Computational Mechanics manuscript No.
(will be inserted by the editor)

The Parametrized Superelement approach for lattice joint modelling and simulation

T. De Weer · N. Vannieuwenhoven · N. Lammens · K. Meerbergen

Received: date / Revised version: date

Abstract From large scale structures such as bridges and cranes to small-scale structures such as lattices, mechanical structures today increasingly consist of "tuneable parts": parts with a simple geometry described by a small set of strongly varying parameters. For example, although a bridge is a macroscale structure with a complex geometry, it is made from simple beams, bolts and plates, all of which depend on spatially varying, geometrical parameters such as their length-to-width ratio. Accelerating this trend is the concurrent improvement of, on one hand, Additive Manufacturing techniques that allow for increasingly complex parts to be manufactured, and on the other hand structural optimization techniques that exploit this expanded design space. However, this trend also poses a challenge to current simulation techniques, as for example the Finite Element Method requires large amounts of elements to represent such structures. We propose to exploit the large conformity between parts inside the mechanical structure through construction of semi-analytic "Parametrized Superelements", built by meshing with solid elements, reduction to a fixed interface and approximation of the reduced stiffness matrices. These new elements can be employed next to standard elements, enabling the simulation of more complex geometries. The technique we propose is applied to lattice structures and provides a flexible, differentiable, more accurate but still efficient way to simulate their elastic response. A net gain in total computation time occurs after simulating more than twenty lattices. As such, the proposed method enables large-scale parameter exploration and optimization of lattice structures for improved energy absorption, mass and/or stiffness.

Keywords parametrized · superelement · lattice · additive · manufacturing

Tom De Weer
Siemens Industry Software NV, Interleuvenlaan 68, 3001 Leuven, Belgium
KU Leuven, Department of Computer Science, Celestijnenlaan 200A, B-3000 Leuven, Belgium
E-mail: tom.deweer@siemens.com

Nick Vannieuwenhoven
KU Leuven, Department of Computer Science, Celestijnenlaan 200A, B-3000 Leuven, Belgium
Leuven.AI, KU Leuven Institute for AI, B-3000 Leuven, Belgium.
E-mail: nick.vannieuwenhoven@kuleuven.be

Nicolas Lammens
Siemens Industry Software NV, Interleuvenlaan 68, 3001 Leuven, Belgium
E-mail: nicolas.lammens@siemens.com

Karl Meerbergen
KU Leuven, Department of Computer Science, Celestijnenlaan 200A, B-3000 Leuven, Belgium
Leuven.AI, KU Leuven Institute for AI, B-3000 Leuven, Belgium.
E-mail: karl.meerbergen@kuleuven.be

Acknowledgements Tom De Weer acknowledges Vlaams Agentschap Innoveren en Ondernemen (VLAIO)) for funding the Baekeland Research Project HBC.2019.2165 "Topology

Optimization of 3D Printed Lattice Structures for Structural Applications".
Nick Vannieuwenhoven was supported by a Postdoctoral Fellowship of the Research Foundation—Flanders (FWO) with project number 12E8119N; Interne Fondsen KU Leuven/Internal Funds KU Leuven with project number C16/21/00; and Junior Onderzoeksproject of the Research Foundation-Flanders (FWO) with project number G080822N.
Karl Meerbergen was supported by Interne Fondsen KU Leuven/Internal Funds KU Leuven.

1 Introduction

Lattices are defined by Ashby [2] as structures composed of struts connected at joints. However, in contrast to the macroscale frames encountered in bridges and cranes, the lattices considered in this paper exist on a smaller scale. They can therefore be viewed as materials themselves and their effective properties, tuneable by changing their topological layout, cover a wide range among a multitude of physical applications, such as acoustic [6], fatigue [21] and energy absorption [35]. In this paper, the mechanical properties of lattices are studied. The geometric complexity of lattices poses a challenge for their construction. Whereas large truss structures can be built with traditional manufacturing techniques such as milling and welding, lattices almost always require Additive Manufacturing (AM) [23]. Even with proper tuning of the AM process parameters, defects can occur that negatively impact the mechanical performance [42]. Nonetheless, the inherent topological complexity is also an opportunity for optimization. Although structural optimization dates back to at least 1904, when Michell [24] published his work on optimal frame structures, and Topology Optimization (TO) was invented by Bendsøe and Sigmund [3, 4] in the 1980s, the last decade has seen a renewed interest in this area. The area of multiscale topology optimization in particular, sparked by improved AM capabilities, has received more attention since optimizing structures at multiple scales can only improve their performance compared to mono-scale structures [38]. Although multiscale implies optimization of a structure's layout at multiple scales, most often only two are used: the macroscale and the microscale. Although many different algorithms exist, the resulting designs often look similar to lattices on every scale: sets of beams connected at joints. Some algorithms therefore preemptively constrain their microstructural layout to consist only of lattice structures [18, 25, 40]. Furthermore, for compliance optimization the theoretically optimal structure, a rank-N laminate, could be viewed as a lattice that exists on multiple scales [12, 38]. The combination of multiscale TO, AM and lattices leads to structural designs existing on multiple scales and containing intricate lattices whose topology changes throughout the design (see Fig. 1). Accurate and efficient simulation of such lattices is thus paramount.

Current state-of-the-art lattice simulation techniques, further elaborated in §2.1, are based on other solid or beam elements. Although a solid element mesh is accurate when it has converged, an enormous computational effort is required to do so. In contrast, a beam element represents a beam with only twelve degrees of freedom (dof) and thus requires orders of magnitude less computation time. However, beam elements are unable to represent the strut joints, i.e. the regions where struts meet. Although certain modifications exist in literature (see §2.1) the underlying problem is not solved, making them at best only accurate for a small set of joints. This paper proposes the use of a so-called "joint element", in combination with the standard beam element, to keep the computational efficiency of beam elements while improving their accuracy.

The main challenge is that joints have too complex a geometry (see §3) to derive an analytic joint element. The joint is therefore meshed with solid elements and a "dual form" of the standard straight face constraints (see §2.3) is applied numerically, after which static condensation (see §2.2) leads to a "joint superelement". Unfortunately, this "direct" approach (see §4) is too computationally expensive since static condensation is a costly procedure that must be repeated for *every* joint inside a lattice (of which there can easily be thousands, see again Fig. 1). We overcome this obstacle by constructing a more efficient but sufficiently accurate approximation based on samples, thus trading the online cost for an offline cost and enabling efficient simulation of a wide variety of lattices. This "approximate" approach (see §5), reduces a variety of joints to their reduced representation, constructs an approximation of that representation based on a tensor decomposition and then expands the approximated, reduced representation to create an analytical and efficient representation of the joint elements. The re-

sults in Section 5.4 show that our proposed method decreases the online cost by a few orders of magnitude while retaining the accuracy of the direct method, at the price of a larger offline cost.
Another illustration of the proposed lattice simulation is presented in §6 where we apply it to a lattice with increasing relative density. The accuracy is shown to be much closer to that of the solid element mesh rather than a beam element mesh, while the online cost is reduced from almost half an hour to forty seconds, representing a speedup factor of 45. The offline cost is roughly the time needed to simulate twenty lattices with a solid element mesh. Thus, a net gain in computation time is observed when simulating more than twenty lattices, e.g. during optimization, or when lattices become so large that using solid elements requires too much computer memory. Finally, §7 draws a conclusion and specifies directions for future work.

2 Background

Three research fields related to the developed method are now discussed. First, §2.1 discusses advantages and disadvantages of the current techniques used for simulating lattices that exist on a single scale. Next, §2.2 discusses how lattices are simulated when they fit inside a larger-scale structure. Finally, §2.3 refreshes beam theory since the PSE method developed here is strongly tied to it and in particular the employed face constraints.

2.1 Mono-scale lattice structure simulation

Lattice structure behaviour under mechanical load is usually simulated with the Finite Element Method (FEM), although no standard meshing approach exists specifically for lattices. Two element types are generally used for lattices in literature: three-dimensional solid elements and one-dimensional beam elements [16].

Their difference lies in the trade-off between accuracy and efficiency. Solid element meshes can accurately capture the mechanical response but require the mesh to be sufficiently fine, leading to high computation times. On the other hand, beam element models require less computational resources: a single beam can be modelled with an element based on beam theory using only twelve dof. However, such models suffer from a diminished accuracy for two main reasons. First, the assumption that planes perpendicular to the beam axis remain planes (see beam theory in §2.3), breaks down for stout struts, i.e. beams with a low length-to-width ratio. Second, and more important, beam elements alone cannot capture the joints, i.e. the locations where beams are connected [9, 10].

The effects of these joints on the mechanical behaviour of lattices is significant for two reasons. First and foremost, even for slender struts modelled accurately by beam theory, joints may significantly contribute to the overall stiffness. This effect increases sharply with the beam radius and is only amplified in AM. Second, in bending-dominated structures (e.g. the diamond lattices in this paper) high stresses occur in the joints, which greatly influence the fatigue life [9, 21].

Since the breakdown of beam element accuracy is inherent to the use of beam elements, research has recently focused on incorporating the joint stiffness into the beam elements. Labeas and Sunaric [20] increase the diameter by 40% at the first and last 10% of every beam. In a paper by Smith et al. [32] follow a similar approach but different values are used. Luxner et al. [22] increase the beam stiffness a 1000 times in a spherical region around the strut joint. Finally, Dong and Zhao [9] use an optimization algorithm to choose the parameters of their proposed "joint stiffening element": another beam element with adaptations at the beam ends.

Although all the above approaches report reasonable accuracy with good efficiency, they lack in flexibility and extensibility to deal with various lattice structures. For example, the joint stiffening element developed by Dong and Zhao is optimized for joints inside Vintiles lattices but is not guaranteed to work for other lattice types. Furthermore, adapting beam elements to incorporate the joint stiffness hides the underlying issue: the lack of proper element to represent joints, something also mentioned by Helou and Kara [16]. The method presented in this paper proposes a generic way to create such a "joint element" and use it inside lattice simulation frameworks.



(a) Bracket made for the ALCOA/GrabCAD challenge [15], from Erdelyi et al. [11].



(b) Optimized conforming lattice structure, reprinted with permission from Wu et al. [39].



(c) Forth bridge, located west of Edinburgh, Scotland.



(d) Inner truss structures of the Forth bridge.

Fig. 1: Mechanical structures that exist on multiple scales. Top: AM structures with large amounts of intricate lattices. Bottom: Forth bridge consisting of truss beams. Pictures taken from Wikipedia [13, 19].

2.2 Multiscale lattice structure simulation

Although the previous section showed that simulating lattices on a single scale is already computationally expensive, lattices are often fitted inside larger structures (see again Fig. 1). A short overview of two simulation methods for such multiscale structures is now given.

A first, prevalent method for multiscale simulation is numerical homogenization. Through the assumption of a separation of scale, detailed microscale structures can be viewed from the macroscale as materials with effective properties determined by their microstructural layout. For lattices made by repetition of a unit cell, computation of the homogenized properties can occur with reasonable accuracy on a single unit cell when Periodic Boundary Conditions are used, provided the scale separation assumption holds. The lattice can then be replaced with a simple iso-, ortho- or anisotropic material, greatly reducing the computational effort at the macroscale [41, 45]. However, the simulation of lattices still suffers from the same efficiency-versus-accuracy issues discussed in the previous section. A recent technique, called FE^2, relies on the same homogenization assumptions but uses an iterative procedure [41]. Instead of extracting homogenized material properties, macroscopic strains are applied to a microscale structure. The resulting microscopical stress is then computed and the average is used to update the macroscopic equilibrium. Although the iterative procedure improves the multiscale accuracy, the numerical efficiency is further reduced.

A second multiscale simulation approach is the use of superelements or substructuring. As a standard technique in FEM this is included in commercial packages such as Simcenter Nastran [30]. In substructuring, a structure is meshed and the resulting elements are grouped into superelements connected by their exterior dof. These superelements are then reduced to

their exterior dof via static condensation, also known as Guyan reduction. In this sense, superelements are Reduced Order Models (ROM). As noted by Romeo and Schulz [29], the repetitive nature of metamaterials made from reoccuring unit cells can be exploited with this approach, since in practice only a single reduction must be performed. However, when a unit cell changes throughout the structure, many superelements must be constructed and the high computational cost associated with this procedure quickly exceeds the gains. To alleviate this problem, the literature employs Parametric Reduced Order Models (PROM): ROM that are allowed to vary in a select set of parameters [1, 26, 34]. For example, Amsallem et al. [1] use a PROM approach for aeroelastic modelling of fighter jets with a varying Mach number. These models are usually created by sampling the (possible high-dimensional) parameter space with a high-fidelity model, reducing these samples to a ROM and fitting a function through the reduced matrices.

Three main challenges faced by literature on PROM are now discussed. First, in order to allow fitting a function through the space of reduced matrices, they must be compatible, i.e. have the same size. For example, Panzer et al. propose standard techniques for model order reduction and then introduce novel *projection methods* to ensure the reduced models are compatible [26]. Second, for the specific case of approximating stiffness and mass matrices it is essential that the PROM method conserves key properties of these matrices such a symmetry and positive semi-definiteness. As an example, Speet [34] uses a projection to the manifold of positive definite, symmetric matrices introduced by Pennec et al. [27] to ensure property conservation, whereas Xu et al. [43] use a Cholesky factorization to ensure symmetry and positive-definiteness of their stiffness tensors modelled by neural networks. Third, the curse of dimensionality is a severe hurdle when constructing a PROM. Consider as a first example the Approximation of Reduced Substructure with Penalization (ARSP) method developed by Wu et al. [40]. Here, a two-dimensional cubic microlattice is condensed to its exterior dof and the resulting stiffness matrices are approximated by creating a basis for the reduced stiffness matrix space with the proper orthogonal decomposition (POD) and approximating the coordinates in that basis with polynomial, multi-dimensional basis functions (although in practice only one parameter, the density, is used). As a second example, the approach by Imediegwu et al. [18] is considered. Here, they simulate a large number of lattice structures with varying beam radii through meshing with hexahedral elements. Next, these complex models are reduced with homogenization and approximated using least squares with global polynomial basis functions (up to degree six). Due to the high number of parameters (six in the case of Imediegwu et al.), the curse of dimensionality hampers the approximation efforts. For example, Imediegwu's global basis functions exist in a six-dimensional space, but a simple second-order polynomial in such a space already requires optimization of $\binom{6+2}{2} = 28$ coefficients[1]. The approximation techniques employed by Wu et al. [40] and Imediegwu et al. [18] thus become intractable for higher dimensionalities.

The approach described in this paper could be classified as PROM. First, the compatibility problem is solved by using straight face constraints, leading to a fixed and reusable interface. Second, the property conservation issue is solved by three additional reduction steps that ensure symmetry, positive semi-definiteness and the correct rigid body modes of the approximated stiffness matrix. Finally, we circumvent the curse of dimensionality by employing the canonical polyadic decomposition (CPD) from Hitchcock [17].

2.3 Basics of beam theory

Beams are widely used in architectural and mechanical designs, leading to a vast literature describing their mechanical properties: beam theory. Discretization of beam theory leads to one-dimensional beam elements which, as discussed in §2.1, can be used to simulate lattices. A short overview, without going into much detail about different beam theories (e.g. Timoshenko versus Euler beam theory), is given here to highlight the main assumptions and simplifications. For more information, the reader is referred to these recent review papers [8, 28].

Assuming linear elasticity holds, mechanical structures are modelled by three general equations: the equi-

[1] One constant term coefficient, 6 linear term coefficients and 21 quadratic term coefficients.

librium equations, the strain-displacement relations and the constitutive equations. For isotropic materials, these can be combined into the Navier-Cauchy displacement formulation

$$(\lambda + \mu)\nabla(\nabla \cdot \mathbf{u}) + \mu\nabla^2\mathbf{u} + \mathbf{F} = 0, \tag{1}$$

where λ and μ are Lamé parameters, $\mathbf{F}$ and $\mathbf{u}$ are force and displacement vectors in $\mathbb{R}^3$, respectively, ∇ is the Nabla operator and $\cdot$ is the dot operator. In combination with proper boundary conditions this is a well-posed problem often solved with FEM since analytical solutions for general structures are scarce. In contrast, a key property of beams, namely their high aspect ratio (the ratio between their length and width), allows for an approximate differential equation with solutions that are in some cases even analytically computable. For example, when forces are applied as point forces at the end of a beam, Timoshenko beam theory predicts widely known second-order polynomial solutions for the beam displacement.

Consider at a beam boundary face Γ the resultant force $\mathbf{f}_\Gamma \in \mathbb{R}^3$ and moment $\mathbf{m}_\Gamma \in \mathbb{R}^3$ of the applied traction $\mathbf{t} \in \mathbb{R}^3$, defined as averages over the area and implicitly assuming the variation over the small width is limited:

$$\mathbf{f}_\Gamma = \int_A \mathbf{t}\,\mathrm{d}A \tag{2}$$

and

$$\mathbf{m}_\Gamma = \int_A \mathbf{t} \times (\mathbf{p} - \mathbf{p}_C)\,\mathrm{d}A, \tag{3}$$

where $\times$ is the cross product and A and $\mathbf{p}_C$ are the area and the centroid of the boundary face Γ, respectively. They can be thought to stem from a first-order approximation of the traction vector $\mathbf{t}_\mathbf{n}$, where $\mathbf{n}$ denotes the beam plane normal,

$$\mathbf{t}_\mathbf{n} \approx \mathbf{t}_f + \mathbf{t}_m \times (\mathbf{p} - \mathbf{p}_C), \tag{4}$$

although the traction vector is allowed to differ from this linear approximation (hence the $\approx$). Here, $\mathbf{t}_f = \frac{\mathbf{f}_\Gamma}{A}$ is the traction component due to the force and $\mathbf{t}_m = \hat{\mathbf{J}}\mathbf{m}_\Gamma$ is the traction component due to the moment. Here, $\hat{\mathbf{J}}$ is a matrix containing second moments of area defined in Appendix A.

A kinematic constraint is placed on the boundary displacement $\mathbf{u}_\Gamma$ to ensure planes perpendicular to the axis remain so after deformation. This constraint, referred to as the straight face constraint, reduces the dof to a translation $\mathbf{u}_C$ and a rotation θ_C:

$$\mathbf{u}_\Gamma = \mathbf{u}_C + \theta_C \times (\mathbf{p} - \mathbf{p}_C). \tag{5}$$

Through this constraint, the Navier-Cauchy equation can be reduced to a differential equation relating the beam deflection to the forces and moments applied at the beam ends.

3 Problem statement and methodology

This section lays the groundwork for the introduction of methods discussed in the next two sections by defining the lattice simulation problem. Two topics are discussed. First, a geometrical description of the considered lattices and joints is given. Second, a general overview of the lattice simulation methodology and the idea of a "joint element" is properly introduced.

3.1 Lattice geometry

In this paper, lattices are made of beams connected at joints and can therefore be considered truss structures similar to those seen in cranes and bridges. Although any configuration of beams is possible, this paper restricts itself to lattices built by repeating a unit cell topology. Fig. 3 shows four common configurations: a cubic (Fig. 3a), a body-centered cubic (BCC, Fig. 3b), a diamond (Fig. 3c) and an octet (Fig. 3d) unit cell. A lattice containing a BCC unit cell repeated three times in every direction is from now on called a $3\times3\times3$ BCC lattice. This paper focuses on diamond lattices containing multiple unit cells. The beam radii are allowed to change throughout the structure, hence the focus on multiple unit cells instead of one.
Geometrically, we represent lattice beams by cylinders and the joints by spheres with one trimmed circular surface for every connecting beam. The radii of these "joint spheres" are chosen such that the beams remain cylindrical, i.e. such that the trimfaces belonging to different beams do not overlap. To emphasize this, the goal of the joints is to ensure the beam geometry is well-defined and can be captured by beam theory. An

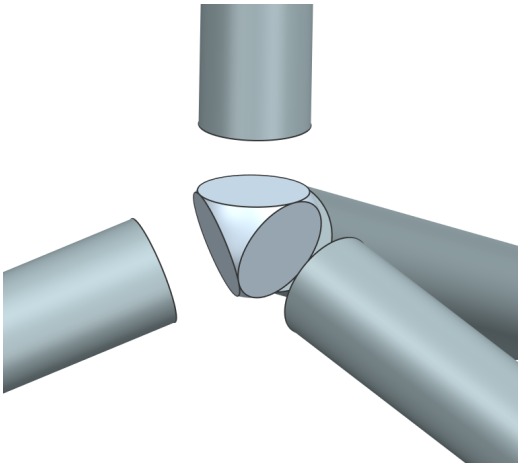
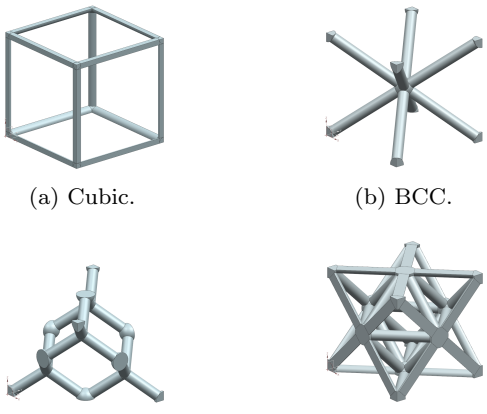


Fig. 2: Exploded view of a joint inside a diamond lattice. Due to the geometry of the joint, the beams touch only at a single point and remain cylindrical. Visualization with Simcenter 3D [31].



(a) Cubic. (b) BCC. (c) Diamond. (d) Octet.

Fig. 3: Common lattice configurations.

exploded-view CAD representation of a joint with four connecting beams is shown in Fig. 2. Note that, once a unit cell type like the diamond is chosen, the angles between the beams are not allowed to change. Only the beam radii can change during for example a parameter exploration or an optimization procedure.

For ease of programming, we restrict lattices in this paper to exist in a hexagonal bounding box. The parts of the joints and beams that lie outside the bounding box are thus trimmed away. Beams on a boundary face or edge are thus cut in two or four, respectively. Similarly, joints on a boundary face, edge or corner are cut in two, four and eight, respectively. Finally, the radii of boundary joints are slightly larger than internal joints to ensure the beams stay cylindrical. Fig. 4 shows a $2 \times 2 \times 2$ diamond lattice containing all four types of joints that can occur in a diamond lattice.

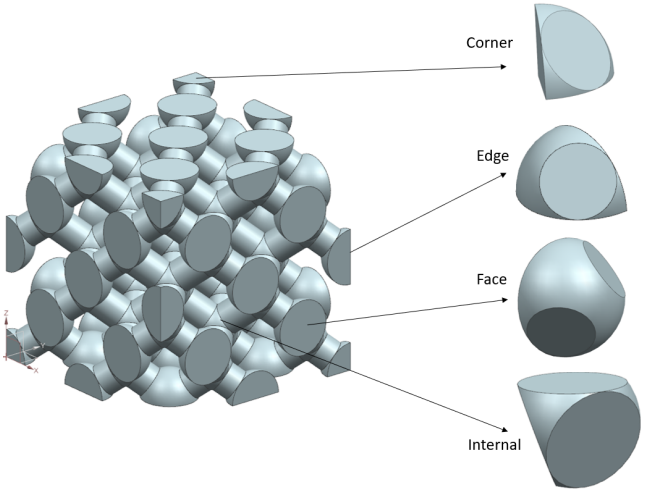


Fig. 4: Different types of lattice joints inside a $2 \times 2 \times 2$ diamond lattice. The joint radii are relatively large with respect to the beam radii to highlight the joint geometries. Visualization with Simcenter 3D [31].

Note that other types of bounding boxes can be chosen and only affect the exterior of the lattice structure. In practice, lattices are fitted inside more difficult bounding boxes, leading to more difficult geometries for the boundary joints. The hexagonal choice is made here for simplicity and to focus on the internal lattice joints. Combining two lattices together causes the face joints at the connection face to become internal, although numerically the applied straight face assumption will make these joints stiffer than standard internal joints.

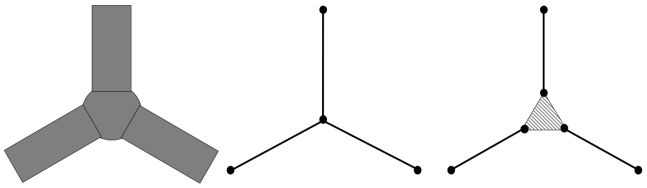


Fig. 5: Left: CAD representation of a 2D lattice joint. Middle: pure beam element mesh. Right: mesh containing shortened beam elements and a "joint element".

3.2 General lattice simulation methodology

As introduced in §2.1 and experimentally shown in §6, lattice structures are simulated with either solid elements, achieving high resolution and accuracy at the cost of low numerical efficiency, or with beam elements, easing the numerical cost but greatly reducing the accuracy. The main issue with the beam element model is not that beam theory is inaccurate but rather the lack of joint modeling. To incorporate the joint stiffness, which becomes increasingly important when a beam's radius increases relative to its length, some modifications to standard beam elements have been proposed (see again §2.1). However, these modifications are quite arbitrary and not generally applicable.

In the proposed methodology, the beam element mesh is changed in two ways. First, the beam elements are shortened to prevent them from touching in a single point. Instead, the beam elements now reflect the beam geometry. Second, a "joint element" is created to represent the joints that connect the shortened beams. A sketch of the proposed change is shown in Fig. 5.

A key element of the joint element that sets it apart from previous literature, is its predefined interface to the exterior. Whereas the techniques described by Romeo and Schulz [29] and Wu et al. [40] reduce their substructures to an exterior number of dof that depends on the meshing, here this interface is fixed, leading to more reusable superelements.

In particular, the interface is as follows. Each joint element has one node with six dof for each external face, i.e. for each connected beam face and each boundary face. The six dof (three translational and three rotational) describe the displacement (translation and rotation) of the external face. Internal joints in a diamond lattice (i.e. those that are not trimmed away by the bounding box, see Fig. 4), have four beam faces and are thus represented by a joint element with four nodes. The corner joint has one beam face and three boundary faces and thus also has four nodes. The edge and face joints have three external faces in total and their joint elements contain three nodes. All nodes are placed at the centroid of the external face they represent.

In order to generically use a joint element in a Finite Element (FE) program, only the joint stiffness matrix $\mathbf{K}_j \in \mathbb{R}^{6n_f \times 6n_f}$ is required. Here, n_f is the number of nodes of the joint element, representing the n_f external faces of the joint it represents. The six degrees of freedom are divided into a translation $\mathbf{u}_f$ and a rotation θ_f. Translational stiffness coefficients relate forces to translations and thus have units N/m. Similarly, rotational coefficients relate moments to rotations and thus have units Nm/rad = Nm. Finally, "mixed" coefficients relate forces to rotations and moments to translations and have units N.

4 Direct construction of a joint element

This section deals with the direct construction method for the joint element introduced in the previous section. Constructing a joint element similar to building beam elements, e.g. through discretization of beam theory, is hampered due to the absence of a corresponding "joint theory". Furthermore, constructing such an analytical joint theory from scratch by starting from similar assumptions is impeded due to its complex geometry: whereas a beam's one-dimensional nature allows for easy construction of shape functions (see e.g. the Euler-Bernoulli beam theory simplification of $u_x = -z\theta$ if x is the axial direction), finding proper shape functions for three-dimensional lattice joints is remarkably harder. A first alternative is now proposed, which is from now on referred to as the Direct Parametrized Superelement (D-PSE) approach. First, §4.1 explains the four-step methodology of the D-PSE method. Section 4.2 then discusses some implementation details that ensure the efficiency and accuracy of the method. The discretization error is then studied in §4.3 by means of a mesh convergence experiment. This section also discusses in more detail the discretization error on the rigid body modes, which (being the eigenvectors for the zero eigenvalues of the stiffness matrix $\mathbf{K}_j$), have the

largest effect on the mechanical response. Finally, §4.4 discusses the D-PSE method in general, highlighting its main advantages and disadvantages and providing arguments for the method explained in the next section, the Approximate Parameterized Superelement (A-PSE) method.

4.1 Methodology

A D-PSE stiffness matrix is constructed in four steps, as visualized in Fig. 6. First, the joint geometry is modelled with a mesh of solid elements, for a total of N_d internal dof. Here, internal denotes the fact that these degrees of freedom are not directly connected to other beams or joints. The internal stiffness matrix $\mathbf{K}_{int} \in \mathbb{R}^{N_d \times N_d}$ is constructed such that $\mathbf{K}_{int}\mathbf{u}_{int} = \mathbf{f}_{int}$. Here, $\mathbf{u}_{int} \in \mathbb{R}^{N_d}$ and $\mathbf{f}_{int} \in \mathbb{R}^{N_d}$ are the displacement and force vector, respectively. The internal force vector $\mathbf{f}_{int}$ is kept for completeness but is zero for this application. Second, the n_f nodes of the joint element are added at the centroids of the boundary faces for a total of $6n_f$ exterior dof, which are captured in the external displacement vector $\mathbf{u}_{ext}$:

$$\mathbf{u}_{ext}^T = \begin{bmatrix} \mathbf{u}_1^T & \theta_1^T & \dots & \mathbf{u}_{n_f}^T & \theta_{n_f}^T \end{bmatrix}.$$

Since the external dof are (for now) disconnected, the mesh is represented by the extended system of equations

$$\begin{bmatrix} \mathbf{K}_{int} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{u}_{int} \\ \mathbf{u}_{ext} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_{int} \\ \mathbf{f}_{ext} \end{bmatrix}, \tag{6}$$

or, more briefly, by $\mathbf{K}_e\mathbf{u}_e = \mathbf{f_e}$. Here, $\mathbf{f}_{ext}$ is the external force vector containing the forces and moments applied by the surrounding structure.

Third, a dual form of the straight face constraints is applied to connect the interior and exterior dof. Whereas standard beam theory (see §2.3) applies "primal" straight face constraints through Eq. 5, thus fixing every point on the face rigidly to the exterior dof and disallowing any warping, dual straight face constraints only enforce the faces to be straight "on average", resulting in a more relaxed constraint. Furthermore, whereas beam theory finds the resultant forces and moments through averaging of the traction vector via Eqs. 2 and 3, the dual form intrinsically requires the traction vector to be a linear, "straight" function. The relation between the primal and dual form is further visualized in Fig. 7. The primal versus dual terminology is adopted here to highlight the similar yet mirrored assumptions and relations found by the two formulations and not because of a Lagrange multiplier derivation.

Consider a joint with n_f external faces. Each external face has a translation vector $\mathbf{u}_C$ and a rotation vector θ_C, where the subscript C refers to the centroid. The following two dual assumptions are made. First, the traction vector on a face is now rigidly constrained to be a linear function:

$$\mathbf{t_n} = \mathbf{t_f} + \mathbf{t_m} \times (\mathbf{p} - \mathbf{p}_C). \tag{7}$$

Second, the face displacement is constrained to be a linear function, but only approximately:

$$\mathbf{u} \approx \mathbf{u}_C + \theta_C \times (\mathbf{p} - \mathbf{p}_C). \tag{8}$$

Here, $\mathbf{u}_C \in \mathbb{R}^3$ and $\theta_C \in \mathbb{R}^3$ are the face's dual displacement and rotation vectors, respectively. Note the difference with standard beam theory (Eqs. 4 and 5), where the equals and approximation sign are switched.

The approximation is now enforced through the zeroth-order average $\frac{1}{A}\int_A \cdot \, \mathrm{d}A$ and the first-order average $\int_A \cdot \times \mathbf{r}_C \, \mathrm{d}A$:

$$\frac{1}{A}\int_A \mathbf{u} \, \mathrm{d}A = \frac{1}{A}\int_A \mathbf{u}_C \, \mathrm{d}A + \frac{1}{A}\int_A \theta_C \times \mathbf{r}_C \, \mathrm{d}A, \tag{9}$$

$$\int_A \mathbf{u} \times \mathbf{r}_C \, \mathrm{d}A = \int_A \mathbf{u}_C \times \mathbf{r}_C \, \mathrm{d}A + \int_A (\theta_C \times \mathbf{r}_C) \times \mathbf{r}_C \, \mathrm{d}A. \tag{10}$$

Here, $\mathbf{r}_C$ denotes $\mathbf{p} - \mathbf{p}_C$. Elaborating leads to expressions for the translational and rotational dof of the face:

$$\mathbf{u}_C = \frac{1}{A}\int_A \mathbf{u} \, \mathrm{d}A, \tag{11}$$

$$\hat{\mathbf{J}}\theta_C = \int_A \mathbf{u} \times \mathbf{r}_C \, \mathrm{d}A. \tag{12}$$

Here, $\hat{\mathbf{J}}$ is a matrix containing second moments of area and is defined in Appendix A. This appendix also provides an expression for the applied force and moment as a function of the unknown terms in Eq. 7.

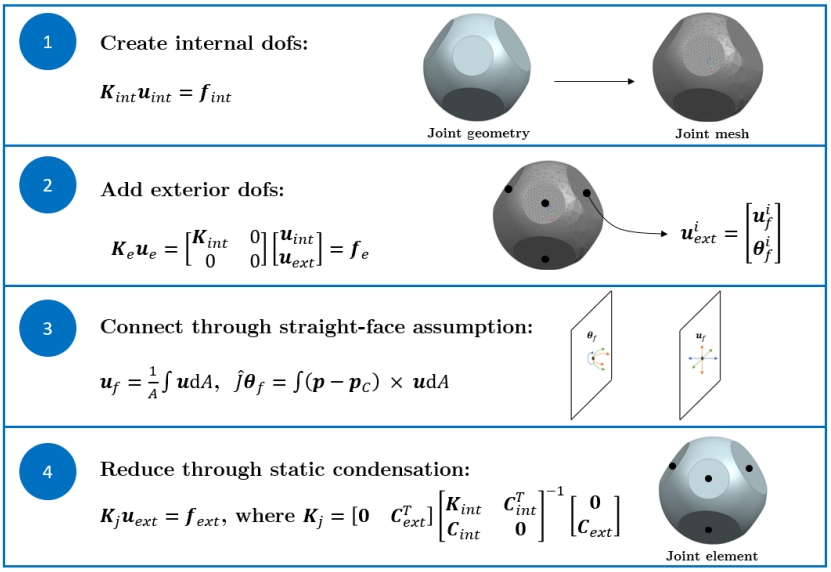


Fig. 6: Overview of the D-PSE approach for generating joint elements. Step 4 requires the solution of a large system of equations, leading to a large online cost.

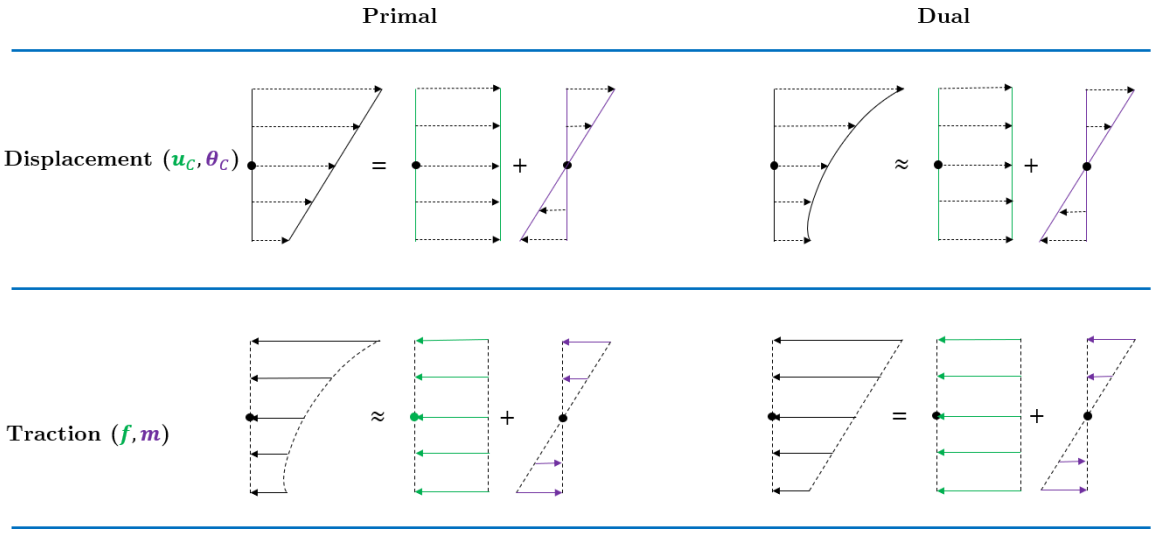


Fig. 7: Comparison of the dual and the primal straight face constraints for both displacements and traction.

In general, the averaged straight face constraints expressed by Eqs. 11 and 12 relate the interior and exterior dof and can be expressed in general matrix form by

$$\begin{bmatrix} \mathbf{C}_{int} & \mathbf{C}_{ext} \end{bmatrix} \begin{bmatrix} \mathbf{u}_{int} \\ \mathbf{u}_{ext} \end{bmatrix} = \mathbf{0}, \tag{13}$$

or, more succinctly, by $\mathbf{C}\mathbf{u}_e = \mathbf{0}$. Using the method of Lagrange multipliers, Eqs. 6 and 13 can be combined into the following representation of the joint element

$$\begin{bmatrix} \mathbf{K}_e & \mathbf{C}^T \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{u}_e \\ \mu \end{bmatrix} = \begin{bmatrix} \mathbf{f}_e \\ \mathbf{0} \end{bmatrix}. \tag{14}$$

Here, μ contains the Lagrange multiplier dof.

Fourth and final, the extended system of equations is reduced to the exterior dof $\mathbf{u_{ext}}$ via static condensation. Consider a system of equations containing unknowns that are kept, denoted with subscript "k", and unknowns that are omitted, denoted with subscript "o":

$$\begin{bmatrix}\mathbf{A}_{kk} & \mathbf{A}_{ko}\\ \mathbf{A}_{ok} & \mathbf{A}_{oo}\end{bmatrix}\begin{bmatrix}\mathbf{x}_k\\ \mathbf{x}_o\end{bmatrix}=\begin{bmatrix}\mathbf{b}_k\\ \mathbf{b}_o\end{bmatrix}.$$

This system can be expressed solely by its kept unknowns $\mathbf{x_k}$ assuming $\mathbf{A}_{oo}$ is invertible:

$$(\mathbf{A}_{kk}-\mathbf{A}_{ko}\mathbf{A}_{oo}^{-1}\mathbf{A}_{ok})\mathbf{x}_k=\mathbf{b}_k-\mathbf{A}_{ko}\mathbf{A}_{oo}^{-1}\mathbf{b}_o.$$

The reduced matrix, also known as its Schur complement, is thus equal to $\mathbf{A}_{kk}-\mathbf{A}_{ko}\mathbf{A}_{oo}^{-1}\mathbf{A}_{ok}$. Applying this technique to Eq. 14, the joint element stiffness matrix $\mathbf{K}_j$, defined as the relation between the exterior dof and forces $\mathbf{K}_j\mathbf{u}_{ext}=\mathbf{f}_{ext}$, is

$$\mathbf{K}_j=-\begin{bmatrix}\mathbf{0} & \mathbf{C}_{ext}^T\end{bmatrix}\begin{bmatrix}\mathbf{K}_{int} & \mathbf{C}_{int}^T\\ \mathbf{C}_{int} & \mathbf{0}\end{bmatrix}^{-1}\begin{bmatrix}\mathbf{0}\\ \mathbf{C}_{ext}\end{bmatrix}. \tag{15}$$

4.2 Implementation

Implementation details of the four steps of the D-PSE approach are now discussed. In general, the main lattice simulation framework is developed in Python (v3.8). For the first step, the joint geometry mesh is created by Gmsh v4.8.4 [14], producing a set of tetrahedra. These tetrahedra are converted to NX Nastran's second-order tetrahedral elements (CTETRA10) by adding nodes to every edge. The assembled stiffness matrix is generated with Nastran (available in Simcenter 3D [31]) and loaded into Python with the pyNastran module v1.3.
For the third step, i.e. the application of the dual straight face constraints of Eqs. 11 and 12, the integrals $\frac{1}{A}\int_A \mathbf{u}\,\mathrm{d}A$ and $\int_A \mathbf{r}\times\mathbf{u}\,\mathrm{d}A$ must be evaluated on n_f faces. In the FE formulation, the displacement field $\mathbf{u}$ on a face is expressed as $\mathbf{u}=\sum\mathbf{u}_i b_i$, where $\mathbf{u}_i$ and b_i are the nodal displacement and the basis function belonging to a node i and the summation goes over the nodes on the face. Elaboration leads to the evaluation of integrals $\int_{A_e} b_i\,\mathrm{d}A$ and $\int_{A_e} r_j b_i\,\mathrm{d}A$ over an element surface A_e, where r_j is the j-th component of $\mathbf{r}$. Since the elements are of second order, the first integral is integrated exactly using second-order Gauss integration. The second integral contains the nonlinear factor r_j and is integrated with a seventh-order Gauss integration scheme, which is found to be satisfactory in §4.3.
For the final step, i.e. the reduction of Eq. 14 to the joint element stiffness matrix $\mathbf{K}_j$, the computation of

$$\begin{bmatrix}\mathbf{K}_{int} & \mathbf{C}_{int}^T\\ \mathbf{C}_{int} & \mathbf{0}\end{bmatrix}^{-1}\begin{bmatrix}\mathbf{0}\\ \mathbf{C}_{ext}\end{bmatrix}$$

is carried out by sparse, parallel LU factorization and repeated backsolving of the multiple right-hand sides with Umfpack [7].

4.3 Mesh convergence and rigid body modes

To assess the correct implementation of the D-PSE method, two studies are now presented. First, a mesh convergence study is performed. Second, a more in-depth study of the rigid body modes is conducted.

Mesh convergence. As summarized in Fig. 8, the mesh convergence study is performed by repeatedly applying the D-PSE method to a joint with an increasing number of elements N_{el} (ranging from a few hundred to at most 100 000). The number of elements is converted to a representative average "element size" h through $h=\sqrt[3]{1/N_{el}}$. The difference between the successive joint element stiffness matrices $\tilde{\mathbf{K}}_j^i=\mathbf{K}_j(N_{el}^i)$ and the "exact" joint stiffness matrix $\mathbf{K}_j=\mathbf{K}_j(N_{el}=\infty)$ is then studied. Of course, since the exact solution is unknown the solution found on the finest mesh is used.

Fig. 9 studies the convergence of the stiffness matrix of an internal lattice joint when its mesh is refined from around five hundred elements to two hundred thousand. The relative error with respect to the finest mesh $\rho(h)=\left\|\tilde{\mathbf{K}}(h)-\mathbf{K}\right\|/\left\|\mathbf{K}\right\|$, where $\|\cdot\|$ denotes the 2-norm, is shown as a function of the element size h next to the orders of convergence $p=2$ and $p=3$ (i.e. $\rho(h)=\mathcal{O}(h^p)$). Clearly, the actual order of convergence lies somewhere between two and three. This is slightly lower than what's theoretically expected, since second-order tetrahedral elements permit an order of convergence of $p=3$ on the displacements [44].

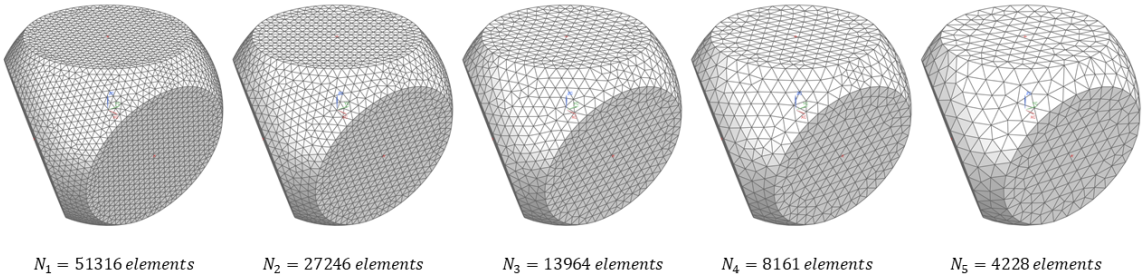


Fig. 8: Illustration of some of the meshes used for the mesh convergence. The mesh size varies from very fine on the left to very coarse on the right. Visualization with Simcenter 3D [31].

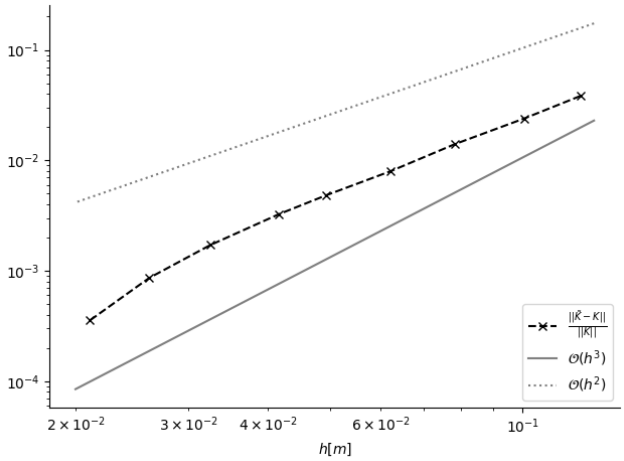


Fig. 9: Relative error convergence of an internal joint, compared to the solution found on the finest mesh contains around two hundred thousand elements.

Rigid body modes. From standard finite element theory it is known that an elemental stiffness matrix must be positive semi-definite, meaning all eigenvalues are nonnegative. The zero eigenvalues correspond to eigenvectors called rigid body modes, i.e. those displacements that do not induce any internal deformation and thus do not require any external work to achieve them. These modes are known a priori: for unconstrained three-dimensional structures there are six (three translations and three rotations) whereas two-dimensional structures have three (two translations and one rotation). More specifically, a three-dimensional rigid body displacement field $\tilde{\mathbf{u}}_{rb}(x, y, z)$ is defined as

$$\tilde{\mathbf{u}}_{rb}(x, y, z) = \mathbf{u}_{rb} + \theta_{rb} \times \mathbf{r}_{rb}, \tag{16}$$

where $\mathbf{u}_{rb}$, θ_{rb} and $\mathbf{r}_{rb}$ are the rigid body translation, rotation and position with respect to the rotation center, respectively. A rigid body mode is thus the sum of a rigid body translation $\mathbf{u}_{rb}$ and a rigid body rotation θ_{rb} around an arbitrary reference point $\mathbf{r}_{rb}$, since any change in $\mathbf{r}_{rb}$ can be viewed as a rigid body translation. Usually, the centroid of the mechanical structure is used as reference point.
Since the displacement of a mechanical structure is computed by solving the system $\mathbf{Ku} = \mathbf{f}$, the mechanical response for force-loaded structures is mainly determined by the lowest eigenvalues of the stiffness matrix. Since the rigid body modes correspond to the lowest possible eigenvalues, it is of utmost importance that these are approximated correctly. This is now shown theoretically.

To verify the theoretical adherence to the rigid body modes, one must ensure that applying such a displacement does not result in internal strain energy. For the standard finite elements used to create the internal stiffness matrix $\mathbf{K}_{int}$, this requirement is already known to hold[2]. Proving the adherence of a joint matrix to the zero-energy rule for rigid body modes therefore only requires showing that the dual straight face constraints captured in $\mathbf{C}_{ext}$ do not destroy the properties of $\mathbf{K}_{int}$. This is done by substituting Eq. 16 into the

[2] In fact, from FEM theory the opposite is known to be occur for some element types evaluated with reduced integration. Spurious zero-energy modes known as "hourglasses" are then observed.

dual straight face constraints of Eq. 11 and 12) and choosing $\mathbf{r}_{rb} = \mathbf{r}_C = \mathbf{r}$, leading to

$$\begin{aligned}\mathbf{u}_C &= \frac{1}{A}\int_A \mathbf{u}\,\mathrm{d}A = \frac{1}{A}\int_A \mathbf{u}_{rb}\,\mathrm{d}A + \frac{1}{A}\int_A \theta_{rb}\times\mathbf{r}\,\mathrm{d}A \\ &= \mathbf{u}_{rb}, \\ \hat{\mathbf{J}}\theta_C &= \int_A \mathbf{u}\times\mathbf{r}\,\mathrm{d}A = \int_A \mathbf{u}_{rb}\times\mathbf{r}\,\mathrm{d}A + \int_A (\theta_{rb}\times\mathbf{r})\times\mathbf{r}\,\mathrm{d}A \\ &= \hat{\mathbf{J}}\theta_{rb},\end{aligned}$$

where the terms $\frac{1}{A}\int_A \theta_{rb}\times\mathbf{r}\,\mathrm{d}A$ and $\int_A \mathbf{u}_{rb}\times\mathbf{r}\,\mathrm{d}A$ are dropped due to the definition of a centroid and where one can see that $\theta_C = \theta_{rb}$ because $\hat{\mathbf{J}}$ is invertible. Applying a rigid body displacement field to a face constrained with a dual straightness restriction thus leads to this face translating with $\mathbf{u}_{rb}$ and rotating with θ_{rb}, implying that no extra work is needed. This theoretically proves the adherence to the rigid body modes.

4.4 Discussion of the D-PSE approach

Although this direct, four-step approach could be seen as a variant of the superelement approach, what centrally distinguishes it from the superelement approach is the use of constraints to achieve a model order reduction. Compared to the standard primal formulation, the results obtained with the dual straight face constraints differ in two ways.

First, due to the averaging a looser constraint is applied than the rigid constraint of the primal formulation, leading to a lower overall stiffness of the joint. This replaces the inherent stiffness overestimation that occurs when using primal straight face constraints of standard beam theory with an underestimation.

Second, primal face constraints rigidly connect faces that share even just a single point, thus causing a singularity when computing the stiffness matrix (since technically some elements would become infinite). Such faces exist in joints belonging to the edge and the corner of a lattice (see Fig. 4). By contrast, the dual formulation produces proper, "finite" stiffness matrices when dealing with connected faces since the proposed displacement field is only enforced on average (see Eqs. 9 & 10).

Although the D-PSE method converges to the exact solution when the mesh is refined, the accuracy of the D-PSE method is somewhat lacking: a global relative error on the stiffness matrix of 1 to 0.1% is achieved for moderate element sizes. However, by construction the rigid body modes are respected up to a higher accuracy. The main accuracy bottleneck is the meshing error and the three-dimensional nature of this problem. Although one cannot get around the latter problem, the former certainly is improvable and left for future work.

The computational efficiency gain can be quite vast compared to standard FEM solutions. It is most notable when the structure contains N substructures but only $n \ll N$ are unique (up to translation and rotation), since then the D-PSE need only be applied n times and the remaining $N-n$ substructures are replaced by a translation/rotation of the first n. Examples of such structures are truss bridges and lattices with constant beam diameters (see Fig. 1). However, in a generic case where a lattice contains a large variety of beams and joints, every joint requires a different joint element. Since creating such a joint element with the direct approach requires the relatively expensive computation of the Schur complement in Eq. 15, a more efficient alternative is required. This is achieved through approximation in the next section, where the expensive direct computation is performed offline and online lattice simulation only uses an approximate joint stiffness matrix representation.

5 Joint element construction through reduction, approximation and expansion

Since the D-PSE method becomes computationally inefficient for lattices containing joints with varying configurations, an approximation of the space of joint stiffness matrices is now attempted. The proposed method, from now on referred to as the A-PSE approach, contains three steps also illustrated in Fig. 10.

1. Section 5.1 discusses the **reduction** step. This step is meant to prevent the approximation of trivial relations such as scaling, rotation and material

dependence and ensures approximated stiffness matrix retains important intrinsic properties such as symmetry and positive semi-definiteness.
2. Section 5.2 deals with the **approximation** step, in which the reduced stiffness matrix is first sampled (see §5.2.1) and then approximated with a tensor-based method (see §5.2.2).
3. Section 5.3 then discusses the **expansion** step, where the goal is to recover the stiffness matrix from its reduced, approximated representation.

Afterwards, §5.4 shows an example and discusses advantages and disadvantages.

5.1 Reduction

Consider a joint with n_f external faces, of which n_b are faces connected to beams, classified as a joint of type T. In the first step of the A-PSE method, the joint is **reduced** to its most bare representation: all information that need not be approximated is removed to obtain a reduced number of both input and output parameters. This is done both to reduce the required number of samples (and thus the computational load) of the subsequent approximation step as well as to prevent any unnecessary reconstruction errors in the expansion step. Indeed, if intrinsic properties such as the stiffness matrix's symmetry are not handled properly before approximation, then after approximation they will only hold up to the approximation error.

Two types of reduction are used: those on the input or sample space are discussed in §5.1.1 and those on the output or stiffness matrix space are discussed in §5.1.2.

5.1.1 Sample space reduction

Two sample space restrictions are employed.
First, the material dependence is removed. Although a lattice joint can be made of any material, only joints made of a "standard" isotropic material with a Young's modulus of $1N/m^2$ and a Poisson ratio of $\nu = 0.3$ are used as training data for the approximation. The Young's modulus can change during the expansion step (see §5.3) and the Poisson ratio should be kept as an input parameter but is for simplicity kept fixed in this paper.

Second, the orientation and the size of the joint is removed as a parameter. Instead, for every joint type T a "standard" configuration in space is chosen, since joints with another configuration can be brought to this standard configuration by means of a linear transformation (see again §5.3). This configuration can be chosen arbitrarily. In this paper, the joints are restricted to have a unit radius and the normal vectors of the beam faces are chosen to point along positive unit directions as much as possible. Since some joint types contain inherent symmetries, symmetry conditions are also applied. As an example, consider the 2D joint with 2 beams shown in Fig. 11. Approximating all possible combinations of r_1 and r_2 is unnecessary since any combination where $r_1 \geq r_2$ can be transformed into a situation where $r_1 \leq r_2$ by means of a reflection around $\mathbf{v}_3 = \begin{bmatrix} 1 & 0.5 \end{bmatrix}^T$. Therefore, only combinations where the symmetry condition $r_1 \geq r_2$ holds are considered. Finally, the radii are also chosen to prevent overlap as illustrated in Fig. 11: beams can touch in at most one point since they would otherwise not remain cylindrical.[3]

As a result of the restrictions of the sample space, the only remaining input parameters that describe a joint of type T are the beam radii $r_i, i = 1, \ldots, n_b$, henceforth also referred to as the features or parameters. They belong to the feature space $\mathcal{F}_T$, which Fig. 12 illustrates for the 2D example of Fig. 11.

5.1.2 Joint stiffness matrix reduction

Three properties of a (joint) stiffness matrix $\mathbf{K}_j$ are enforced by reducing it to a minimal representation.
First, symmetry is considered. Since applying a force and measuring the displacement is (physically) equal to applying that displacement and measuring the force, stiffness matrices are symmetric.[4] Therefore, only the lower triangular part is considered.
Second, as discussed in §4.3, stiffness matrices have a null space corresponding to the rigid body modes.

[3] Note however that this is an upper bound: beams are not constrained to touch in a single point. They can also not touch at all, although in practice joints are constructed such that at least one beam pair is touching.

[4] Mathematically, this stems from the Navier equation (Eq. 1) being self-adjoint.

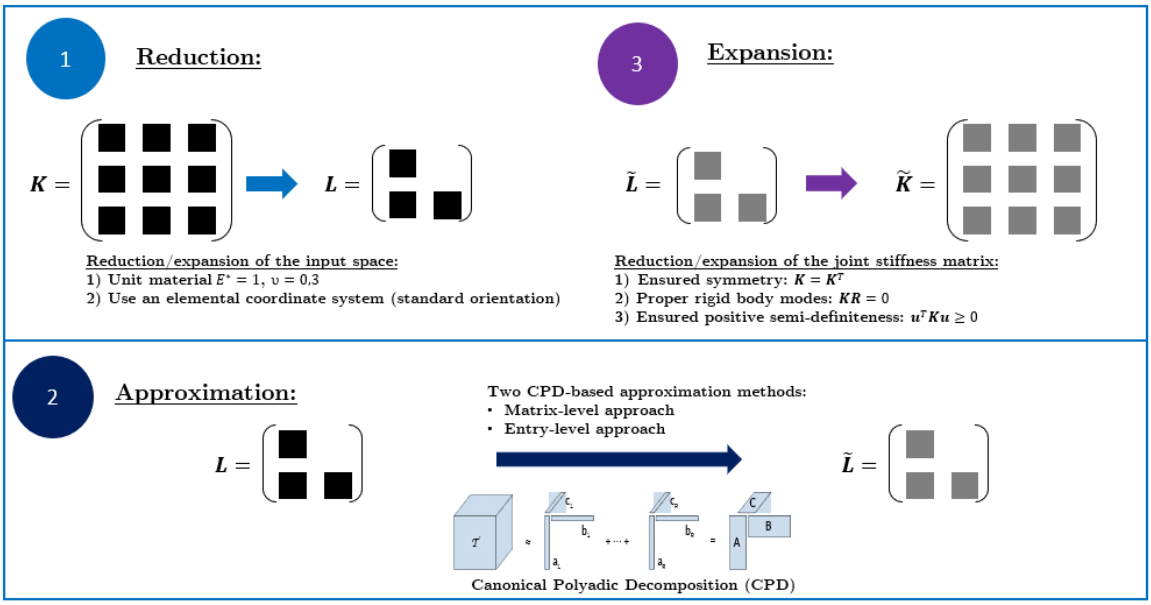


Fig. 10: Overview of the proposed approach. Since the direct joint computation is replaced by an approximation that is computed offline, the online computational cost is much lower.

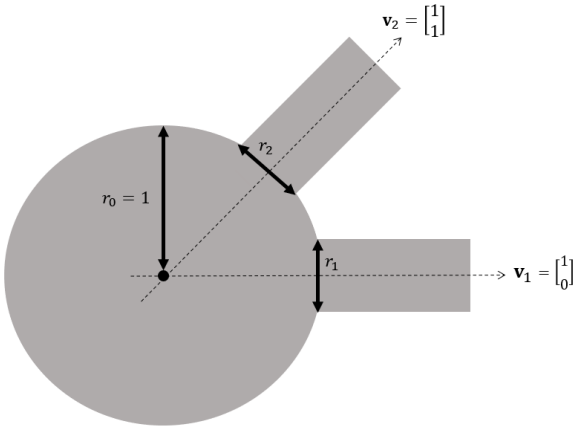


Fig. 11: Illustration of the standard configuration for a 2D joint with 2 beam orientations $\mathbf{v}_1$ and $\mathbf{v}_2$.

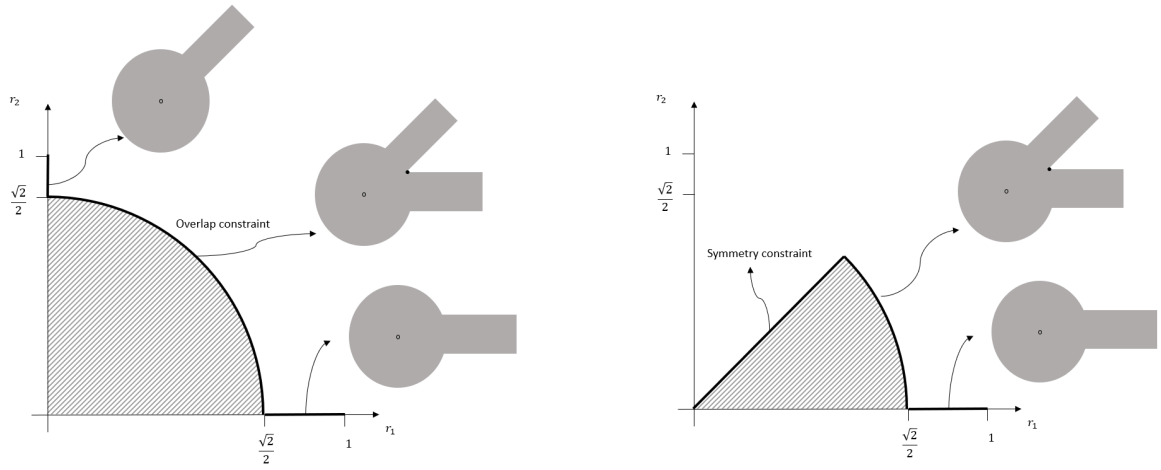
Since these modes are known beforehand, they are used to further reduce the joint matrix.

Consider the six rigid body modes of a joint stiffness matrix $\mathbf{K}_j \in \mathbb{R}^{N\times N}$ captured by $\mathbf{R} \in \mathbb{R}^{N\times 6}$ such that every column of $\mathbf{R}$ contains the translations and rotations of the faces under a linearly independent rigid body transformation (see Eq. 16). By definition, $\mathbf{K}_j\mathbf{R} = \mathbf{0}$ and by splitting the degrees of freedom into six dependent degrees of freedom (subscript d) and the remaining independent ones (subscript i), one gets

$$\mathbf{K}_j\mathbf{R} = \begin{bmatrix} \mathbf{K}_{ii} & \mathbf{K}_{id} \\ \mathbf{K}_{di} & \mathbf{K}_{dd} \end{bmatrix} \begin{bmatrix} \mathbf{R}_i \\ \mathbf{R}_d \end{bmatrix} = \mathbf{0}. \tag{17}$$

As will be shown in §5.3.1, reconstruction of $\mathbf{K}_j$ only requires the independent stiffness matrix $\mathbf{K}_{ii}$ and the a priori known rigid body modes $\mathbf{R}$. Therefore, only $\mathbf{K}_{ii}$ must be approximated. Note that $\mathbf{K}_{ii}$ is still symmetric but has six zero eigenvalues less than $\mathbf{K}_j$. Remaining zero eigenvalues are caused by one or more of the beam radii being zero, since this means six dofs become disconnected. Reconstruction is discussed in §5.3.1).

A third, key property is positive semi-definiteness (PSD) of $\mathbf{K}_j$. Whereas symmetry requires $\mathbf{K}_j$ to have real eigenvalues and rigid body modes force $\mathbf{K}_j$ to have six zero eigenvalues, PSD requires all eigenvalues to be nonnegative. Note that PSD also still holds for $\mathbf{K}_{ii}$. Physically, this means a positive force cannot result in a negative displacement. It is enforced with Xu et al.'s approach [43] by noting that there exists a Cholesky decomposition $\mathbf{K}_{ii} = \mathbf{L}\mathbf{L}^T$, where $\mathbf{L} \in \mathbb{R}^{n_r\times n_r}$ is lower



(a) Feature space without symmetry constraint.

(b) Feature space with the symmetry constraint $r_1 \geq r_2$.

Fig. 12: Illustration of the 2D feature space (r_1, r_2) for the joint of Fig. 11, showing both the overlap and the symmetry constraints.

triangular and real. Approximating $\mathbf{L}$ instead of $\mathbf{K}_{ii}$ then ensures PSD. Although this approach works well for positive definite matrices, in the edge case of positive *semi*-definiteness the Cholesky decomposition is not unique[5], introducing unnecessary approximation error to Xu et al.'s approach. This occurs when one of the beam radii is zero, thus causing six zero rows and columns and hence six zero eigenvalues. In this paper, this numerical stability issue is solved by employing Tikhonov regularization: a diagonal term $\epsilon\mathbf{I}$ is added to $\mathbf{K}_{ii}$ before Cholesky decomposition, thus removing the zero eigenvalues and ensuring its uniqueness. By choosing ϵ larger than the machine epsilon but smaller than the expected approximation error, accuracy is retained.

In conclusion, one needs to approximate the map from the feature space $\mathcal{F}_T$, represented by feature samples $r_i, i = 1, \ldots, n_b$, to the space of joint stiffness matrices $\mathcal{J}_T$, represented by the elements in the lower-triangular matrix $\mathbf{L}$, where $\mathbf{K}_{ii} = \mathbf{L}\mathbf{L}^T$ and $\mathbf{K}_{ii}$ is a subblock of the joint stiffness matrix containing the independent degrees of freedom stemming from the rigid body modes.

[5] Unless pivoting is used to somehow fix the location of the zero eigenvalues.

5.2 Approximation

In this section, the method for approximating the reduced joint representation $\mathbf{L}$, a lower-triangular matrix of size $6(n_f - 1) \times 6(n_f - 1)$, is explained. It consists of a method to sample the feature $\mathcal{F}_T$ using the D-PSE method, discussed in §5.2.1, and a method to build an approximation based on these samples in §5.2.2. Although matrix approximation is a straightforward concept, the multidimensional nature of this application poses practical problems. Both subsections therefore only give a general overview of the method and refer to the appendix for more detailed information.

5.2.1 Sampling the feature space

Reduction is followed by a sampling of the feature space $\mathcal{F}_T$ to generate training and test data for the subsequent approximation step. Here, data must be understood as a set of feature samples, i.e. tuples $(r_1, \ldots, r_{n_b}) = (f_1, \ldots, f_{n_b})$, where the features correspond to external face radii $0 < r_i < 1$ of a joint in standard configuration. The training samples, of which usually a few hundred are constructed, are used as input for the approximation step, whereas the test samples are meant to gauge the approximation error and prevent both over- and underfitting. Due to the use of a tensor-based approximation method, the

training data is generated on a multidimensional grid. The details of the sampling algorithm are explained in Appendix B.
As an example of the sampling algorithm's results, Figure 13 visualizes the distribution of the training and test samples for the internal diamond lattice joint. Each sample is converted to a joint element with the D-PSE method, after which the resulting stiffness matrix is reduced to its reduced representation $\mathbf{L}$.

5.2.2 Approximating the reduced representation $\mathbf{L}$

This section discusses the approximation of the map from the feature space $(r_1, \ldots, r_{n_b}) \in \mathcal{F}_T$ to the reduced joint stiffness matrix space $\mapsto \mathbf{L} \in \mathcal{J}_T$. To reiterate, the goal is to circumvent the computationally heavy D-PSE method by a more efficient but still accurate routine. Although this is conceptually a straightforward operation, approximating high-dimensional maps is up to this date still a challenging computational task due to the curse of dimensionality. A high-level overview is given here. For more detailed information, the reader is referred to Appendix C.

Two approximation types are developed: a matrix-level technique that approximates the complete $\mathbf{L}$ matrix and an entry-level technique that approximates each entry of $\mathbf{L}$ separately. The main idea of both techniques is to construct an approximation that separates the dependence on each feature and then combines them through first multiplication and then summation. As an example, for the matrix-level approach the normalized, reduced stiffness matrix $\mathbf{L}_n = \mathbf{L} / \|\mathbf{L}\|$ is approximated by

$$\mathbf{L}_n(f_1, \ldots, f_{n_b}) \approx \sum_{r=1}^{R_1} u_r^1(f_1) \cdot \cdots \cdot u_r^{n_b}(f_{n_b}) \cdot \mathbf{L}_r^b.$$

Here, $u_r^i(f_i)$ is a function depending only on the i-th feature, $\mathbf{L}_r^b$ is the r-th basis matrix and the parameter R_1 determines the complexity of the model.

5.3 Expansion

Evaluation of the approximation leads to an approximation $\tilde{\mathbf{L}}$ of the reduced representation $\mathbf{L}$ of the joint matrix $\mathbf{K}_j$. This is followed by expansion of $\tilde{\mathbf{L}}$, leading to an approximation $\tilde{\mathbf{K}}_j$ of the joint matrix. This expansion routine thus reverses the reduction routine and in the process not only ensures matrix symmetry, positive semi-definiteness and the correct rigid body modes but also transforms the parametrized superelement from its standard to its actual material and orientation.
Positive semi-definiteness and symmetry are guaranteed since $\mathbf{K}_{ii} = \mathbf{L}\mathbf{L}^T$. What remains are the rigid body modes, the orientation and the material, discussed in §5.3.1, §5.3.2 and §5.3.3, respectively.

5.3.1 Reapplying rigid body modes

Given Eq. 17 and an approximation of $\mathbf{K}_{ii}$, the missing parts of the complete joint matrix are found through

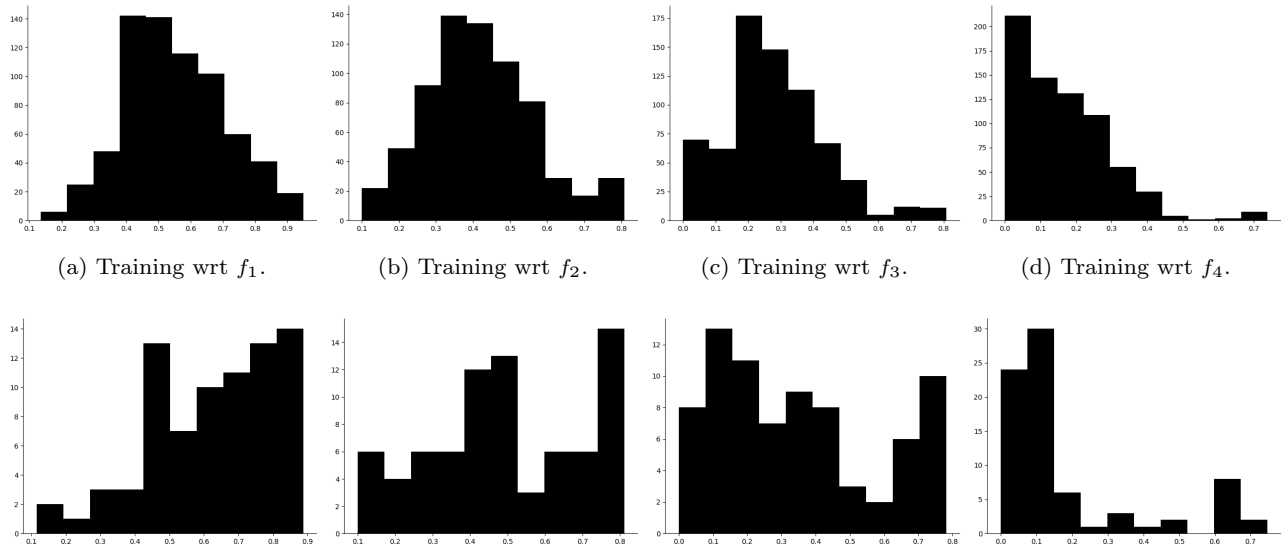
$$\mathbf{K}_{id} = \mathbf{K}_{di}^T = -\mathbf{K}_{ii}\mathbf{R}_i\mathbf{R}_d^{-1},$$
$$\mathbf{K}_{dd} = -\mathbf{K}_{di}\mathbf{R}_i\mathbf{R}_d^{-1}.$$

Here, $\mathbf{R}_d$ is invertible when the dependent dof are chosen properly and the resulting joint matrix $\mathbf{K}_j$ is assured to have the correct rigid body modes.

5.3.2 Scaling and coordinate transformation

After reduction, approximation and expansion of the symmetry and rigid body modes, the resulting stiffness matrix represents a joint with unit material in its standard configuration. This section elaborates on the transformation from the joint stiffness matrix $\mathbf{K}_j^S$ in the standard configuration S to $\mathbf{K}_j^A$, the stiffness matrix in its actual configuration A. In [36], this is described for transformations $\mathbf{p} \mapsto \mathbf{A}\mathbf{p} + \mathbf{b}$ that preserve angles and volume, namely rotation and reflection.[6] A necessary requirement for $\mathbf{A}$ is orthogonality (i.e. $\mathbf{A}^T\mathbf{A} = \mathbf{A}\mathbf{A}^T = \mathbf{I}$). A brief overview of [36] is given here.
Through a coordinate change interpretation, it is possible to show that $\mathbf{K}_j^A = \mathbf{M}^T\mathbf{K}_j^S\mathbf{M}$, where $\mathbf{M}$ is a block-diagonal matrix with $\mathbf{A}$ on its diagonal repeated once for every set of three dof of the (joint) stiffness matrix.

[6] The translational component $\mathbf{b}$ is added for completeness but can easily be shown to have no effect on the stiffness matrix.



(a) Training wrt f_1. (b) Training wrt f_2. (c) Training wrt f_3. (d) Training wrt f_4.

(e) Test wrt f_1. (f) Test wrt f_2. (g) Test wrt f_3. (h) Test wrt f_4.

Fig. 13: Sample distribution of the training set (700 samples) and test set (77 samples) of internal diamond joint with four features (f_1 up to f_4). The training samples are generated on a $25\times25\times25\times25$ grid. Samples with a large f_4 are hard to generate due to the symmetry constraint $f_1 \geq f_2 \geq f_3 \geq f_4$.

Furthermore, if the transformation is expressed as a sequence of m subtransformations $\mathbf{p} \mapsto \mathbf{A}_i\mathbf{p}+\mathbf{b}_i$, then

$$\mathbf{K}_j^A = \mathbf{M}_m^T \cdots \mathbf{M}_1^T \mathbf{K}_j^S \mathbf{M}_1 \cdots \mathbf{M}_m,$$

where $\mathbf{M}_i$ is again a block-diagonal matrix, now with $\mathbf{A}_i$ on its diagonal:

$$\mathbf{M}_i = \begin{bmatrix} \mathbf{A}_i & & \\ & \ddots & \\ & & \mathbf{A}_i \end{bmatrix}.$$

These results are now extended to uniform scaling, i.e. $\mathbf{A}=\alpha\mathbf{I}$, by means of the following dimensional analysis. Consider a 6×6 elementary stiffness matrix $\mathbf{K}$ describing the stiffness between the translational and rotational dof of a single node such that $\mathbf{Ku}=\mathbf{f}$. It is now stated that each stiffness element is the product of 1) a material parameter (an element of the stiffness tensor $\mathbb{S}$ s.t. $\sigma=\mathbb{S}:\varepsilon$) with units $\mathrm{N/m^2}$ and 2) a size parameter that depends solely on the geometry of the superelement.

Only the size parameter is affected by the uniform scaling. It depends on the uniform scaling through it's units. Consider therefore the following subdivision of $\mathbf{K}$ based on which types of dofs it connects (translational or rotational):

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{uu} & \mathbf{K}_{u\theta} \\ \mathbf{K}_{\theta u} & \mathbf{K}_{\theta\theta} \end{bmatrix}.$$

The translational stiffness elements in $\mathbf{K}_{uu}$ relate displacements to forces and therefore have units Nm. The corresponding size parameter thus must have units m, which scales linearly with α. Through a similar derivation, the mixed blocks $\mathbf{K}_{u\theta}$ and $\mathbf{K}_{\theta u}$ and the rotational block $\mathbf{K}_{\theta\theta}$ are found to scale with α^2 and α^3, respectively.

Thus, undoing a uniform scaling can be implemented in a similar way as undoing the rotations and reflections: by means of a block-diagonal matrix $\mathbf{M}_{m+1}$ containing $\sqrt{\alpha}\mathbf{I}$ and $\alpha^{3/2}\mathbf{I}$ on blocks corresponding to translational and rotational dof, respectively. By multiplying all $m+1$ transformation matrices $\mathbf{M}_i$ into $\mathbf{M}=\prod_{i=1}^{m+1}\mathbf{M}_i$, the effect of rotation, reflection and uniform scaling on the stiffness matrix $\mathbf{K}_j^S$ can be summarized as

$$\mathbf{K}_j^A = \mathbf{M}^T\mathbf{K}_j^S\mathbf{M}.$$

5.3.3 Material transformation

The final expansion step deals with going from the standard unit material to the actual material of the joint. Both the Young's modulus E the Poisson ratio (or any two other Lamé parameters) should be changed. Since the solid joint mesh only contains solid elements, its stiffness matrix $\mathbf{K}_{int}$ is linearly related to the Young's modulus and one can then easily show that the joint stiffness matrix, related to $\mathbf{K}_{int}$ via Eq. 15, is also linear in E. Going from the unit Young's modulus of $E_{unit} = 1\text{N/m}^2$ to the actual Young's modulus E_A thus only requires multiplication by E_A/E_{unit}. The Poisson ratio is tougher to incorporate and should in general be incorporated as an extra parameter during sampling and approximation. In this paper, it is kept constant for simplicity.

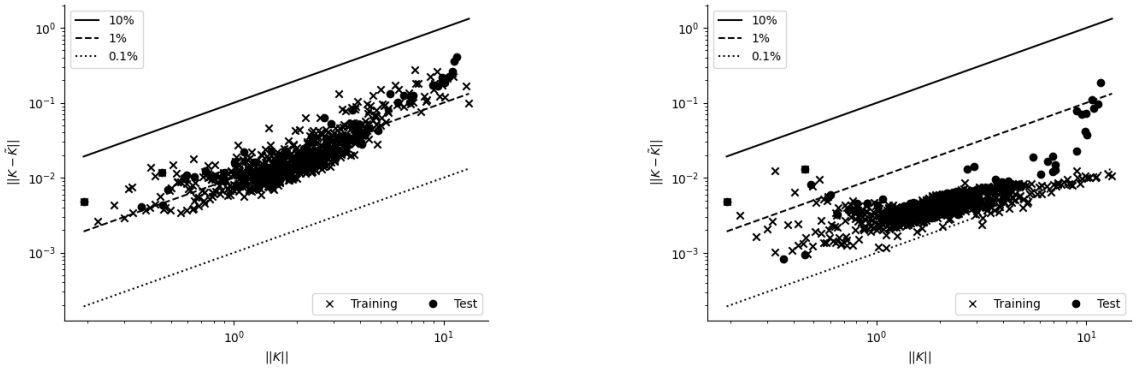
5.4 Numerical example and discussion

A numerical example is now used to discuss three elements of the A-PSE method: its accuracy, its computational efficiency and its relation to the literature, more specifically the ARSP method developed by Wu et al. [40]. The numerical example focuses on the internal diamond lattice joint (see Fig. 4), which connects to four beams and thus requires approximation of a four-dimensional space. Both the matrix-level and the entry-level approaches are applied to a set of 700 training samples and 77 test samples, of which the distribution is shown in Fig. 13. For $\mathbf{T}_1$, the matrix-level method uses 34 basis matrices and rational functions with sixth-degree numerators and second-degree denominators for the factor vectors $\mathbf{u}_n^k$. For $\mathbf{T}_2$, a rank of $R = 10$ is used and the factor vectors $\mathbf{v}_n^k$ are restricted to lie on fifth-degree polynomials. The entry-level method also uses rational functions with sixth-degree numerators and second-degree denominators for its factor vectors $\mathbf{w}_n^k$. Solution of the nonlinear least-squares optimization problem is performed by Tensorlab v3 [37].

Accuracy. Figure 14 shows the absolute errors (computed with the L2 norm) as a function of the L2 norm, achieved by both methods on the same samples. From this figure, three conclusions on accuracy will now be drawn. First, from the comparison of Fig. 14a and Fig. 14b, it is clear that the entry-level approach has achieved a higher accuracy, albeit at a higher computational cost. Second, since the matrix-level approach first scales the norm of the matrices, the relative error is minimized and the sample error distribution is parallel to the lines of constant relative error. By contrast, the entry-level approach is unable to minimize the relative error and instead optimizes the absolute error. This causes training samples with a large norm to be overfitted and explains the rift between the test and training samples for high norms in Fig. 14b. However, the test set error mostly stays below the expected 1% discretization error (see §4.3). Third, for very low norms the sample accuracy breaks down both for the entry-level and the matrix-level approach. This is very apparent in Fig. 14b where at low norms both the relative training and test set error go up. This is presumed to occur because of the discretization error of the D-PSE method, since samples in this region correspond to joints with a small radius (around 10% of the joint radius) and the mesh is unable to properly capture these small faces with sufficient accuracy.

Computational efficiency. A conclusion on computational efficiency is now drawn based on Table 1. Two important factors must be mentioned before studying the detailed timings. First, since the A-PSE method uses the D-PSE method for sampling, Table 1 reports computation times normalized with respect to the time needed to construct a single diamond lattice joint element with the D-PSE method. The number of elements used in the D-PSE method is chosen such that a discretization error of around 1% is achieved on the stiffness matrix, which results in at least 40k tetrahedral elements and a computation time of around nine minutes (on a computer with 2× Intel(R) Xeon(R) CPU, 14 cores and 126GB RAM).

Second, parallelism greatly influences the computation time but not the computational cost, i.e. the number of floating-point operations. Only the computation time is reported in Table 1. The sample generation is carried out with eleven processes working in parallel and is bottlenecked by the RAM used during the sparse LU factorization. The tensor decompositions and online approximation evaluations are carried out on a laptop with an Intel(R) Core(TM) i7, 6 cores and 64GB RAM. The entry-level A-PSE method uses explicit parallelism for both the tensor decompositions and the evaluation, whereas the matrix-level method does not. This imme-



(a) Matrix-level approach.

(b) Entry-level approach.

Fig. 14: Errors on the same training and test set of an internal diamond lattice joint.

diately showcases the "embarrassingly parallel" nature of the entry-level method: it lends itself perfectly to aggressive parallelization.

Three conclusions are now drawn based on the results of Table 1. First, the sampling is undoubtedly the most expensive step and parallelization greatly helps reducing the computation time from a few days to under twelve hours. Since this step is offline, it need only be carried out once.

Second, the online cost of evaluating the joint approximations is negligible compared to the D-PSE method cost. As the next section will show, evaluating thousands of joint approximations during the meshing step of a complete lattice simulation is less expensive than solving the resulting assembled stiffness matrix. This is in line with standard finite element solvers, where stiffness matrix assembly has lower computational complexity than linear system solving.

Third, the computational cost of the entry-level approach is higher than the matrix-level approach due to its vastly larger number of parameters. However, aggresive parallelization alleviates most issues here. For our setup, only a 15% increase in decomposition time is observed.

Relative computation time [-]	Offline		Online
	Sampling	CPD	
D-PSE	0.	0.	1.
A-PSE, matrix-level	68.7	7.55	2.72e-5
A-PSE, entry-level	68.7	8.66	5.8e-5

Table 1: Computation time comparison between the matrix- and entry-level approach. Values are normalized with respect to the computation time for evaluation of a single joint with the D-PSE method, which is around nine minutes.

6 Numerical example

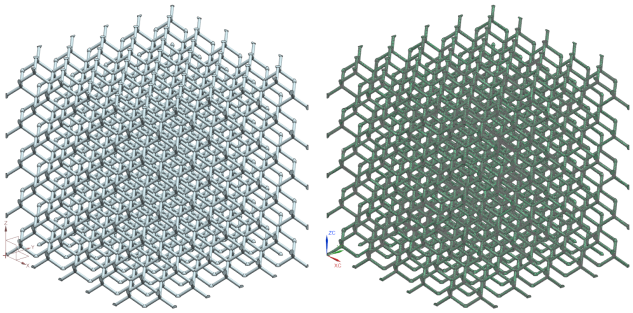
This section provides a numerical example to showcase that joint elements can be used to efficiently simulate global properties of large-scale lattices. To this end, the shear and compression behaviour of a $5 \times 5 \times 5$ diamond lattice (shown in Fig. 15a) with increasing relative density is simulated. Three meshing techniques are used: the solid mesh (Fig. 15b), the purely beam element mesh without any of the joint compensation techniques discussed in §2.1 and the combined beam and joint element mesh (referred to as the PSE mesh). For the face, edge and corner joints (see §5.4), PSE joint elements are built with the parameters of Table 2.

The following two setups are used to extract global properties.

– **Pure compression**: the lattice structure is fixed in the vertical direction at the bottom and a unit vertical displacement δ_y is applied at the top. An illustration of the exact boundary conditions is

	R_1	R_2	min. elements	$u_n^k(f_k)$ is rational function with	$v_n^k(f_k)$ is polynomial of degree
Face	10	10	30 000	4th degree numerator, 2nd degree denominator	5
Edge	4	10	30 000	4th degree numerator, 2nd degree denominator	3
Corner	4	10	30 000	4th degree numerator, 2nd degree denominator	3

Table 2: Details regarding the creation of joint elements for the face, edge and corner joints inside a diamond lattice. All approximations use the matrix-level method since the number of beam n_b is low (two for the face joints, one for the edge and corner joints). R_1 and R_2 correspond to the complexity of the approximation and are explained in detail in Appendix C.



(a) Geometry. (b) Solid mesh.

Fig. 15: Geometry and solid mesh of a $5\times5\times5$ diamond lattice, for $\bar{\rho}=2.61\%$. Generated and visualized with Simcenter 3D [31].

shown in Fig. 16a. The vertical reaction force R_y is used to compute the apparent Young's modulus

$$\bar{E} = \frac{R_y L}{A\delta_y},$$

where L and A are the height and cross-sectional area of the lattice structure.

– **Pure shear**: the lattice structure is again fixed at the bottom and a unit horizontal displacement δ_x is applied at the top. An illustration of the exact boundary conditions is shown in Fig. 16b. The horizontal reaction force R_x is used to compute the apparent shear modulus

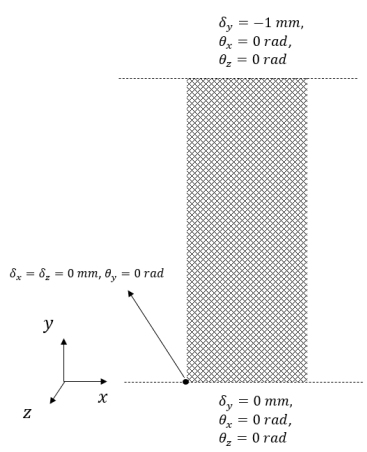
$$\bar{G} = \frac{R_x L}{A\delta_x},$$

where L and A are again the height and cross-sectional area of the lattice structure.

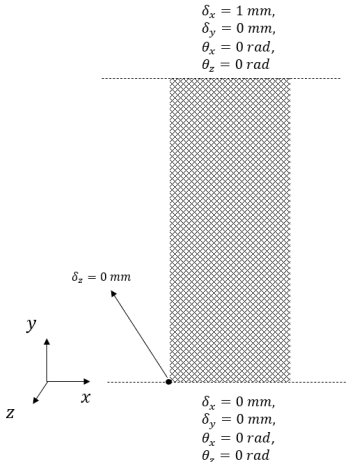
Accuracy. Figures 17 and 18 show the $\bar{E}(\bar{\rho})$ and $\bar{G}(\bar{\rho})$ curves obtained with the three mesh types and the errors with respect to the solid mesh. Both the beam element mesh and the mesh containing joint elements underestimate the stiffness. For the beam element mesh, this is most likely because the joint stiffness is not incorporated, although an overestimation could also have occurred due to the strict straight face constraints. Adding joint elements reduces the error, although the more relaxed dual form of the straight face constraints still underestimates the apparent Young's modulus found with the solid element mesh. However, a great advantage of using joint elements, despite their stiffness underestimation, is that no arbitrary beam element modifications are required to incorporate the joints, as discussed in §2.1. Furthermore, joint elements allow evaluation of stress concentrations near joints, although further validation in this direction is left to future work.

Computation time. A great advantage of the PSE method is the computational efficiency vs accuracy trade-off. Table 3 shows rough estimates of the computation times needed for generating and solving the solid, PSE and beam meshes.

The number of elements used for the solid mesh was decided based on a mesh convergence test and ranges between 400 000 and 500 000 second-order tetrahedral elements. Generating these meshes alone is a very expensive task due to the complicated CAD geometry: one solid mesh creation took around six minutes to complete. Furthermore, not included in the table is the scripted generation of the CAD geometry in Simcenter 3D [31] which, due to the large number of beams and joints, takes a little more than three minutes and becomes progressively harder when more unit cells are added. However, the most expensive task remains the solution of the assembled stiffness matrix, which (de-

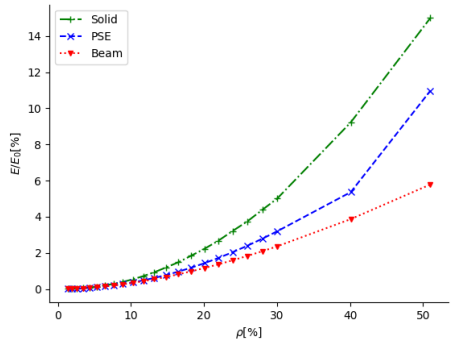


(a) A negative unit y translation is applied to the top while the bottom y translation is kept fixed. Both x and z rotations are kept fixed at the top and bottom. A point chosen at the lower left fixes the in plane translation and the y rotation.

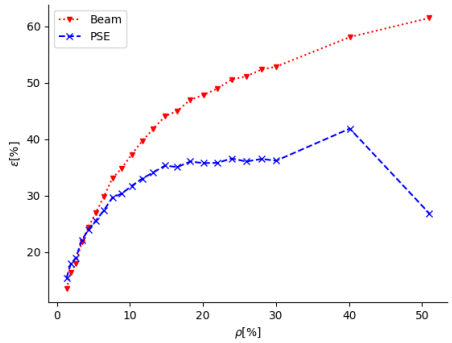


(b) A unit x translation is applied to the top while the top y and bottom x and y translation is kept fixed. Both x and z rotations are kept fixed at the top and bottom. A point chosen at the lower left fixes the z translation.

Fig. 16: Applied boundary conditions and loads for compression (left) and shear (right) simulation.



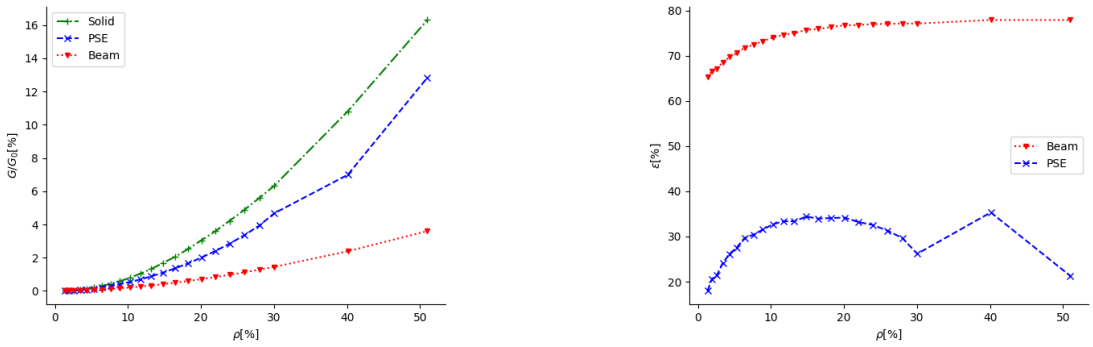
(a) The $\bar{E}(\bar{\rho})$ curve.



(b) Relative error with respect to the solid mesh.

Fig. 17: Comparison of the $\bar{E}(\bar{\rho})$ curve of a diamond lattice structure using solid elements (labelled "Solid"), beam elements (labelled "Beam") and a combination of both joint and beam elements (labelled "PSE").

pending on the number of elements) took between five and twenty minutes for a single solid mesh. This is in large contrast to the PSE mesh, containing 1162 joint elements and 2000 beam elements, and the simple beam element mesh containing its 2000 beam elements in total. With this small number of elements, meshing becomes a much easier task and takes only a fraction of the computation time. Solving a beam element mesh took roughly nine seconds, of which three seconds were used to generate and assemble the elemental beam stiffness matrices and the remaining six seconds were used to solve the assembled stiffness matrix. The PSE mesh took around 40 seconds to solve, of which ten were used to generate and assemble both the beam and the joint element stiffness matrices. Regarding the offline cost, a net gain in computation time occurs



(a) The $\bar{G}(\bar{\rho})$ curve.

(b) Relative error with respect to the solid mesh.

Fig. 18: Comparison of the $\bar{G}(\bar{\rho})$ curve of a diamond lattice structure using solid elements (labelled "Solid"), beam elements (labelled "Beam") and a combination of both joint and beam elements (labelled "PSE").

	Meshing	Matrix assembly	Matrix solve
Solid	300-500	− − −	300-1000
PSE	$\sim 0.$	11-12	26-35
Beam	$\sim 0.$	3.3-3.6	5.7-6.4

Table 3: Rough computation time comparison between the solid, PSE and beam element mesh, expressed in seconds. The stiffness matrix assembly includes the elemental stiffness matrix construction and is added to show matrix solving remains the most expensive computational step. For the solid mesh, this timing is included in the linear system solving time. All times were measured on a laptop with an Intel(R) Core(TM) i7, 6 cores and 64GB RAM.

when simulating more than twenty to thirty lattices, depending on lattice size, mesh density and approximation accuracy among others. The PSE method should thus only be used when one needs to simulate larger amounts of lattices, e.g. during optimization. Another option is to use the PSE method when the lattice becomes so large that a solid mesh is unsolvable. Indeed, when increasing to number of unit cells from $5\times5\times5$ to $10\times10\times10$ and more, multiple millions of solid elements are required and the limits of computational resources are easily exceeded.

7 Conclusions and future work

An efficient and accurate method for simulation of structures with large amounts of parts that vary in a select set of parameters is proposed and applied to lattice structures. To this end, the lattice beams are meshed with standard Timoshenko beam elements and the lattice joints are represented by a Parameterized Superelement. In the D-PSE method, a joint element is constructed by meshing the geometry with solid elements, applying dual straight face constraints to create a fixed interface and finally statically condensating the structure to its exterior degrees of freedom. This method is shown to converge experimentally when the mesh refines and shown to adhere to the rigid body modes. Furthermore, a significant model order reduction is achieved. To circumvent the expensive static condensation during simulation, approximation is performed by the A-PSE method. Here, the space of joint stiffness matrices is first sampled, then reduced to its most basic form, after which the multidimensional space is approximated with tensor decompositions and expansion is applied. Numerical examples showcase the accuracy and computational efficiency of using the PSE method for lattice simulation. Although the global stiffness is slightly underestimated due to the relaxed dual form of the straight face constraints, the error with respect to an expensive solid mesh stays within forty percent and the formulation allows evaluation of the forces and stresses inside the lattice joints. Due

to the offline cost of sampling and fitting the joint element space, the PSE method should only be used when simulating more than twenty lattice structures. Due to its low online cost, the PSE method is ideally suited for lattice structure optimization, parameter exploration and nonlinear fatigue simulations.

Future work. Four directions for future work and development are now outlined. First and foremost, more validation (both numerical and experimental) should be performed to fully assess the capabilities of the proposed lattice simulation methodology. Second, the approach must be extended to lattice unit cells other than the diamond unit cell. Some initial work has already been carried out here but is impeded due to increasing difficulty of fitting higher-dimensional spaces. Third, improvements for the discretization and approximation error should be studied, such as higher-order meshing, improved sampling and approximation of the compliance matrix instead of the stiffness matrix to better approximate the smallest eigenvalues. Fourth, the effect of the Poisson ratio on the joint element matrix and the possibility of a priori modelling of this effect must be investigated.

A Traction vector on a plane surface

Consider a planar surface on which a reaction force $\mathbf{f}$ and a reaction moment $\mathbf{m}$ is applied. If the traction vector is linear over the surface, then it can be written as

$$\mathbf{t_n}(x,y,z) = \mathbf{t}_f + \mathbf{t}_m \times (\mathbf{p} - \mathbf{p}_C), \tag{18}$$

where $\mathbf{p}_C$ is the surface centroid and $\mathbf{t_f}$ and $\mathbf{t_m}$ are unknown but constant vectors that depend on the reaction force and moment via the averaging Eqs. 2 and 3, respectively. Substituting Eq. 18 into Eq. 2, it can readily be seen that $\mathbf{t_f} = \mathbf{f}/A$, i.e. applying a force to a plane surface leads to a constant traction vector. Substituting Eq. 18 into Eq. 3 leads to:

$$\begin{aligned}\mathbf{m} &= \int_A (\mathbf{t}_f + \mathbf{t_m} \times (\mathbf{p} - \mathbf{p}_C)) \times (\mathbf{p} - \mathbf{p}_C)\,\mathrm{d}A \\ &= \int_A \mathbf{t}_f \times (\mathbf{p} - \mathbf{p}_C)\,\mathrm{d}A \\ &\quad + \int_A (\mathbf{t}_m \times (\mathbf{p} - \mathbf{p}_C)) \times (\mathbf{p} - \mathbf{p}_C)\,\mathrm{d}A \\ &= \int_A \|\mathbf{p} - \mathbf{p}_C\|_2^2\, \mathbf{t}_m\,\mathrm{d}A - \int_A ((\mathbf{p} - \mathbf{p}_C) \cdot \mathbf{t}_m)(\mathbf{p} - \mathbf{p}_C)\,\mathrm{d}A.\end{aligned}$$

The last equality holds due to the vector triple product. Introducing the second moment of area tensor $\{\mathbf{J}\colon J_{i,j} = \int_A (x_i - x_{i,C})(x_j - x_{j,C})\,\mathrm{d}A\}$, this can be rewritten as:

$$(\mathbf{J} - \mathrm{tr}(\mathbf{J})\mathbf{I}_3)\mathbf{t_m} = \mathbf{m},$$

where $\mathbf{I}_3$ is the 3×3 unit matrix. The matrix $\hat{\mathbf{J}}$ is defined as

$$\hat{\mathbf{J}} = \mathbf{J} - \mathrm{tr}(\mathbf{J})\mathbf{I}_3,$$

simplifying the expression for the rotational component of the traction vector to $\hat{\mathbf{J}}\mathbf{t_m} = \mathbf{m}$.

B Sampling algorithm details

The sampling algorithm's details are now elaborated. It consists of two main decision steps and an acceptance-rejection step.

The first decision step chooses the required number of samples for the training and the test set. For the training set, a decision is made based on the required number of samples $N_{training}$ for the subsequent approximation to be unique. That is, there should be at least as many samples as there are parameters (e.g. basis function coefficients) in the approximate model. Of course, the number of approximation parameters is only known after increasing their number until approximation error convergence is achieved. This was done for some joint types starting from a rough estimate of $N_{training} \approx R \cdot n_b \cdot N_D$, where R is the expected rank and N_D the expected number of basis function coefficients for a single dimension, based on [5, Question 4]. Initial attempts showed $R \approx 30$ and $N_D \approx 6$, yielding between 100 and 800 training samples for diamond lattice joints. It must be noted that the amount of training samples increases (usually by at most 10 samples) during approximation, since samples are added in an adaptive way, close to test samples with a large approximation error. To conclude, the number of test samples is chosen such that the test set contains roughly ten percent of the total amount of samples.

The second decision step chooses the number and distribution of "sparse" samples, i.e. samples containing one or more zero features. These sparse samples thus represent joints that lack one or more beams. These samples are important since they lie at the boundary of the sample domain and are thus otherwise very hard to sample. In general, training and test sets are created consisting of roughly thirty percent sparse samples. When a sample is chosen to be sparse, a decision is made on the measure of "sparsity" of this sample or, in other words, how many features should be zero. This decision is made by uniformly dividing the sparse samples over all possible measures of sparsity, going from only a single zero feature to all of them except one (ensuring of course that the joint still has external faces).

Finally, an acceptance-rejection method creates the samples in three substeps. Consider the required generation of a sample $(r_1, \ldots, r_{n_b})$ with n_0 zero features. First, a random selection is performed: a total of n_0 features are chosen and kept at zero. Second, the remaining $n_b - n_0$ features are generated in a randomized fashion. Third, the generated samples are either accepted or rejected based on both the symmetry and the overlap constraints mentioned in §5.1 and visualized in Fig. 12. The workings of the second substep are now further explained.

Test sample features are generated from two types of normal distributions, $\mathcal{N}_1(\mu_1, \sigma_1)$ and $\mathcal{N}_2(\mu_2, \sigma_2)$. $\mathcal{N}_1$ has a large standard deviation σ_1 (e.g. 0.25) to ensure the bulk of the feature space is properly sampled, whereas $\mathcal{N}_2$ has a smaller deviation σ_2 (usually 0.1) but a larger mean μ_2 (around 0.8) to better sample the feature space boundaries. For both, features are bounded to be in the interval $[r_{min}, r_{max}]$ by applying $\max(\min(r, r_{max}), r_{min})$ to a sampled value r. Here, r_{min} is usually chosen at 0.1 since meshing problems occur for smaller radii.

Training sample features are generated in a similar way. However, since a tensor decomposition is used during approximation, a discrete grid is employed. Consider an n_b-dimensional feature space with N points in every dimension, thus containing a total of N^{n_b} points (see Fig. 19). The k-th dimension of this multidimensional grid is indexed by i_k, where $k = 1, \ldots, n_b$, such that $f_k(i_k) = f_k^{\min} + i_k \frac{f_k^{max} - f_k^{min}}{N-1}$, $i_k = 0, \ldots, N-1$. Discrete versions of the normal distributions [7] are employed to sample the grid. This results in a sampled multi-index $(i_1, \ldots, i_{n_b})$ that correspond to joints with a reduced representations $\mathbf{L}(f_1(i_1), \ldots, f_{n_b}(i_{n_b}))$. Since the sampling is quite sparse and a lot of samples are rejected, for a dimension k only a subset of all possible indices $i_k \in [0, \ldots, N-1]$ are used. The indices i_k are therefore omitted in favor of the reduced indices

$$j_k \in [1, N_k], \tag{19}$$

where $N_k \leq N+1$. For example, in Fig. 19 there is no sample with $i_1 = 0$ and thus $N_1 = N-1 = 2$. The reduced joint representations can also be written as $\mathbf{L}(f_1(j_1), \ldots, f_{n_b}(j_{n_b}))$, which we abbreviate to $\mathbf{L}(j_1, \ldots, j_{n_b})$.

A major challenge with the above sampling method is the difficulty of sampling close to the feature space boundary (see Fig. 12). This boundary is essential since in practice only joints lying on this boundary are used. Consider a joint in the bulk region, such as the one shown in Fig. 11. The radius of this joint can be reduced without causing overlap, and therefore, as is also clear from §3, such a joint will be omitted in favor the closest joint on the feature boundary. However, during sampling the joint radius is fixed at 1 and complicates sampling the boundary. Although the $\mathcal{N}_2$ distribution used for this purpose comes reasonable close to the feature space boundary for test samples, training samples are limited by the hypergrid. An obvious improvement could therefore be to refine the grid spacing to a certain tolerance. From a higher level though, this challenge stems from the fact that the feature space has one superfluous parameter, namely the joint radius currently kept at 1. Future work will focus on this challenge.

[7] The normal distributions are discretized by simply evaluating them at the features on the grid and rescaling them to ensure the total probability stays 1.

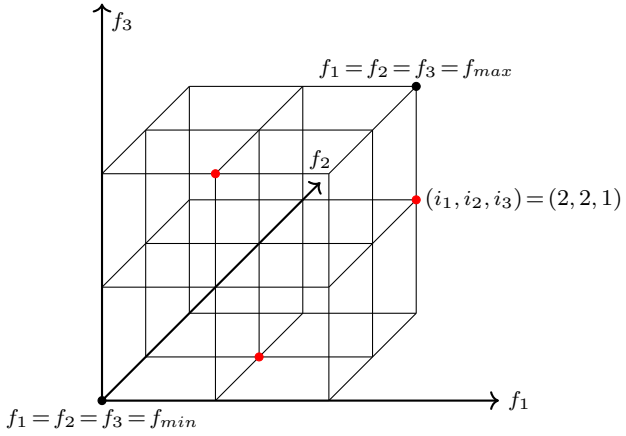


Fig. 19: Multidimensional feature grid for features f_1, f_2 and f_3 with $N = 3$ and three randomly selected samples shown in red.

C Approximation algorithm details

Two techniques are developed for the approximation of the map from the feature space to the reduced stiffness matrix space: a matrix-level and an entry-level technique. Both employ tensor decomposition, more specifically the canonical polyadic decomposition (CPD) [17]. A detailed overview of both methods will be given here.

In general, CPD decomposes a k-dimensional tensor $T \in \mathbb{R}^{n_1 \times \cdots \times n_k}$ into an approximate sum of rank-1 tensors, similar to a singular value decomposition,

$$T \approx \sum_{r=1}^{R} \mathbf{u}_r^1 \otimes \cdots \otimes \mathbf{u}_r^k,$$

where $\mathbf{u}_r^j \in \mathbb{R}^{n_j}$ are called factor vectors and R is the length or rank of the CPD. Recall that the tensor product $\mathbf{A} = \mathbf{u}_r^1 \otimes \cdots \otimes \mathbf{u}_r^k$ is defined elementwise as

$$\mathbf{A}(i_1, \ldots, i_k) = \mathbf{u}_r^1(i_1) \cdot \cdots \cdot \mathbf{u}_r^k(i_k).$$

In particular, $\mathbf{A} = \mathbf{u}_r^1 \otimes \mathbf{u}_r^2 = \mathbf{u}_r^1 (\mathbf{u}_r^2)^T$.

Both the matrix- and the entry-level method convert the reduced joint matrix representation $\mathbf{L}$ of the sample points to two or more tensors. More specifically, the matrix-level method constructs a tensor $\mathbf{T}_1$ containing the normalized stiffness matrices of the samples and another tensor $\mathbf{T}_2$ containing the sample L2 norms, whereas the entry-level method constructs a tensor for every entry of $\mathbf{L}$. These tensors are *incomplete* since not all points on the hypergrid are sampled.
After the tensors are generated, they are assumed to be decomposable into a CPD of small length where some or

all of the factor vectors $\mathbf{u}_r^j$ are restricted to be the evaluation univariate functions u_r^j ("factor functions") of their corresponding feature f_j. That is, the entries in the factor vectors are the evaluations of the factor functions on the multidimensional regular, n_b-dimensional grid from the sample generation step. This is enforced through *structured data fusion* [33]. In this paper, both polynomial and rational functions are used.
Next, a decomposition that satisfies these restrictions is found through the solution of a nonlinear least-squares optimization problem that minimizes the error between the decomposition and the reduced joint stiffness matrices of the sampled joint configurations inside the tensors.
Finally, the decomposition and more specifically, the factor function's coefficients, are extracted and transformed into an approximation $I(f_1,\ldots,f_{n_b})$ of the reduced joint stiffness matrix representation $\mathbf{L}(f_1,\ldots,f_{n_b})$, such that $\mathbf{L}(f_1,\ldots,f_{n_b})\approx I(f_1,\ldots,f_{n_b})$.

Matrix-level approximation. In the matrix-level approach, two tensors are used. The first tensor, $\mathbf{T}_1$, contains the normalized, reduced joint stiffness matrices and is thus used to approximate the normalized joint matrix. The second tensor, $\mathbf{T}_2$, contains the L_2 *norms* of the reduced joint matrices and is used to approximate the norm of the joint matrix. This division into a tensor for the matrix norm and a tensor for the normalized matrix is employed to prevent the optimization algorithm from focusing solely on joint matrices with a high norm.

The first tensor $\mathbf{T}_1$ has n_b+1 dimensions, i.e.

$$\mathbf{T}_1 \in \mathbb{R}^{N_1\times\cdots\times N_{n_b}\times N_{n_b+1}}.$$

Each dimension N_k is indexed by a reduced index j_k from Eq.19. The first n_b dimensions are the feature dimensions and indexed by the first n_b reduced hypergrid indices. They are from now on referred to as the feature indices. The last, n_b+1-th dimension is the 'matrix' dimension and its index $j_{n_b+1}\in[1,N_{n_b+1}]$ selects an entry from the linearized $\mathbf{L}_n$ matrix by converting it to a vector $\mathbf{l}_n\in\mathbb{R}^{N_{n_b+1}}$. Since $\mathbf{L}_n\in\mathbb{R}^{n_r\times n_r}$ is lower-triangular, $N_{n_b+1}=\binom{n_r+1}{2}$.
As an example, consider a sample with feature indices $(j_1,\ldots,j_{n_b})$ and its corresponding reduced representation of the joint matrix $\mathbf{L}(j_1,\ldots,j_{n_b})$. The normalization of this matrix is then defined as

$$\mathbf{L}_n(j_1,\ldots,j_{n_b}) = \frac{\mathbf{L}(j_1,\ldots,j_{n_b})}{\|\mathbf{L}(j_1,\ldots,j_{n_b})\|_2}\in\mathbb{R}^{n_r\times n_r}.$$

The entries of $\mathbf{L}_n(j_1,\ldots,j_{n_b})$ are then found by indexing $\mathbf{T}_1$ as $\mathbf{T}_1[j_1,\ldots,j_{n_b},:]$. Here, the : index is a slice across the matrix dimension and thus contains the entries of the lower-triangular part of $\mathbf{L}_n(j_1,\ldots,j_{n_b})$. Selecting an entry of that matrix is defined as $\mathbf{T}_1[j_1,\ldots,j_{n_b},j_{n_b+1}]=\mathbf{l}_n(j_1,\ldots,j_{n_b})[j_{n_b+1}]$. The second tensor $\mathbf{T}_2$ has n_b feature dimensions and only contains the norms of the reduced joint matrices, i.e. $\mathbf{T}_2[l_1,\ldots,l_{n_b}]=\|\mathbf{L}(l_1,\ldots,l_{n_b})\|_2$.

Both $\mathbf{T}_1$ and $\mathbf{T}_2$ are then approximated with a CPD. The first is decomposed as follows:

$$\mathbf{T}_1 \approx \sum_{r=1}^{R_1} \mathbf{u}_r^1\otimes\cdots\otimes\mathbf{u}_r^{n_b}\otimes\mathbf{u}_r^{n_b+1}.$$

The feature factor vectors $\mathbf{u}_r^k$, with $k=1,\ldots,n_b$, are restricted to be the evaluations of polynomial or rational functions $u_r^k(f_k)$. Their degree, together with the number of terms R_1, is increased until the approximation error reached the discretization error observed during the mesh convergence tests. Based on our experiments, around thirty terms and rational functions with a sixth-degree numerator and a second-degree denominator usually suffice to properly represent the joints inside a diamond lattice. The last factor vectors $\mathbf{u}_r^{n_b+1}$, $r=1,\ldots,R_1$, are not restricted. They are thus optimized freely and afterwards delinearized into R_1 lower-triangular basis matrices $\mathbf{L}_r^b\in\mathbb{R}^{n_r\times n_r}$, $r=1,\ldots,R_1$. The following functional approximation for the normalized, reduced joint matrix representation $\mathbf{L}_n(f_1,\ldots,f_{n_b})$ is then obtained:

$$\begin{aligned}\mathbf{L}_n(f_1,\ldots,f_{n_b}) &\approx \sum_{r=1}^{R_1} u_r^1(f_1)\cdot\cdots\cdot u_r^{n_b}(f_{n_b})\cdot\mathbf{L}_r^b\\ &=: \mathbf{I}_1(f_1,\ldots,f_{n_b}).\end{aligned}$$

The second tensor is decomposed as

$$\mathbf{T}_2 \approx \sum_{r=1}^{R_2} \mathbf{v}_r^1\otimes\cdots\otimes\mathbf{v}_r^{n_b},$$

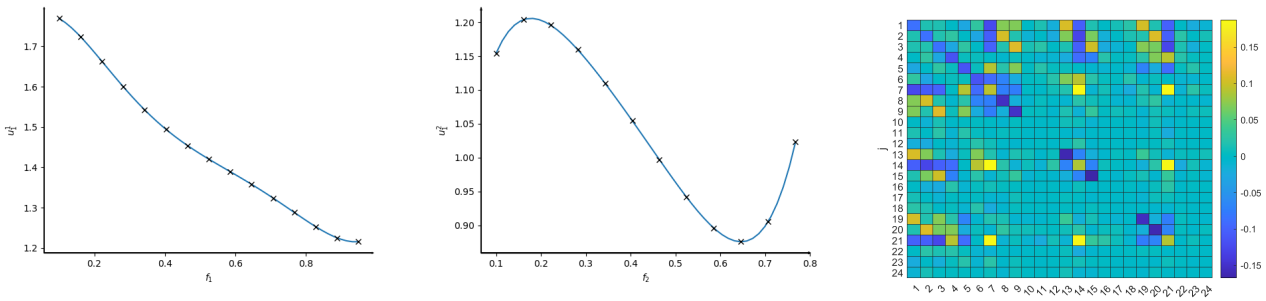
where, again, the feature factor vectors $\mathbf{v}_r^k$, with $k=1,\ldots,n_b$, are restricted to be the evaluations of polynomial or rational functions $v_r^k(f_k)$. Due to the very smooth behaviour of the joint matrix norm, a low degree and a limited number of terms R_2 is required to represent it with sufficient accuracy. In this paper, around ten terms and fifth-degree polynomials are used. The functional approximation $I_2(f_1,\ldots,f_{n_b})$ for the reduced joint matrix norm $\|\mathbf{L}(j_1,\ldots,j_{n_b})\|_2$ is then obtained

$$\begin{aligned}\|\mathbf{L}(j_1,\ldots,j_{n_b})\|_2 &\approx \sum_{r=1}^{R_2} v_r^1(f_1)\cdot\cdots\cdot v_r^{n_b}(f_{n_b})\\ &=: I_2(f_1,\ldots,f_{n_b})\end{aligned}$$

and the approximation of the reduced joint matrix $\mathbf{L}(f_1,\ldots,f_{n_b})$ can be written as

$$\mathbf{L}(f_1,\ldots,f_{n_b}) \approx \mathbf{I}_1(f_1,\ldots,f_{n_b})\cdot I_2(f_1,\ldots,f_{n_b}).$$

Figure 20 visualizes a basis matrix and two factor vectors with corresponding factor functions of an exemplar matrix-level approximation of a corner joint. The factor vectors are interpolated exactly by the factor functions since this is a constraint enforced during the least-squares optimization.



(a) Values of $\mathbf{u}_1^1$ and its interpolation $u_1^1(f_1)$, a polynomial of degree 6.

(b) Values of $\mathbf{u}_1^2$ and its interpolation $u_1^2(f_2)$, a polynomial of degree 6.

(c) Third basis matrix $\mathbf{K}_3^b$.

Fig. 20: Visualization of the matrix-level CPD approximation for a corner joint in a diamond lattice.

Entry-level approximation While the matrix-level approximator simultaneously constructs an approximation for all entries inside $\mathbf{L}$, the entry-level method approximates every entry separately, similarly to how the matrix-level method approximates the matrix norm.

Consider again the reduced representation of the joint stiffness matrices $\mathbf{L}(j_1,\dots,j_{n_b})$. For every lower-diagonal entry (m,n), $\binom{n_r+1}{2}$ in total, a tensor $\mathbf{T}_{m,n}\in\mathbb{R}^{N_1\times\cdots\times N_{n_b}}$ is constructed such that

$$\mathbf{T}_{m,n}(j_1,\dots,j_{n_b}) = \mathbf{L}(j_1,\dots,j_{n_b})[m,n].$$

The tensor $\mathbf{T}_{m,n}$ is then decomposed as

$$\mathbf{T}_{m,n} \approx \sum_{r=1}^{R_{m,n}} \mathbf{w}_{r,m,n}^1 \otimes\cdots\otimes \mathbf{w}_{r,m,n}^{n_b},$$

where $\mathbf{w}_{r,m,n}^k$ are feature factor vectors indexed by the feature dimension index $k=1,\dots,n_b$, the CPD term index $r=1,\dots,R_{m,n}$ and the indices into the lower-triangular part of $\mathbf{L}$, m and n. The feature factor vectors are again restricted to be the evaluations of rational functions $w_{r,m,n}^k(f_k)$. Furthermore, entries are known to become zero when their corresponding beam has zero radius. The corresponding factors are therefore restricted to have that property as well. Based on our experiments, around 15 terms, a sixth-degree numerator and a second-degree denominator suffice to obtain sufficient accuracy. The approximation $I_{m,n}(f_1,\dots,f_{n_b})$ for the (m,n) entry of the reduced joint matrix representation $\mathbf{L}(l_1,\dots,l_{n_b})[i,j]$ is then obtained:

$$\begin{aligned}\mathbf{L}(l_1,\dots,l_{n_b})[m,n] &\approx \sum_{r=1}^{R_{m,n}} w_{r,m,n}^1(f_1)\cdot\cdots\cdot w_{r,m,n}^{n_b}(f_{n_b})\\ &=: I_{m,n}(f_1,\dots,f_{n_b}).\end{aligned}$$

References

1. D. Amsallem and C. Farhat. Interpolation method for adapting reduced-order models and application to aeroelasticity. *AIAA Journal*, 46(7):1803–1813, 2008. ISSN 00011452. doi: 10.2514/1.35374.
2. M. F. Ashby. The properties of foams and lattices. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 364 (1838):15–30, 2006. ISSN 1364503X. doi: 10.1098/rsta.2005.1678.
3. M. P. Bendsøe and Noboru Kikuchi. Generating optimal topologies in structural design using a homogenization method. *Computer Methods in Applied Mechanics and Engineering*, 71:197–224, 1988. ISSN 00457825. doi: 10.1016/0045-7825(88)90086-2.
4. M. P. Bendsøe and O. Sigmund. *Topology Optimization*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003. ISBN 978-3-642-07698-5. doi: 10.1007/978-3-662-05086-6.
5. P. Breiding, F. Gesmundo, M. Michałek, and N. Vannieuwenhoven. Algebraic compressed sensing, 2021. URL `http://arxiv.org/abs/2108.13208`.
6. J. Christensen and F. J. G. de Abajo. Anisotropic Metamaterials for Full Control of Acoustic Waves. *Phys. Rev. Lett.*, 108(12):124301, 2012. doi: 10.1103/PhysRevLett.108.124301.
7. T. A. Davis. Algorithm 832: UMFPACK V4.3—an Unsymmetric-Pattern Multifrontal Method. *ACM Trans. Math. Softw.*, 30(2):196–199, 2004. ISSN 0098-3500. doi: 10.1145/992200.992206.
8. Deepak S A, Dushyanthkumar G L, and Rajesh Shetty. Classical and Refined Beam and Plate Theories: A Brief Technical Review. *International Journal of Research*, 4(September), 2018.
9. G. Dong and Y. F. Zhao. Numerical and experimental investigation of the joint stiffness in lattice structures fabricated by additive manufacturing. *International Journal of Mechanical Sciences*, 148(September): 475–485, 2018. ISSN 00207403. doi:

10.1016/j.ijmecsci.2018.09.014.
10. G. Dong, Y. Tang, and Y. F. Zhao. A survey of modeling of lattice structures fabricated by additive manufacturing. *Journal of Mechanical Design, Transactions of the ASME*, 139(10), 2017. ISSN 10500472. doi: 10.1115/1.4037305.
11. W. Erdelyi, Hunor and Remouchamps, Alain and Donders, Stijn and Farkas, Laszlo and Liefooghe, Christophe and Craeghs, Tom and Van Paepegem. Lattice structure design for additive manufacturing based on topology optimization. In *NAFEMS*, 2017.
12. G. A. Francfort and F. Murat. Homogenization and optimal bounds in linear elasticity. *Archive for Rational Mechanics and Analysis*, 94(4):307–334, 1986. ISSN 00039527. doi: 10.1007/BF00280908.
13. G. Gastin. Forthbridge feb 2013, 2013. URL `https://creativecommons.org/licenses/by-sa/3.0/legalcodehttps://commons.wikimedia.org/wiki/File:Forthbridge{_}feb{_}2013.jpg`.
14. C. Geuzaine and J. Remacle. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331, 2009. ISSN 0029-5981.
15. GrabCAD. Airplane Bearing Bracket Challenge — Engineering & Design Challenges — GrabCAD, 2016. URL `https://grabcad.com/challenges/airplane-bearing-bracket-challenge`.
16. M. Helou and S. Kara. Design, analysis and manufacturing of lattice structures: An overview. *International Journal of Computer Integrated Manufacturing*, 31(3):243–261, 2018. ISSN 13623052. doi: 10.1080/0951192X.2017.1407456.
17. F. L. Hitchcock. The Expression of a Tensor or a Polyadic as a Sum of Products. *Journal of Mathematics and Physics*, 6(1-4):164–189, 1927. doi: https://doi.org/10.1002/sapm192761164.
18. C. Imediegwu, R. Murphy, R. Hewson, and M. Santer. Multiscale structural optimization towards three-dimensional printable structures. *Structural and Multidisciplinary Optimization*, 60(2):513–525, aug 2019. ISSN 16151488. doi: 10.1007/s00158-019-02220-y.
19. S. Johnston. The guts of the Forth Bridge - geograph.org.uk - 1318782. URL `https://commons.wikimedia.org/wiki/File:The{_}guts{_}of{_}the{_}Forth{_}Bridge{_}-{_}geograph.org.uk{_}-{_}1318782.jpghttps://creativecommons.org/licenses/by-sa/2.0/legalcode`.
20. G. N. Labeas and M. M. Sunaric. Investigation on the static response and failure process of metallic open lattice cellular structures. *Strain*, 46(2):195–204, 2010. ISSN 00392103. doi: 10.1111/j.1475-1305.2008.00498.x.
21. K. Lietaert, A. Cutolo, D. Boey, and B. Van Hooreweder. Fatigue life of additively manufactured Ti6Al4V scaffolds under tension-tension, tension-compression and compression-compression fatigue load. *Scientific Reports*, 8(1):1–9, 2018. ISSN 20452322. doi: 10.1038/s41598-018-23414-2.
22. M. H. Luxner, J. Stampfl, and H. E. Pettermann. Finite element modeling concepts and linear analyses of 3D regular open cell structures. *Journal of Materials Science*, 40(22):5859–5866, 2005. ISSN 00222461. doi: 10.1007/s10853-005-5020-y.
23. T. Maconachie, M. Leary, B. Lozanovski, X. Zhang, M. Qian, O. Faruque, and M. Brandt. SLM lattice structures: Properties, performance, applications and challenges, 2019. ISSN 18734197.
24. A. G. M. M. M.C.E. LVIII. The limits of economy of material in frame-structures. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 8(47):589–597, 1904. doi: 10.1080/14786440409463229.
25. A. Panesar, M. Abdi, D. Hickman, and I. Ashcroft. Strategies for functionally graded lattice structures derived using topology optimisation for Additive Manufacturing. *Additive Manufacturing*, 19:81–94, 2018. ISSN 22148604. doi: 10.1016/j.addma.2017.11.008.
26. H. Panzer, J. Mohring, R. Eid, and B. Lohmann. Parametric Model Order Reduction by Matrix Interpolation. *At-Automatisierungstechnik*, 58(8): 475–484, 2010. ISSN 01782312. doi: 10.1524/auto.2010.0863.
27. X. Pennec, P. Fillard, and N. Ayache. A riemannian framework for tensor computing. *International Journal of Computer Vision*, 66(1):41–66, 2006. ISSN 09205691. doi: 10.1007/s11263-005-3222-z.
28. J. N. Reddy, E. Ruocco, J. A. Loya, and A. M. Neves. Theories and analysis of functionally graded beams. *Applied Sciences (Switzerland)*, 11(15):1–24, 2021. ISSN 20763417. doi: 10.3390/app11157159.
29. R. Romeo and R. Schultz. *Model Reduction of Self-Repeating Structures with Applications to Metamaterial Modeling*, pages 49–62. 2020. ISBN 978-3-030-12242-3. doi: 10.1007/978-3-030-12243-0_9.
30. Siemens Digital Industries Software. *Simcenter Nastran Superelement User's Guide*. 2019.
31. Siemens Digital Industries Software. Simcenter 3D, version 2021.2, 2021. URL `https://www.plm.automation.siemens.com/global/en/products/simcenter/simcenter-3d.html`.
32. M. Smith, Z. Guan, and W. J. Cantwell. Finite element modelling of the compressive response of lattice structures manufactured using the selective laser melting technique. *International Journal of Mechanical Sciences*, 67:28–41, 2013. ISSN 00207403. doi: 10.1016/j.ijmecsci.2012.12.004.
33. L. Sorber, M. Van Barel, and L. Lathauwer. Structured Data Fusion. *Selected Topics in Signal Processing, IEEE Journal of*, 9:586–600, 2015. doi: 10.1109/JSTSP.2015.2400415.
34. J. Speet. *Parametric reduced order modeling of structural models by manifold interpolation techniques: Application on a jacket foundation of an offshore wind turbine*. PhD thesis, Delft University of Technology, 2017.
35. T. Tancogne-Dejean, A. B. Spierings, and D. Mohr. Additively-manufactured metallic micro-lattice materials for high specific energy absorption under

static and dynamic loading. *Acta Materialia*, 116:14–28, 2016. ISSN 13596454. doi: 10.1016/j.actamat.2016.05.054.
36. L. Varga. Transformation procedures to accelerate Finite Element analyses. *Periodica Polytechnica Transportation Engineering*, 15(2):185–199, 1987. doi: N/A. URL `https://pp.bme.hu/tr/article/view/6749`.
37. N. Vervliet, O. Debals, L. Sorber, M. Van Barel, and L. De Lathauwer. Tensorlab 3.0, 2016. URL `https://www.tensorlab.net`.
38. J. Wu, O. Sigmund, and J. P. Groen. Topology optimization of multi-scale structures: a review. *Structural and Multidisciplinary Optimization*, 2021. ISSN 16151488. doi: 10.1007/s00158-021-02881-8.
39. J. Wu, W. Wang, and X. Gao. Design and Optimization of Conforming Lattice Structures. *IEEE Transactions on Visualization and Computer Graphics*, 27 (1):43–56, 2021. ISSN 19410506. doi: 10.1109/TVCG.2019.2938946.
40. Z. Wu, L. Xia, S. Wang, and T. Shi. Topology optimization of hierarchical lattice structures with substructuring. *Computer Methods in Applied Mechanics and Engineering*, 345:602–617, 2019. ISSN 00457825. doi: 10.1016/j.cma.2018.11.003.
41. L. Xia and P. Breitkopf. Recent Advances on Topology Optimization of Multiscale Nonlinear Structures. *Archives of Computational Methods in Engineering*, 24 (2):227–249, apr 2017. ISSN 18861784. doi: 10.1007/s11831-016-9170-7.
42. L. Xiao, S. Li, W. Song, X. Xu, and S. Gao. Process-induced geometric defect sensitivity of Ti–6Al–4V lattice structures with different mesoscopic topologies fabricated by electron beam melting. *Materials Science and Engineering A*, 778(February): 139092, 2020. ISSN 09215093. doi: 10.1016/j.msea.2020.139092.
43. K. Xu, D. Z. Huang, and E. Darve. Learning constitutive relations using symmetric positive definite neural networks. *Journal of Computational Physics*, 428: 110072, 2021. ISSN 0021-9991. doi: 10.1016/j.jcp.2020.110072.
44. O. C. Zienkiewicz, R. L. Taylor, and J. Z. Zhu. *The finite element method: its basis and fundamentals*. Elsevier, 2005.
45. T. I. Zohdi and P. Wriggers. *An Introduction to Computational Micromechanics, Corrected Second Printing*. Number 20. Springer, 2005. ISBN 978-3-540-77482-2\r978-3-540-32360-0. doi: 10.1007/978-3-540-32360-0.