are arrays of cells where L_i, δ_i, n_i are the size of the array, the size of the cell and the number of cells in the respective direction $i \in \{x, y, z\}$. The left and right boundaries are denoted by X_1 and X_2, analogously for Y and Z.

3.1. Stacked Miura-Ori foil

In this section we present some numerical and real experiments with a Miura-Ori based unit cell. First the geometry is described and a surrogate model established. Subsequently, the influence of array sizes and design gradients are analyzed. Finally two optimization examples of medium and large size are shown.

3.1.1. Unit cell geometry

The concept of metamaterials made from stacked Miura-Ori is known from the literature (Schenk et al., 2014). The cell geometry consists of four parallelograms with a thickness of 0.4 mm which are arranged to an arrow like shape as shown in Fig. 3. The parameters

$$\delta x = 10\,\text{mm}, \quad \delta y = 2\,\text{mm}, \quad \delta z = 14.14\,\text{mm} \tag{34}$$

are the dimensions of the cell and must remain constant when changing the design of the cell. Consequently the offset d_o and the height of the lower layer h_2 remain variable where $\delta y = h_1 - h_2 = \text{const}$.

The cell shows an anisotropic auxetic behavior. In the scope of this work we are especially interested in the Poisson's ratio

$$\hat{\nu}_{xz} = \frac{\mathrm{d}\hat{\mathbf{E}}_{zz}}{\mathrm{d}\hat{\mathbf{E}}_{xx}} \tag{35}$$

which is the transversal contraction in z-direction following an elongation in x-direction. It can be controlled by the design parameters d_o and h_2, respectively as shown in Figs. 3(b) and 3(c). It is calculated by applying a strain in x-direction and measuring the elongation in z-direction for a single cell. For large d_o the cell shows an auxetic behavior with $\hat{\nu}_{xz} < 0$ which is caused by the unfolding of the parallelograms. This effect decreases up to $\hat{\nu}_{xz}(d_o = 0) \approx 0$. The Poisson's ratio decreases with strain in x-direction when the foils are increasingly stretched. Due to the different heights h_1 and h_2 of the stacked cells (see Fig. 3(a)) unfolding of the lower layer is completed earlier. Therefore, we can influence the transversal contraction with h_2 as well. For $h_2 = 1.0$ mm (see blue curve in Fig. 3(c)) we can observe a change in sign of the transverse contraction which indicates that the lower foil is stretched completely and the contraction is dominated by the effective behavior of the base material.

The design parameters of the Miura-Ori cell influence its outer shape, which makes a periodic arrangement of cells with different designs impossible. To tackle this problem, the boundary points of neighboring cells are averaged. When connecting two cells in x-direction, the points P_3, P_6 and P_9 are merged with P_1, P_4 and P_7 of the neighboring cell. In y-direction, the cells are connected at P_2, P_5 and P_8 on the upper side and P_{10}, P_{11} and P_{12} on the lower side. Therefore, a gradient in the design parameters introduces an error to the constitutive law as the microscopic domain $\tilde{\Omega}$ in the BVP (see (9)) is assumed to be a periodic cell. Therefore the parameter distribution must be sufficiently smooth in order to reduce the distortion of cells.

3.1.2. Establishing the surrogate model

For the Miura-Ori cell a two dimensional design space for d_o and h_2 is covered. Additionally, the 3D strain space is covered in order to capture the nonlinear effect especially of h_2 at large strains shown in Fig. 3(c). Therefore we establish a surrogate model depending on the design space and the strain space in terms of the Green–Lagrange tensor

$\hat{\mathbf{E}}$ according to (27). We sample homogenized stresses for

$$
\begin{aligned}
\boldsymbol{\alpha}_1 &\rightarrow d_o \in \{0,\dots,16\}\mathrm{mm}, && n = 5\\
\boldsymbol{\alpha}_2 &\rightarrow h_2 \in \{0,\dots,3.0\}\mathrm{mm}, && n = 7\\
\hat{\mathbf{E}}_1, \hat{\mathbf{E}}_3 &\rightarrow \hat{\mathbf{E}}_{xx}, \hat{\mathbf{E}}_{zz} \in \{0.05,\dots,0.15\}, && n = 9\\
\hat{\mathbf{E}}_2 &\rightarrow \hat{\mathbf{E}}_{yy} \in \{-0.1,\dots 0.2\}, && n = 7\\
\hat{\mathbf{E}}_4, \hat{\mathbf{E}}_5, \hat{\mathbf{E}}_6 &\rightarrow \hat{\mathbf{E}}_{xy}, \hat{\mathbf{E}}_{xz}, \hat{\mathbf{E}}_{yz} \in \{-0.045,\dots,0.045\}, && n = 7
\end{aligned}
\tag{36}
$$

resulting in an overall number of 6 806 335 data points. Due to the symmetry of shear stresses the amount of data points that is sampled reduces to 1 270 080. For each data point, the BVP (9) has to be solved. Due to unfeasible boundary conditions in certain points, especially on the boundaries of the strain domain, a solution is not possible such that in the end, we sample 603 111 data points. The missing points are a posteriori reconstructed using a FD approximation and by mirroring w.r.t. the shear strains (Yvonnet et al., 2013).

The geometries and meshes are generated using Gmsh (Geuzaine and Remacle, 2009). Each cell is discretized with approximately 650 to 1800 linear triangular shell elements, depending on d_o and h_2. The microscopic BVPs are solved using the open source FE-solver CalculiX (Calculix, 2021).

The TRD is done with $R = 50$ and second order B-splines are used for the interpolation of the data. A high value of R was necessary to improve the approximation at the border of the strain and design space, which in turn improves the robustness of the simulation.

3.1.3. *Validation of the surrogate model*

The surrogate model described in Section 2.4 is computed through homogenization of a periodic unit cells under periodic boundary conditions. Therefore, it is only applicable if the separation of scales is large enough, i.e. for sufficiently large arrays. Further, the gradient of the design parameters must be sufficiently small. The aim of this section is to estimate the error introduced by the surrogate model depending on the size of the unit cell array. We will focus on the one hand on the influence of boundary conditions and on the other hand the influence of the gradient $\partial d_o/\partial x$ as representative example of the design variables $\boldsymbol{\alpha}$.

In addition to the accuracy we will rate the efficiency of the surrogate model. Therefore, we will compare computation times using the surrogate model to solving a fully resolved model.

For this study, arrays with

$$n_x \in \{4, 8, 16, 32, 64\}, \quad n_y = 4, \quad n_z = 1$$

cells are studied. Fig. 4(a) shows a description of the geometry and the boundary conditions for the case $n_x = 4$. The dimensions of the geometry are given by

$$L_x = n_x\delta_x, \quad L_y = n_y\delta_y, \quad L_z = n_z\delta_z$$

where δ_x, δ_y and δ_z are the dimensions of the cell defined in (34) and Fig. 3(a). The design parameter d_o is varied along x with

$$d_o = f(x) = 10 + 5\sin\left(\frac{2x - L_x}{2L_x}\pi\right) \qquad x \in [0, L_x] \tag{37}$$

resulting in a smooth transition between $d_o(X_1) = 5$ and $d_o(X_2) = 15$. Consequently, $\partial d_o/\partial x$ decreases with increased L_x (and n_x). The meshes for the resolved and the surrogate model are shown in Fig. 4(b). The resolved model is meshed with the same element size used in the data generation process for the surrogate model. The surrogate model uses one brick-element per cell. Both, the surrogate model and the resolved model are solved using Calculix (2021).

Table 1
Comparison of computation times between resolved and surrogate model. Each time is the average of 3 runs.

Number of cells [−]	16	32	64	128	256
Resolved [s]	10.77	36.05	65.07	141.33	328.33
Surrogate [s]	0.38	0.78	1.50	3.00	6.31
Speed-up [−]	28.34	46.21	43.38	47.11	51.98

We compare the displacements of the surrogate model and a resolved simulation in z-direction under tension in x-direction. Each variant is subjected to a macroscopic strain of 5% and simulated with confined and unconfined boundary conditions:

$$
\begin{array}{ll}
\quad\text{Confined:} & \text{Unconfined:}\\
X_1 : \hat{\mathbf{u}}_x = \hat{\mathbf{u}}_z = 0 & X_1 : \hat{\mathbf{u}}_x = 0\\
Y_1 : \hat{\mathbf{u}}_y = 0 & Y_1 : \hat{\mathbf{u}}_y = 0\\
Z_1 : \hat{\mathbf{u}}_z = 0 & Z_1 : \hat{\mathbf{u}}_z = 0\\
X_2 : \hat{\mathbf{u}}_x = 0.05L_x & X_2 : \hat{\mathbf{u}}_x = 0.05L_x\\
\quad\ \hat{\mathbf{u}}_z = 0. &
\end{array}
\tag{38}
$$

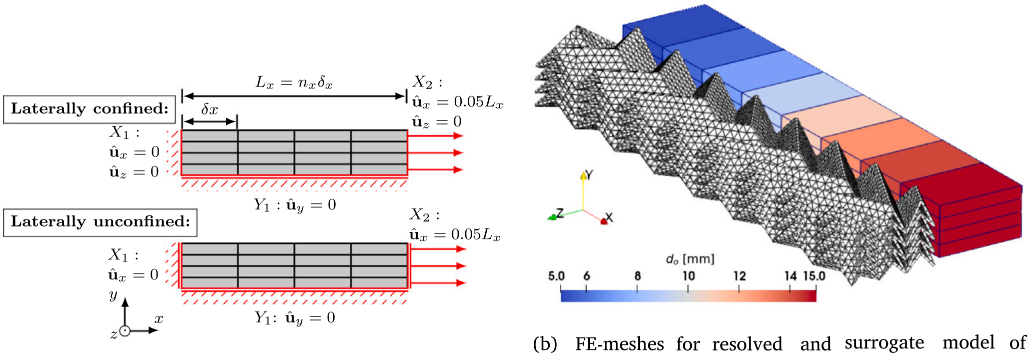
The influence of the boundary conditions is reported in Fig. 5(a) where $\hat{\mathbf{u}}_z$ is plotted over the normalized X-coordinate. For clarity, only the plots for $n_x \in \{4, 8, 16\}$ are shown. The displacements $\hat{\mathbf{u}}_z$ on the boundary Z_2 of the resolved model are calculated at discrete locations $x_i = i \cdot \delta_x$ for $i \in \{0, 1, \dots, n_x\}$ by averaging the nodal displacements in the interval $(x_i - \delta_x/2) < X \leq (x_i + \delta_x/2)$.

One can see that on the left boundary, after the first cell, the value of $\hat{\mathbf{u}}_z$ is almost equal in the confined and unconfined configuration. On the right boundary, displacements on the first two cells are different. Recall that on the left boundary $d_o = 5\,\mathrm{mm}$ while on the right boundary $d_o = 15\,\mathrm{mm}$. Therefore $\hat{\mathbf{u}}_z$ is larger on the right boundary due to the larger Poisson's ratio (see Fig. 3(b)) and thus also the difference to the unconfined configuration. Due to the displacement BCs, the cell boundary is forced to remain planar. However, the out-of-plane deformation under periodic BCs increases with d_o. Therefore the auxetic unfolding of the cell is suppressed on the boundary—an effect that increases with d_o. From this, we conclude that an influence of the boundary conditions takes place only in the first 1–2 cells.

Next we define an error as mean absolute difference of $\hat{\mathbf{u}}_z$ at the locations x_i defined above. In Fig. 5(b) this error is plotted versus n_x with red lines. Further we calculate the same error in the domain $X/L_x \in [0.125, 0.625]$ indicated in Fig. 5(a) where we have independence of the boundary conditions. Both errors decrease when more cells are used in x-direction. The large error across the entire domain of the unconfined configuration (red dashed line) is introduced due to the different behavior of the resolved and the surrogate model on the boundary. In contrast to the resolved cell, the surrogate model can deform freely as its faces remain planar due to the discretization by one brick-element per unit cell. The error in the confined configuration is smaller as the error on the boundary is zero due to the applied BCs ($\hat{\mathbf{u}}_z(X_1) = \hat{\mathbf{u}}_z(X_2) = 0$). The decrease of the error in the middle of the geometry (blue lines) should be independent of boundary effects and is therefore attributed to the lower gradient of the design parameter. For $n_x > 16$ we see almost no decrease anymore.

We can conclude that unconfined BCs introduce a large error as the surrogate does not resolve out of plane deformations on the boundaries and therefore deforms freely. Besides the boundary effects, the influence of the gradient $\partial d_o/\partial x$ is rather small. Nevertheless a smooth distribution of d_o and h_2 is important to ensure manufacturability of the cells.

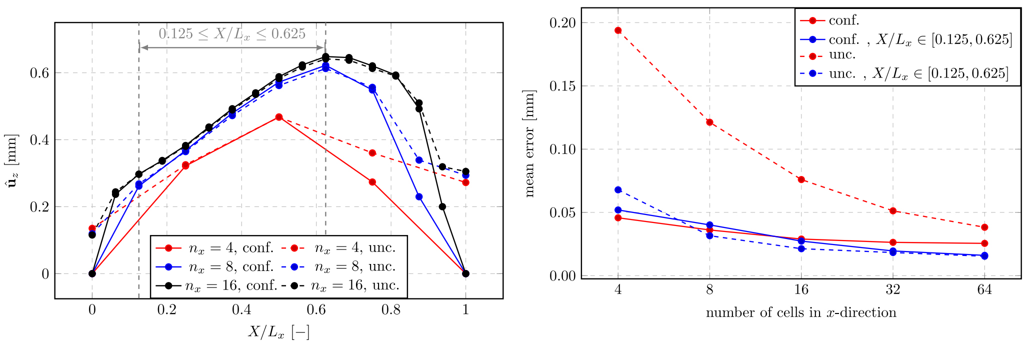
To rate the efficiency of surrogate model, computation times of each simulation are measured. The results are shown in Table 1. Each measured time is the average of 3 runs. The amount of Newton–Raphson iterations is between 9 and 10 in each computation. As stated above, we use one element per unit cell in the surrogate model whereas in the resolved model approximately 1000 finite elements are necessary. However, the evaluation of the equations for our surrogate model (see


Laterally confined:
$L_x = n_x\delta_x$
δx
X_2 : $\hat{\mathbf{u}}_x = 0.05L_x$ $\hat{\mathbf{u}}_z = 0$
X_1 : $\hat{\mathbf{u}}_x = 0$ $\hat{\mathbf{u}}_z = 0$
Y_1 : $\hat{\mathbf{u}}_y = 0$
Laterally unconfined:
X_2 : $\hat{\mathbf{u}}_x = 0.05L_x$
X_1 : $\hat{\mathbf{u}}_x = 0$
Y_1: $\hat{\mathbf{u}}_y = 0$
y
x
z
Y
Z
X
d_o [mm]
5.0 6 8 10 12 14 15.0

(a) Description of geometry and boundary conditions for confined and unconfined configuration. The case $n_x = 4$ is depicted.

(b) FE-meshes for resolved and surrogate model of an array with $n_x = 8$. Geometry split along x-axis. Coloring indicates value of the design parameter d_o, black edges show discretization

Fig. 4. Problem description for the validation study.



(a) Displacement $\hat{\mathbf{u}}_z$ on Z_2 of the resolved model for different array sizes and BCs

(b) Average error between resolved and surrogate model depending on the number of cells in x-direction

Fig. 5. Results of the study with different array sizes and BCs.

(B.1)) take longer than the evaluation of a standard elastic material law. Therefore, the computation time does not scale with the number of elements. In this study we could observe a speed-up of approximately 50 for arrays with more than 16 cells.

3.1.4. *Sinusoidal shape with foil*

In this section we treat the problem shown in Fig. 6: Optimize the shape of an array consisting of 40×2×12 = 960 cells, each equipped with two design parameters. The resulting optimization problem has 1920 design parameters. Based on the conclusions of Section 3.1.3 and for comparison with experimental results, we define boundary conditions comparable to a tensile test with a macroscopic strain of 5% on the array:

$$\begin{aligned} &\hat{\mathbf{u}}(X_1) = [0,0,0] \\ &\hat{\mathbf{u}}(X_2) = [20\,\mathrm{mm},0,0] \\ &\hat{\mathbf{u}}_y(Y_2) = 0. \end{aligned} \tag{39}$$

Under the given boundary conditions we want to minimize the cost-function (see Eq. (11)) where the desired shape on the boundaries Z_1 and Z_2 is given by a displacement $\hat{\mathbf{u}}_z^*$ described by

$$\begin{aligned} f_1(x) &= p_1 \sin(p_2(x+p_3)) + p_4(x+p_5) \quad \text{on } Z_1 \\ f_2(x) &= -f_1(-x+400) \quad \text{on } Z_2 \end{aligned}$$
$$\text{with } p_1 = 1.87,\ \ p_2 = 0.0187,\ \ p_3 = 9.04,\ \ p_4 = -2.22\times10^{-3},\ \ p_5 = 329.1. \tag{40}$$

The same target function will be used in Section 3.2.2 with a different cell.

The initial guess for the design parameters is a uniform distribution of $d_o = 7.5\,\mathrm{mm}$ for the offset and $h_2 = 1.5\,\mathrm{mm}$ for the height of the second layer. The parameters d_o and h_2 are restricted between 1 mm and 15 mm and 0.1 mm and 2.9 mm, respectively.

The programmed behavior of the final result is validated on the one hand with a resolved simulation. Therefore, the geometry is reconstructed and meshed with approximately 6×10^5 linear triangular shells using Gmsh (Geuzaine and Remacle, 2009).

On the other hand, the practical implementation of our programmed material is demonstrated with a manufactured sample. As mentioned above, the Miura-Ori cell is based on folded, stacked foils. The base material is the thermoplastic polyurethane (TPU) *Platilon U2102A* by *Covestro* in the form of thin films with a thickness of 300 μm. The folding is done in a thermoforming process on a *ILLIG KFG 37a* at a temperature of 380 °C within 30 s followed by cooling with a blower. Therefore, molds are 3D-printed with thermoplastic *ABS*. Due to fabrication restrictions, the original resolved geometry in Fig. 7(c) is scaled down by 25%. The molds can be used multiple times which enables the scaling of the process. After thermoforming, the single layers are connected point-wise with solvent welding. For the solution, 15% TPU are solved in dimethylformamide and applied with a pipette. The tensile test is performed on a *Hegewald & Peschke Inspekt Table Blue* with a load-cell of 500 N. Boundary conditions according to (39) are applied: The sample is clamped on Z_1 and Z_2. A spacer is used between the layers to compensate the height difference. Finally a displacement of $0.75\cdot20\,\mathrm{mm} = 15\,\mathrm{mm}$ is applied at a testing speed of $5\,\mathrm{mm\,s^{-1}}$. The displacements on the surface of the sample are measured with digital image correlation using the software *GOM Correlate* (Gom, 2021). The correlated displacements are shifted by a constant to compensate

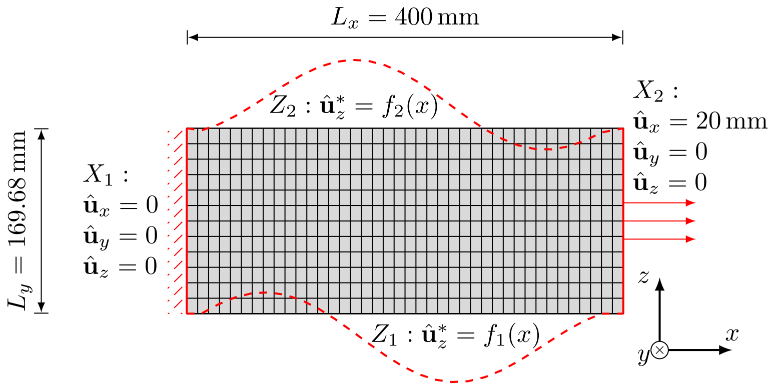
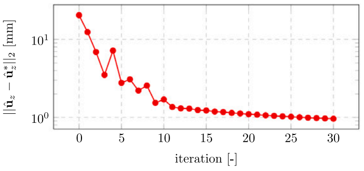
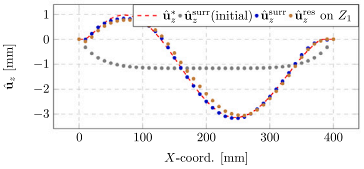


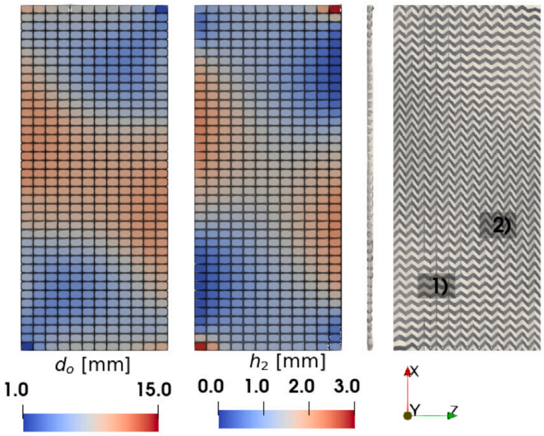
Fig. 6. Description of the optimization problem: Array of 40 × 2 × 12 cells under tensile-test conditions and a target shape on Z_1 and Z_2.



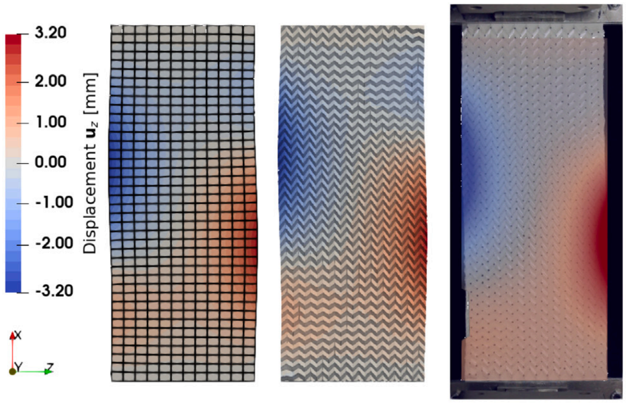
(a) Objective function: difference between target and displacement on the boundaries Z_1 and Z_2



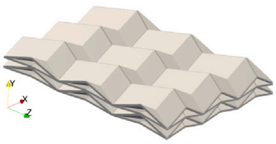
(b) Target function, shape of surrogate model (initial and optimized) and resolved model for the side Z_1. Z_2 is point-wise symmetric.



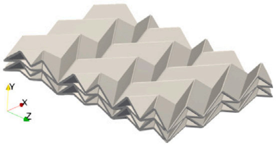
(c) Parameter distribution and resolved geometry. Magnified structures of **(1)** and **(2)** are below.



(d) Displacement field $\mathbf{u}_z$ on the surrogate model (left, min/max −3.18/3.07mm), resolved model (middle, min/max −3.28/3.20mm) and manufactured sample (right, min/max −3.44/3.45 mm) at 5% strain. Target min/max: −3.14/3.14 mm



(e) Magnified structure at **(1)** in 7(c)



(f) Magnified structure at **(2)** in 7(c)

Fig. 7. Optimization of a sinusoidal shape for a Miura-Ori foil stack with two layers.

for relative movements between camera and sample such that the displacements are centered around zero.

The results of the optimization and comparison to resolved simulation and manufactured sample are shown in Fig. 7. The error evolution in every iteration is plotted in Fig. 7(a). The optimization converges after approximately 20 iterations. We start with a large step length θ in the gradient descent (22) and reduce the step when the error increases.

The solution is a distribution of the design parameters d_o and h_2 and is shown in Fig. 7(c), along with the resolved geometry. Smoothness of the design parameters is imposed by using a Tikhonov regularization in (22) with $\kappa = 1 \times 10^{-7}$ for d_o and $\kappa = 1 \times 10^{-4}$ for h_2 in order to ensure manufacturability of the optimized geometry.

In Fig. 7(b) the target displacement $\hat{\mathbf{u}}_z^*$, the displacement on the boundary when using the surrogate model $\hat{\mathbf{u}}_z^{\text{surr}}$ and when using a resolved model $\hat{\mathbf{u}}_z^{\text{res}}$ are shown for the boundary Z_1. For clarity, only the side Z_1 is shown in the plots as the target function and the solution are point-wise symmetric (see Fig. 6). The displacements $\hat{\mathbf{u}}_z^{\text{res}}$ on the boundary of the resolved geometry are again calculated by averaging (see Section 3.1.3) and are therefore considered to be macroscopic quantities. In the initial configuration one can see that the geometry expands in z-direction under tension in x-direction due to the auxetic properties of the cell. After the optimization, the surrogate model and the resolved simulation agree very well with the target displacement. The displacement fields of the surrogate model, the resolved model and the manufactured sample are shown in Fig. 7(d) together with the minimum and maximum values. In general, the agreement of the three cases to the target is very good. The higher minimum and maximum values in the displacement field of the resolved model and the manufactured sample are attributed to larger fluctuations compared to macroscopic fields where quantities are homogenized. The largest relative difference to the target is observed in the manufactured sample where the magnitude of displacements are approximately 10% larger. Here, an additional source of error may be the initial clamping force at the beginning of the measurement which causes an additional loading of the sample.

In general, parametrized Miura-Ori cells can be used to program the behavior of foils. The production of samples in a lab-scale and the accordance of their deformation behavior to the optimization target could be demonstrated. Further, the thermoforming procedure has the potential to be used for the production on an industrial scale.

3.1.5. Large scale optimization: two sides of a die

Generating the database for our surrogate model is computationally expensive due to the large amount of microscopic BVP that have to be solved. Nevertheless a precomputed surrogate model gives access to large scale problems. To demonstrate the efficiency of our approach, we optimize the design of a cube with $50 \times 250 \times 35 = 437\,500$ cells via d_o and h_2, i.e. 875 000 design parameters. The problem setup is shown in Fig. 8. The geometry is fully clamped on X_1 and X_2. On X_2 a displacement of 1.5 mm is applied. On Y_1 and Y_2 $\hat{\mathbf{u}}_y$ and $\hat{\mathbf{u}}_z$ are restricted. The target shapes are two sides of a die with flat surfaces and 4 eyes on Z_1 and 3 eyes on Z_2 as depicted in Figs. 9(b) and 9(c) right side. It combines two length scales, the flat face on the one hand and local elevations on the other hand. As initial guess we use a uniform distribution of $d_o = 10\,\text{mm}$ and $h_2 = 1.5\,\text{mm}$.

The optimization converges after approximately 20 iterations, where the error is reduced by a factor of 100 as shown in Fig. 9(a). Again, the step length θ in the gradient descent is reduced as soon as the error increases. In Figs. 9(b) and 9(c) the displacements in z-direction are shown. In the initial configuration there is a rounded elevation on both faces. In the optimized configuration both the flat shape and the local elevations fit very well with the target. The optimized distributions of d_o and h_2 are shown in Figs. 9(d) and 9(d). Similar to the target, the solution varies on a small scale in the region of the clampings and the local elevations and on a large scale through the whole volume.

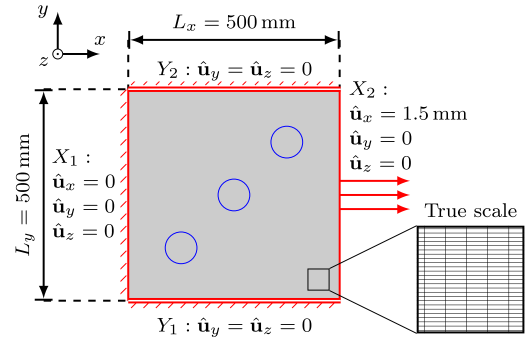


Fig. 8. Problem description for the large scale example. Magnification is in true scale and shows the arrangement of the cells. Blue circles symbolize the target shape $\hat{\mathbf{u}}^*$.

Reconstructing a resolved geometry for this example would require 3.5×10^6 surfaces (8 per cell) and the resulting mesh between 300×10^6 and 800×10^6 finite elements (650–1800 per cell, see Section 3.1.2). Recall that during the optimization the FE problem has to be set up and solved twice per iteration (once for basis problem (20) and the adjoint problem (21), see also Fig. 2), which is not possible and makes the usage of a surrogate model necessary. Both, the generation of such geometries and the FE-simulation of such large scale problems is challenging and beyond the scope of this paper.

3.2. Honeycomb cell

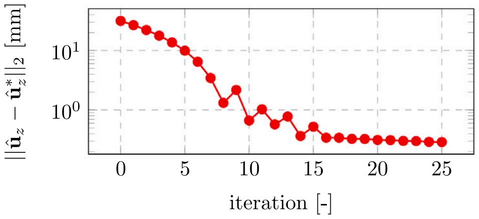
Our optimization method is applicable to a wide range of unit-cells where the description of their design-dependent mechanical behavior is not restricted to data-based surrogate models used in the previous examples. If available, analytical models are well suited due to their efficiency. To demonstrate the flexibility of our optimization approach, we present in this section the optimization of the problem in Section 3.1.4, albeit with a different unit cell.

3.2.1. Cell geometry and model

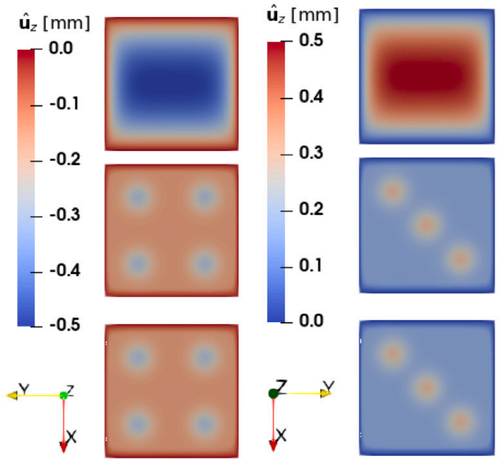
The cell is depicted in Fig. 10 and referred to as honeycomb-cell. It is made of thin plates that form a hexagon. The angle γ can be changed to obtain a specific Poisson's ratio (Han and Lu, 2018; Mirzaali et al., 2018). For angles larger than 90 degrees the Poisson's ratio is positive, otherwise negative (auxetic). The cells can be connected to an array with locally different behavior. To ensure connectivity, nodes that are shared by more than one cell have to be averaged. Similar to the Miura-Ori cell (see Section 3.1.1), this leads to slightly unsymmetrical cells. To make sure that we can assume periodic behavior the gradient of γ between neighbor cells should be low. For small deformations, a homogenized model can be obtained analytically under the assumption of elastic beams at plane strain. The size of the unit cells is constant (h, l) and the chosen angle is changing the lengths of the single beams respectively. More detail can be found in Wenz et al. (2021), Appendix.

3.2.2. Sinusoidal shape with honeycomb array

As stated above, this example uses the same geometry, boundary conditions and target function as in Section 3.1.4 (see Fig. 6). Therefore we have an array of $40 \times 12 = 480$ cells with one design parameter per cell. The geometry is stretched by 5% and our initial design is a uniform distribution of the design parameter with $\gamma = 60°$ in every cell. For the optimization we restrict γ in a range between 45° and 120°, exploiting the capability of the cell to change between positive and negative Poisson's ratio. Fig. 11 shows a comparison of target, initial and final displacement on the boundary of the surrogate model and a comparison to the resolved model.

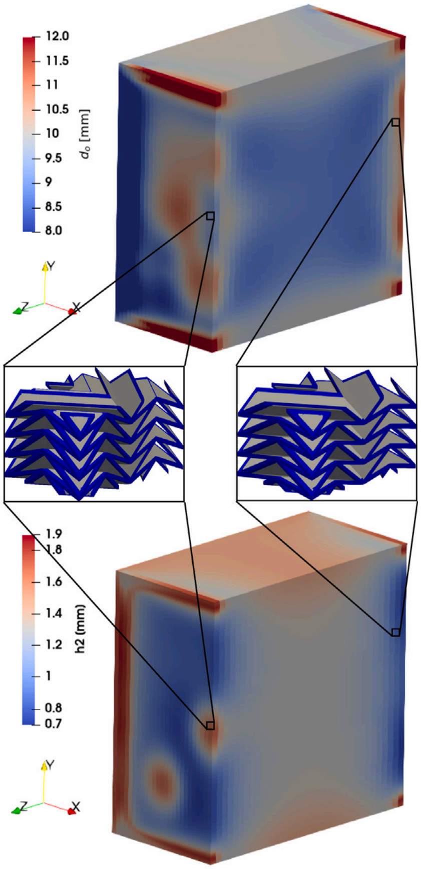


(a) Objective function: difference between target and displacement on the boundaries Z_1 and Z_2



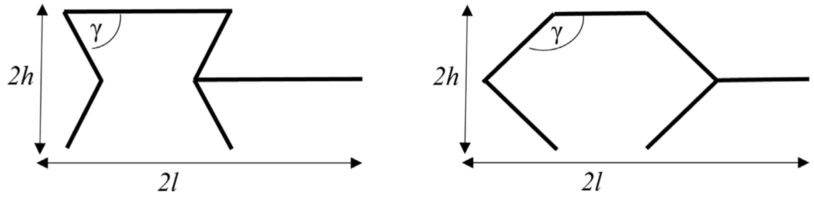
(b) Initial displacement, final displacement and target in z-direction on Z_1

(c) Initial displacement, final displacement and target in z-direction on Z_2



(d) Optimized geometries with microstructures

Fig. 9. Results of the optimization. Optimized distributions in (d) of d_o (min: 2.7 mm, max: 15 mm) and h_2. The geometry is split in half on x/z-plane to show the parameter distribution inside the array. The microstructure for selected regions is displayed in between. The feature edges are highlighted in blue.

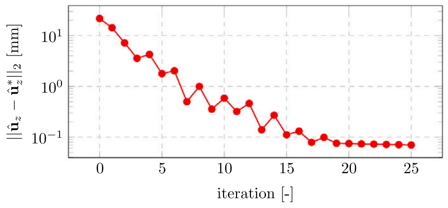


(a) Angle smaller than 90 degrees

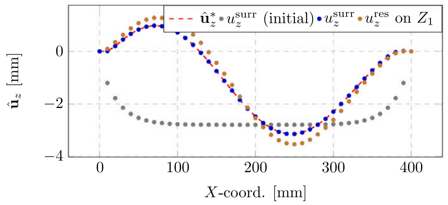
(b) Angle larger than 90 degrees

Fig. 10. Honeycomb cells.

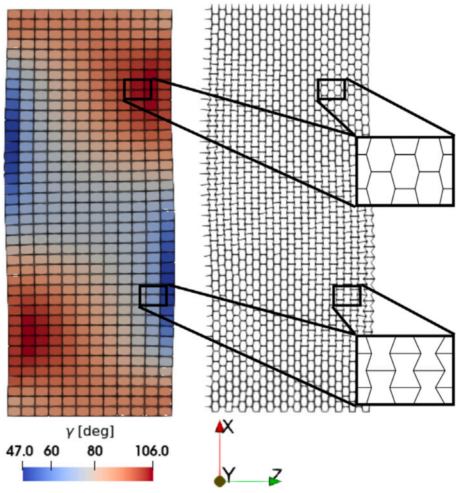
Again, for clarity we only show the lines for Z_1 as target and solution are point-wise symmetric. With the initial design, the array expands due to the auxetic properties of the cell for $\gamma = 60°$. The optimization yields a smooth distribution of γ shown in Fig. 11(c) without using regularization. Also the final solution does not exploit the whole range of γ that was allowed. Therefore, convergence is faster and the final error smaller compared to the optimization with the Miura-Ori cell as one can see in the error plots Figs. 11(a) and 7(a). The displacements on the boundary of the optimized geometry depicted in Fig. 11(b) match very well with the target. Again there is a mismatch between the surrogate model and the resolved model due to the gradients in the strain fields and the design parameter. The distortions of the cells can be seen in the deformed resolved geometry (see Fig. 11(c)). Besides this, the large deformations introduce an additional error as the analytically derived surrogate model is only suitable for small deformations. This can be shown by reducing the stretching by a factor of 10 to 0.5%. As a consequence, the distance norm between surrogate and resolved model decreases by a factor of approximately 40 from 2.70 mm to 6.74×10^{-2} mm which is larger than the decrease expected due to the scaled displacements by approx 0.1.



(a) Objective function: difference between target and displacement on the boundaries Z_1 and Z_2



(b) Target function, shape of surrogate model (initial and optimized) and resolved model for the side Z_1. Z_2 is point-wise symmetric



(c) Optimized distribution of γ and the corresponding resolved geometry. Magnified structure shows cells with $\gamma > 90°$ and auxetic cell with $\gamma < 90°$

Fig. 11. Optimization of a sinusoidal shape with a beam-based honeycomb cell.

Algorithm 1 Programming Materials through Optimization

```
Initialize the design $\boldsymbol{\alpha}$
Initialize error $e \leftarrow \infty$
Choose error tolerance $\delta < \infty$
while $e > \delta$ do
    Solve the original BVP (20):
        Compute $\mathbf{S}(\hat{\mathbf{E}}, \boldsymbol{\alpha})$ and $\mathbb{C}(\hat{\mathbf{E}}, \boldsymbol{\alpha})$ using the surrogate model in (32)
        Apply numerical solver, e.g. Newton-Raphson
    Solve the adjoint problem (21):
        Compute $\mathbf{S}(\hat{\boldsymbol{\Xi}}, \boldsymbol{\alpha})$ and $\mathbb{C}(\hat{\boldsymbol{\Xi}}, \boldsymbol{\alpha})$ using the surrogate model in (32)
        Apply numerical solver, e.g. Newton-Raphson
    Update the design as described in (22):
        Compute derivative $\partial_{\boldsymbol{\alpha}} \mathbb{C}(\hat{\mathbf{E}}, \boldsymbol{\alpha})$ using the surrogate model in (32)
        Compute design derivative $\mathcal{S} = D_{\boldsymbol{\alpha}} \check{\mathcal{J}}$ and apply gradient descent
    Update the error:
        $e \leftarrow \check{\mathcal{J}}(\boldsymbol{\alpha})$
end while
```

4. Discussion

We propose a framework to program a desired mechanical response and shape morphing of unit cell based metamaterials under given boundary conditions through the distribution of design variables. We consider arrays of unit cells, equipped with design parameters that are used to adjust the mechanical behavior of each cell. The distribution of those design parameters is subject of our optimization.

The resulting multiscale optimization problem has a large number of design parameters and is therefore solved using an adjoint optimization approach. The macroscopic BVP and the adjoint problem are solved using a surrogate model describing the nonlinear design dependent mechanical behavior of the cells. This allows us to compute the derivatives w.r.t. the design that are needed in the optimization. At the same time it provides a fast solution method for the structural analysis. The mechanical behavior of simple cells can be described using analytical models, which are very efficient. For more complex cases, arbitrary cells and large deformation we use a data-driven approach. A database of effective stresses on a multidimensional grid of strains and cell designs is precomputed. The database is then compressed using tensor rank decomposition. Interpolation within the tensor entries is done using B-spline interpolation. The precomputation of the database can easily be done in parallel as all sampling points are independent. The generation of the database is further accelerated by reconstruction of missing points and using the symmetry of stress data w.r.t. shear strains. By using parametrized unit cells, manufacturability of the programmed material is ensured.

Many publications deal with the optimization of multiscale structures, especially their topology (Wu et al., 2021). We extent the state of the art by applying the optimization to large deformations. Thus, new challenges especially regarding the derivation of the adjoint optimization problem and the surrogate model have to be tackled. Recent publications use data based surrogate models with polynomials (Imediegwu et al., 2019; Wang et al., 2020) or neural networks (White et al., 2019; Zheng et al., 2021) to interpolate in the design space. In contrast to these, we expand the tensor based surrogate model for finite deformations of hyperelastic materials introduced in Yvonnet et al. (2013) and map both, design and strain space. Further our work distinguishes from classical topology optimization as no cells are added or removed and relies on the distribution of design parameters only. This leads to a robust design regarding the failure of single cells and production imperfections.

By combining computational homogenization, a data-driven surrogate model and adjoint optimization, a modular framework which is applicable to various kind of unit cells and highly nonlinear material behaviors is obtained.

We show numerical examples with two different cells: A stacked Miura-Ori cell described with a data-driven surrogate model and a honeycomb cell which is analytically described. In a first example, we study the applicability of the data-driven model depending on the size of cell array by comparing its results to those of a fully resolved model. Especially unconfined boundary conditions introduce a large error near the boundary while the variation of design parameters between neighboring cells causes only small errors.

Based on these findings, we optimize the distribution of d_o and h_2 in an array of Miura-Ori cells to form a curved boundary under tension. Comparison of target, surrogate model, resolved model and a manufactured sample show a good agreement.

The same target is used to optimize an array of honeycomb cells. In the optimized solution there are cells with $\gamma > 90°$ and $\gamma < 90°$, resulting in a local positive and negative Poisson's ratio. The difference to the resolved model is slightly larger compared to the previous example as the analytical model only applies for small strains.

To demonstrate the efficiency of our framework, we optimize a large scale example with 437 500 cells. The target shapes are two surfaces of a die, combining different length scales. Due to the high resolution, the target is matched very well. The optimization yields a smooth distribution of the design parameters through the whole design domain.

Different issues deserve further investigations. First, an ideal step size would lead to faster convergence. However, adjusting the step size is a difficult task as it depends on the problem setup and has to be adjusted for each parameter for all cells. Second, the computational expense for generating the surrogate model can be reduced. The reconstruction of missing data points can potentially be used for a majority of the data points. The identification of points to compute is a crucial task to reduce the amount of precomputations. Third, the estimation of the error introduced in the surrogate model as a consequence of gradients between neighboring cells needs further investigations. Within this work, we studied the error depending on the gradient of a single design parameter in one direction. A general method to quantify the error depending on local gradients of the design parameters is desirable. Finally, a comparison of optimization at different scales of the target function considered in this paper could be interesting. We generate a material while classical topology optimization generates a structure. In general, an optimization of all scales will yield results that are closest to the optimum. An investigation on which scale to optimize regarding the trade-off between optimality, computational expense, robustness and manufacturability seems interesting but is beyond the scope of this paper.

In general we believe that the presented framework is applicable for a large class of problems that are not limited to elasticity. An application to inelastic behavior such as viscoelasticity and plasticity will be studied in the future. Also an extension to other target functions, such as distributions of tractions, stresses or dissipated energies are topics which deserve further attention.

Declaration of competing interest

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

Acknowledgments

This work was supported by Fraunhofer Cluster of Excellence Programmable Materials (CPM).

C. Eberl partially acknowledges funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC-2193/1 – 390951807.

Appendix A. Adjoint solution approach

In order to satisfy the constraint condition in the optimization problem (5) we follow the adjoint solution approach. At first we consider the Lagrangian for a different residual functional $\bar{\mathcal{R}}$:

$$\mathcal{L}(\hat{\mathbf{u}}, \hat{\lambda}, \boldsymbol{\alpha}) = \mathcal{J}(\hat{\mathbf{u}}, \boldsymbol{\alpha}) + \left\langle \hat{\lambda}, \bar{\mathcal{R}}(\hat{\mathbf{u}}, \boldsymbol{\alpha}) \right\rangle_{V''} . \tag{A.1}$$

The Lagrange multiplier $\hat{\lambda}$ is also called the dual variable to the primal variable $\hat{\mathbf{u}}$. Now we return to the enforcement of the constraint in variational form from (13): We assume that $\bar{\mathcal{R}}(\hat{\mathbf{u}}, \boldsymbol{\alpha})$ has an identical form as in (12), but now is a linear operator w.r.t. $\hat{\mathbf{u}}$, hence the corresponding $\bar{C}(\hat{\mathbf{u}}, \boldsymbol{\alpha})$ is linear, too. In particular, we define $\bar{C}(\hat{\mathbf{u}}, \boldsymbol{\alpha})$ as:

$$\left\langle \bar{C}(\hat{\mathbf{u}}, \boldsymbol{\alpha}), \hat{\mathbf{v}} \right\rangle := \int_{\hat{\Omega}} \nabla \hat{\mathbf{u}} \cdot \mathbb{C}(\boldsymbol{\alpha}) \cdot \nabla \hat{\mathbf{v}} \, \mathrm{d}\mathbf{X} \tag{A.2}$$

where $\mathbb{C}(\boldsymbol{\alpha})$ is a symmetrical, fourth order tensor and a function of the design $\boldsymbol{\alpha}$. Hence, $\bar{\mathcal{R}}(\hat{\mathbf{u}}, \boldsymbol{\alpha}) \in V'$ (and also $\bar{C}(\hat{\mathbf{u}}, \boldsymbol{\alpha}) \in V'$), which is the dual space of V, the space of all linear operators from V to the real numbers. Therefore, the dual $\hat{\lambda}$ is an element of the bidual space V'' and $\langle \hat{\lambda}, \cdot \rangle_{V''}$ denotes the application of $\hat{\lambda}$ on V'.

Primal and dual variable form a saddle point $(\hat{\mathbf{u}}_s, \hat{\lambda}_s)$, if $\hat{\mathbf{u}}_s$ is the minimizer and $\hat{\lambda}_s$ the maximizer of the Lagrange functional $\mathcal{L}$, or if they fulfill the necessary condition for extrema:

$$\begin{aligned} \left\langle \partial_{\hat{\mathbf{u}}} \mathcal{L}(\hat{\mathbf{u}}_s, \hat{\lambda}_s, \boldsymbol{\alpha}), \hat{\mathbf{v}} \right\rangle_{V'} &= 0 \quad \forall \hat{\mathbf{v}} \in V, \\ \left\langle \partial_{\hat{\lambda}} \mathcal{L}(\hat{\mathbf{u}}_s, \hat{\lambda}_s, \boldsymbol{\alpha}), \hat{\boldsymbol{\vartheta}} \right\rangle_{V'''} &= 0 \quad \forall \hat{\boldsymbol{\vartheta}} \in V''. \end{aligned} \tag{A.3}$$

Note, that we pursue a simple presentation of the adjoint solution method. Therefore we reduce notations and skip variational terms as often as possible in the following in order to avoid complicated terms. For a more mathematically detailed deduction we refer to Frei et al. (2013), Hinze et al. (2009) and Michaleris et al. (1994).

It is known that there exists an isometry between V and V'' and that V is a reflexive Hilbert space. From that fact it can be shown that the second equation in (A.3) is identical to the equilibrium equation in (13), if we replace $\mathcal{R}$ with $\bar{\mathcal{R}}$ there. Therefore only the first equation in (A.3) leads us to a solution for the multiplier via:

$$0 = \partial_{\hat{\mathbf{u}}} \mathcal{L}(\hat{\mathbf{u}}_s, \hat{\lambda}_s, \boldsymbol{\alpha}) = \partial_{\hat{\mathbf{u}}} \mathcal{J}(\hat{\mathbf{u}}_s, \boldsymbol{\alpha}) + \left\langle \hat{\lambda}_s, \partial_{\hat{\mathbf{u}}} \bar{\mathcal{R}}(\hat{\mathbf{u}}_s, \boldsymbol{\alpha}) \right\rangle_{V''} \tag{A.4}$$

which is called dual or adjoint problem. Note that the differentiated terms have to be understood in variational sense. Then, using the definition from (A.2) the adjoint problem can be written as:

Find $\hat{\lambda}_s \in V$, such that

$$\begin{aligned} 0 &= \left\langle \partial_{\hat{\mathbf{u}}} \mathcal{L}(\hat{\mathbf{u}}_s, \hat{\lambda}_s, \boldsymbol{\alpha}), \hat{\mathbf{v}} \right\rangle_{V'} \\ &= \langle \partial_{\hat{\mathbf{u}}} \mathcal{J}(\hat{\mathbf{u}}_s, \boldsymbol{\alpha}), \hat{\mathbf{v}} \rangle_{V'} + \left\langle \hat{\lambda}_s, \bar{C}(\hat{\mathbf{v}}, \boldsymbol{\alpha}) \right\rangle_{V''} \\ &= \langle \partial_{\hat{\mathbf{u}}} \mathcal{J}(\hat{\mathbf{u}}_s, \boldsymbol{\alpha}), \hat{\mathbf{v}} \rangle_{V'} + \left\langle \bar{C}^{*}(\hat{\lambda}_s, \boldsymbol{\alpha}), \hat{\mathbf{v}} \right\rangle_{V'} \quad \forall \hat{\mathbf{v}} \in V, \end{aligned} \tag{A.5}$$

where $\bar{C}^{*}$ denotes the adjoint of $\bar{C}$. Note that the derivative $\partial_{\hat{\mathbf{u}}} \bar{\mathcal{R}}$ in (A.4) reduces to $\bar{C}$ since $\bar{\mathcal{R}}$ is linear. Besides, we may take $\hat{\lambda}_s \in V$, since V is reflexive. Furthermore, from (A.2) it is clear that $\bar{C}$ is self-adjoint, such that $\bar{C}^{*} = \bar{C}$ in the last term.

Now, the saddle point simplifies the computation of the total derivative of the cost functional $D_{\boldsymbol{\alpha}} \check{\mathcal{J}}$ for the scheme in (14). Note that the primal variable of the saddle point $\hat{\mathbf{u}}_s(\boldsymbol{\alpha})$ satisfies the constraint condition in (5), too. Hence, we can conclude:

$$\begin{aligned} D_{\boldsymbol{\alpha}} \check{\mathcal{J}}(\boldsymbol{\alpha}) &= D_{\boldsymbol{\alpha}} \mathcal{L}(\hat{\mathbf{u}}_s(\boldsymbol{\alpha}), \hat{\lambda}_s, \boldsymbol{\alpha}) \\ &= \partial_{\boldsymbol{\alpha}} \mathcal{L}(\hat{\mathbf{u}}_s(\boldsymbol{\alpha}), \hat{\lambda}_s, \boldsymbol{\alpha}) + \partial_{\hat{\mathbf{u}}} \mathcal{L}(\hat{\mathbf{u}}_s(\boldsymbol{\alpha}), \hat{\lambda}_s, \boldsymbol{\alpha}) \partial_{\boldsymbol{\alpha}} \hat{\mathbf{u}}_s(\boldsymbol{\alpha}) \\ &\overset{(A.4)}{=} \partial_{\boldsymbol{\alpha}} \mathcal{L}(\hat{\mathbf{u}}_s(\boldsymbol{\alpha}), \hat{\lambda}_s, \boldsymbol{\alpha}) \\ &= \partial_{\boldsymbol{\alpha}} \mathcal{J}(\hat{\mathbf{u}}_s(\boldsymbol{\alpha}), \boldsymbol{\alpha}) + \partial_{\boldsymbol{\alpha}} \bar{C}(\hat{\mathbf{u}}_s, \hat{\lambda}_s, \boldsymbol{\alpha}). \end{aligned} \tag{A.6}$$

The derivative $\partial_{\boldsymbol{\alpha}} \bar{C}$ is also self-adjoint because of $\bar{C}$. Note that all equations in (A.6) are considered in variational sense.

Algorithm 2 Precomputing the data base and TRD

```
Initialize data tensor $\mathbf{D} \in \mathbb{R}^{m_1 \times \ldots \times m_M}$
for $k = 1, \ldots, M_{\boldsymbol{\alpha}}$ do                ▷ Set up of the design variants
    Define $\mathcal{I}_k$
    for $i_k \in \mathcal{I}_k$ do
        Define the discrete design parameter $\boldsymbol{\alpha}_{i_k}$
    end for
end for
for $k = 1, \ldots, M_{\mathbf{E}}$ do                ▷ Set up of the strain variants
    Define $\mathcal{J}_k$
    for $j_k \in \mathcal{J}_k$ do
        Define the discrete strain value $\hat{\mathbf{E}}_{j_k}$
    end for
end for
for $I \in \mathcal{I}$ do
    $(i_1, \ldots, i_{M_{\boldsymbol{\alpha}}}, j_1, \ldots, j_{M_{\mathbf{E}}}) \leftarrow I$
    Process configuration $(\boldsymbol{\alpha}_{i_1}, \ldots, \boldsymbol{\alpha}_{i_{M_{\boldsymbol{\alpha}}}}, \hat{\mathbf{E}}_{j_1}, \ldots, \hat{\mathbf{E}}_{j_{M_{\mathbf{E}}}})$:
        Solve the microscopic BVP in (9) to obtain $\hat{\mathbf{S}}_l$        ▷ $\hat{\mathbf{S}}_l$ can be any entry of $\hat{\mathbf{S}}$
    Insert in the data tensor: $\mathbf{D}_I \leftarrow \hat{\mathbf{S}}_l$
end for
$\left[D_k^r\right]_{r=1,\ldots,R,k=1,\ldots,M} \leftarrow \mathrm{TRD}(\mathbf{D})$                ▷ TRD from (28)
return $\left[D_k^r\right]_{r=1,\ldots,R,k=1,\ldots,M}$
```

Algorithm 3 SVI from (32)

```
Required: Configuration $(\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_{M_{\boldsymbol{\alpha}}}, \hat{\mathbf{E}}_1, \ldots, \hat{\mathbf{E}}_{M_{\mathbf{E}}})$
$\bar{\mathbf{D}} \leftarrow 0$
for $r = 1, \ldots, R$ do                ▷ In this loop $N_l$ are interpolating basis functions
    $\bar{\mathbf{D}}_r \leftarrow 1$
    for $k = 1, \ldots, M_{\boldsymbol{\alpha}}$ do
        $\bar{\mathbf{D}}_r \leftarrow \bar{\mathbf{D}}_r * \sum_{l=1}^{m_k} (D_k^r)_l \; N_l(\boldsymbol{\alpha}_k)$        ▷ Here: $m_k = |\mathcal{I}_k|$
    end for
    for $k = 1, \ldots, M_{\mathbf{E}}$ do
        $\bar{\mathbf{D}}_r \leftarrow \bar{\mathbf{D}}_r * \sum_{l=1}^{m_k} (D_k^r)_l \; N_l(\hat{\mathbf{E}}_k)$        ▷ Here: $m_k = |\mathcal{J}_k|$
    end for
    $\bar{\mathbf{D}} \leftarrow \bar{\mathbf{D}} + \bar{\mathbf{D}}_r$
end for
return $\bar{\mathbf{D}}$
```

Appendix B. Differentiation of interpolating functions

In Section 2.4.2 we explain how the SVI allows to easily interpolate and differentiate at data points, within the structured data hypercube. Note that differentiations w.r.t. design variables (here $\boldsymbol{\alpha}$) are often difficult to compute for problems dealing with topology or shape optimization. A great advantage of our approach is that these derivatives can now be computed by differentiating common interpolating functions. For instance, if we want to assemble the design derivative of the stiffness tensor $\partial_{\boldsymbol{\alpha}} \hat{\mathbb{C}}$ in (19), we differentiate the interpolating function in (32). Assuming we sampled the ith entry of the stress tensor $\hat{\mathbf{S}}$ (in Voigt notation) for the data tensor $\mathbf{D}$, the differentiation of the following entry of $\mathbb{C}$ is straightforward:

$$\frac{\partial \hat{\mathbb{C}}_{ij}}{\partial \boldsymbol{\alpha}_n} = \frac{\partial^2 \hat{\mathbf{S}}_i}{\partial \boldsymbol{\alpha}_n \partial \hat{\mathbf{E}}_j} \approx \frac{\partial^2 \bar{\mathbf{D}}}{\partial \boldsymbol{\alpha}_n \partial \hat{\mathbf{E}}_j} = \sum_{r=1}^{R} \left(\prod_{\substack{k=1,\\ k \neq n}}^{M_{\boldsymbol{\alpha}}} \bar{D}_k^r \prod_{\substack{l=1,\\ l \neq j}}^{M_{\mathbf{E}}} \bar{C}_l^r \right) \frac{\partial \bar{D}_n^r}{\partial \boldsymbol{\alpha}_n} \frac{\partial \bar{C}_j^r}{\partial \hat{\mathbf{E}}_j} \tag{B.1}$$

where $n = 1, \ldots, M_{\boldsymbol{\alpha}}$ and $i, j = 1, \ldots, M_{\mathbf{E}}$. If only the entry $\mathbb{C}_{ij}$ is desired, the differentiation w.r.t. $\boldsymbol{\alpha}_n$ is simply skipped. Continuing this procedure for all entries of $\hat{\mathbf{S}}$, $\boldsymbol{\alpha}$ and $\mathbf{E}$, we get the whole tensor $\partial_{\boldsymbol{\alpha}} \hat{\mathbb{C}}$. Recall from (19) that this tensor is important for our optimization framework.

Appendix C. Algorithms

See Algorithms 1, 2 and 3.

References

Allaire, G., Bonnetier, E., Francfort, G., Jouve, F., 1997. Shape Optimization by the Homogenization Method, Vol. 76. (1), Springer Science and Business Media LLC, pp. 27–68. http://dx.doi.org/10.1007/s002110050253.

Andersen, G., Cowan, D., Piatak, D., 2007. Aeroelastic modeling, analysis and testing of a morphing wing structure. In: 48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference. American Institute of Aeronautics and Astronautics, http://dx.doi.org/10.2514/6.2007-1734.

Aubin, C.A., Gorissen, B., Milana, E., Buskohl, P.R., Lazarus, N., Slipher, G.A., Keplinger, C., Bongard, J., Iida, F., Lewis, J.A., Shepherd, R.F., 2022. Towards enduring autonomous robots via embodied energy. Nature 602 (7897), 393–402. http://dx.doi.org/10.1038/s41586-021-04138-2.

Awad, L.N., Bae, J., O'Donnell, K., De Rossi, S.M., Hendron, K., Sloot, L.H., Kudzia, P., Allen, S., Holt, K.G., Ellis, T.D., Walsh, C.J., 2017. A soft robotic exosuit improves walking in patients after stroke. Sci. Transl. Med. 9 (400), http://dx.doi.org/10.1126/scitranslmed.aai9084.

Ballester-Ripoll, R., Paredes, E.G., Pajarola, R., 2016. A Surrogate Visualization Model Using the Tensor Train Format. In: SA '16, Association for Computing Machinery, New York, NY, USA, http://dx.doi.org/10.1145/3002151.3002167.

Bandler, J., Biernacki, R., Chen, S.H., Grobelny, P., Hemmers, R., 1994. Space mapping technique for electromagnetic optimization. IEEE Trans. Microw. Theory Tech. 42 (12), 2536–2544. http://dx.doi.org/10.1109/22.339794.

Bendsøe, M.P., Kikuchi, N., 1988. Generating optimal topologies in structural design using a homogenization method. Comput. Methods Appl. Mech. Engrg. 71 (2), 197–224. http://dx.doi.org/10.1016/0045-7825(88)90086-2.

Bendsøe, M.P., Sigmund, O., 2004. Typology Optimization Ebook. Springer, Berlin, Heidelberg, p. 381, URL https://link.springer.com/book/10.1007/978-3-662-05086-6{#}toc.

Booker, A.J., Dennis, J.E., Frank, P.D., Serafini, D.B., Torczon, V., Trosset, M.W., 1999. A rigorous framework for optimization of expensive functions by surrogates. Struct. Optim. 17 (1), 1–13. http://dx.doi.org/10.1007/bf01197708.

de Boor, C., 1978. A Practical Guide To Spline, Vol, Volume 27. http://dx.doi.org/10.2307/2006241.

2021. CALCULIX - a free software three-dimensional structural finite element program. http://www.calculix.de/, Accessed: 2021-10-08.

Chen, W., Tong, L., Liu, S., 2016. Design of periodic unit cell in cellular materials with extreme properties using topology optimization. Proc. Inst. Mech. Eng. L 232 (10), 852–869. http://dx.doi.org/10.1177/1464420716652638.

Chinesta, F., Leygue, A., Bordeu, F., Aguado, J.V., Cueto, E., Gonzalez, D., Alfaro, I., Ammar, A., Huerta, A., 2013. PGD-based computational vademecum for efficient design, optimization and control. Arch. Comput. Methods Eng. 20, http://dx.doi.org/10.1007/s11831-013-9080-x.

Daynes, S., Weaver, P.M., Potter, K.D., 2009. Aeroelastic study of bistable composite airfoils. J. Aircr. 46 (6), 2169–2174. http://dx.doi.org/10.2514/1.44287.

Ferrer, A., Cante, J., Hernández, J., Oliver, J., 2018. Two-scale topology optimization in computational material design: An integrated approach. Internat. J. Numer. Methods Engrg. 114 (3), 232–254. http://dx.doi.org/10.1002/nme.5742, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.5742.

Fratzl, P., Jacobs, K., Möller, M., Scheibel, T., Sternberg, K., 2019. Materials Research: Inspired by Nature Innovation. acatech DISCUSSION.

Frei, S., Andrä, H., Pinnau, R., Tse, O., 2013. Optimizing fiber orientation in fiber-reinforced materials using efficient upscaling. Comput. Optim. Appl. 62 (1), 111–129. http://dx.doi.org/10.1007/s10589-013-9630-z.

Geuzaine, C., Remacle, J.-F., 2009. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. Internat. J. Numer. Methods Engrg. 79 (11), 1309–1331. http://dx.doi.org/10.1002/nme.2579.

2021. GOM correlate pro: Strains and displacements from motion pictures. https://www.gom.com/en/products/gom-suite/gom-correlate-pro, Accessed: 2021-10-08.

Grant, D.T., Abdulrahim, M., Lind, R., 2010. Design and analysis of biomimetic joints for morphing of micro air vehicles. Bioinspir. Biomim. 5 (4), 045007. http://dx.doi.org/10.1088/1748-3182/5/4/045007.

Han, Y., Lu, W., 2018. Evolutionary design of nonuniform cellular structures with optimized Poisson's ratio distribution. Mater. Des. 141, 384–394. http://dx.doi.org/10.1016/j.matdes.2017.12.047.

Hill, R., 1972. On constitutive macro-variables for heterogeneous solids at finite strain. Proc. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci. 326 (1565), 131–147. http://dx.doi.org/10.1098/rspa.1972.0001.

Hinze, M., Pinnau, R., Ulbrich, M., Ulbrich, S., 2009. Optimization with PDE constraints. In: Mathematical Modelling: Theory and Applications, Vol. 23. Springer, New York.

Hirschler, T., Antolin, P., Buffa, A., 2021. Fast and multiscale formation of isogeometric matrices of microstructured geometric models. arXiv:2107.09568.

Imediegwu, C., Murphy, R., Hewson, R., Santer, M., 2019. Multiscale structural optimization towards three-dimensional printable structures. Struct. Multidiscip. Optim. 60 (2), 513–525. http://dx.doi.org/10.1007/s00158-019-02220-y.

Le, B.A., Yvonnet, J., He, Q.-C., 2015. Computational homogenization of nonlinear elastic materials using neural networks. Internat. J. Numer. Methods Engrg. 104 (12), 1061–1084. http://dx.doi.org/10.1002/nme.4953.

Maconachie, T., Leary, M., Lozanovski, B., Zhang, X., Qian, M., Faruque, O., Brandt, M., 2019. SLM lattice structures: Properties, performance, applications and challenges. Mater. Des. 183, 108137. http://dx.doi.org/10.1016/j.matdes.2019.108137.

Mark, A.G., Palagi, S., Qiu, T., Fischer, P., 2016. Auxetic metamaterial simplifies soft robot design. IEEE, pp. 4951–4956. http://dx.doi.org/10.1109/ICRA.2016.7487701.

Michaleris, P., Tortorelli, D.A., Vidal, C.A., 1994. Tangent operators and design sensitivity formulations for transient non-linear coupled problems with applications to elastoplasticity. Internat. J. Numer. Methods Engrg. 37 (14), 2471–2499. http://dx.doi.org/10.1002/nme.1620371408, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/nme.1620371408, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.1620371408.

Mirzaali, M., Janbaz, S., Strano, M., 2018. Shape-matching soft mechanical metamaterials. Sci. Rep. 8, http://dx.doi.org/10.1038/s41598-018-19381-3.

Ogden, R., 1984. Non-linear elastic deformations. Eng. Anal. 1 (2), 119. http://dx.doi.org/10.1016/0264-682X(84)90061-3, URL https://linkinghub.elsevier.com/retrieve/pii/0264682X84900613.

Orlik, J., Panasenko, G., Shiryaev, V., 2016. Optimization of textile-like materials via homogenization and beam approximations. Multiscale Model. Simul. 14 (2), 637–667. http://dx.doi.org/10.1137/15M1017193.

Oseledets, I.V., 2011. Tensor-train decomposition. SIAM J. Sci. Comput. 33 (5), 2295–2317. http://dx.doi.org/10.1137/090752286.

2022. ProgMatSim: Software tool for the simulation and optimization of programmable materials. https://websites.fraunhofer.de/progmatcode, Accessed: 2022-02-22.

2021. Programmable shape changing and mechanics. https://cpm.fraunhofer.de/en/main_topics/programmable_shape_changing_mechanics.html, Accessed: 2021-10-08.

Savostyanov, D., Oseledets, I., 2011. Fast adaptive interpolation of multi-dimensional arrays in tensor train format. In: The 2011 International Workshop on Multidimensional (ND) Systems. pp. 1–8. http://dx.doi.org/10.1109/nDS.2011.6076873.

Schenk, M., Guest, S.D., McShane, G.J., 2014. Novel stacked folded cores for blast-resistant sandwich beams. Int. J. Solids Struct. 51, 4196–4214. http://dx.doi.org/10.1016/j.ijsolstr.2014.07.027.

Sethian, J.A., Wiegmann, A., 2000. Structural boundary design via level set and immersed interface methods. J. Comput. Phys. 163 (2), 489–528. http://dx.doi.org/10.1006/jcph.2000.6581.

Stanford, B., Ifju, P., 2008. Aeroelastic topology optimization of membrane structures for micro air vehicles. Struct. Multidiscip. Optim. 38 (3), 301–316. http://dx.doi.org/10.1007/s00158-008-0292-x.

Tamara, B.W.B., Kolda, G., 2009. Tensor decompositions and applications. SIAM 51 (3), 455–500. http://dx.doi.org/10.1137/07070111X.

Wang, C., Gu, X., Zhu, J., Zhou, H., Li, S., Zhang, W., 2020. Concurrent design of hierarchical structures with three-dimensional parameterized lattice microstructures for additive manufacturing. Struct. Multidiscip. Optim. 61 (3), 869–894. http://dx.doi.org/10.1007/s00158-019-02408-2.

Weisheit, L., Wenz, F., Eckert, M., Lichti, T., Baumann, S., Hübner, C., Eberl, C., Andrä, H., 2020. Domänenübergreifende workflows zur effizienten Entwicklung Programmierbarer Materialien. Z. Wirtsch. Fabrikbetr. 115 (7–8), 470–475. http://dx.doi.org/10.3139/104.112353.

Wenz, F., Schmitt, I., Leichner, A., Lichti, T., Baumann, S., Andrä, H., Eberl, C., 2021. Designing shape morphing behavior through local programming of mechanical metamaterials. Adv. Mater. http://dx.doi.org/10.1002/adma.202008617.

White, D.A., Arrighi, W.J., Kudo, J., Watts, S.E., 2019. Multiscale topology optimization using neural network surrogate models. Comput. Methods Appl. Mech. Engrg. 346, 1118–1135. http://dx.doi.org/10.1016/j.cma.2018.09.007.

Wu, J., Sigmund, O., Groen, J.P., 2021. Topology optimization of multi-scale structures: a review. Struct. Multidiscip. Optim. 63, 1455–1480. http://dx.doi.org/10.1007/s00158-021-02881-8.

Yvonnet, J., Gonzalez, D., He, Q.-C., 2009. Numerically explicit potentials for the homogenization of nonlinear elastic heterogeneous materials. Comput. Methods Appl. Mech. Engrg. 198 (33), 2723–2737. http://dx.doi.org/10.1016/j.cma.2009.03.017, URL https://www.sciencedirect.com/science/article/pii/S0045782509001492.

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

Zheng, L., Kumar, S., Kochmann, D.M., 2021. Data-driven topology optimization of spinodoid metamaterials with seamlessly tunable anisotropy. Comput. Methods Appl. Mech. Engrg. 383, 113894. http://dx.doi.org/10.1016/j.cma.2021.113894, arXiv:2012.15744.