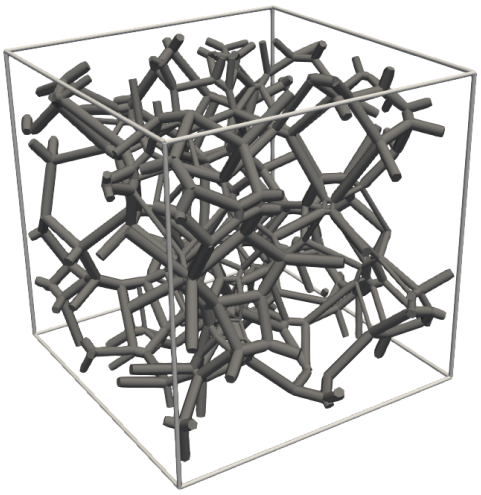
Accelerating Multiscale Simulations of Open-Porous Materials using Machine Learning-Based Surrogate Models



Lucas F. Mager

Accelerating Multiscale Simulations of Open-Porous Materials using Machine Learning-Based Surrogate Models

---

Inaugural-Dissertation

zur

Erlangung des Doktorgrades

der Mathematisch-Naturwissenschaftlichen Fakultät

der Universität zu Köln

UNIVERSITÄT
ZU KÖLN

vorgelegt von

Lucas Fabian Mager

aus Mainz

Köln | 2026

Berichterstatter: Prof. Dr. Axel Klawonn
(Universität zu Köln)

Prof. Dr. Oliver Rheinbach
(TU Bergakademie Freiberg)

Datum der Disputation: 20. April 2026

Abstract

Open-porous materials, such as aerogels, have exceptional thermal insulation properties, making them ideal for applications involving extreme temperatures. Their combination of high-performance insulation properties and very low density is beneficial for different aerospace applications. The macroscopic mechanical behavior of these materials is closely connected to their complex, fibrillar nanostructure. However, it is not computationally feasible to predict this behavior using direct numerical simulation due to the great difference between the structural length scales. Consequently, multiscale homogenization schemes are required to bridge the gap between the nanostructure and the macroscopic geometry.

This thesis presents a computational framework for the efficient multiscale simulation of open-porous materials, with a specific focus on biopolymer aerogels. The baseline method utilizes an FE^2 approach, in which the macroscopic geometry is discretized using the finite element method (FEM), and the microscopic response is determined by computing the equilibrium of a representative volume element (RVE). To accurately capture the fibrillar nanostructure, the RVEs are modeled as frame structures based on Euler-Bernoulli beam elements. These RVEs are created using a Laguerre-Voronoi tessellation based on experimentally measured pore size distributions.

While the FE^2 method provides a robust basis for the simulation of open-porous materials, repeatedly computing the solution of the microscopic boundary value problem at every macroscopic integration point requires a significant computational effort. To overcome this limitation and reduce the computation time, this work introduces machine learning-based surrogate models to replace the expensive mi-

croscopic solver. Different neural network (NN) architectures are developed and evaluated, ranging from dense feed-forward neural networks trained on specific material data to more versatile graph neural networks (GNNs) processing the graph-like nanostructure of varying RVEs. Furthermore, a Mixture of Experts (MoE) approach is proposed to create a generalized surrogate model that is adaptable to different open-porous materials. Numerical results demonstrate that the surrogate models significantly accelerate the simulations while maintaining high accuracy compared to the direct beam frame solver.

A parallel implementation of the multiscale solver achieves an additional reduction in computation time. To validate the multiscale simulation approach, the simulation results are compared to experimental data obtained from mechanical testing and Digital Image Correlation (DIC) of biopolymer aerogel specimens. Validating the approach against these experimental results supports its applicability to real-world material design and analysis.

Zusammenfassung

Durch ihre außergewöhnlichen Materialeigenschaften eignen sich Aerogele und ähnliche offenporige Materialien ideal für Anwendungen, bei denen sie extremen Temperaturen ausgesetzt sind. Speziell für Projekte aus dem Bereich der Luft- und Raumfahrt liefern ihre starken Dämmeigenschaften zusammen mit der niedrigen Dichte klare Vorteile für die Anwendung. Das mechanische Verhalten dieser Materialien ist hierbei hauptsächlich durch ihre fibrillare Nanostruktur bestimmt. Dieses Verhalten mit klassischen numerischen Methoden und einer vollen Auflösung der Nanostruktur zu simulieren, wäre jedoch rechnerisch nicht praktikabel. Der enorme Unterschied in der Größenordnung zwischen der makroskopischen Geometrie und der Nanostruktur würde eine Auflösung der Strukturen verlangen, die nicht realisierbar wäre. Um die Abhängigkeiten zwischen der Nanostruktur und der makroskopischen Geometrie in der Simulation zu berücksichtigen sind somit Multiskalen-Methoden erforderlich.

Diese Arbeit stellt eine effiziente Methode zur Multiskalen-Simulation offenporiger Materialien vor, wobei ein besonderer Fokus auf die Vorhersage des mechanischen Verhaltens von Biopolymer-Aerogelen gelegt wird. Das grundlegende Modell basiert dabei auf der FE^2-Methode. In diesem Multiskalen-Ansatz wird die makroskopische Geometrie mit Hilfe der Finite-Elemente-Methode (FEM) diskretisiert und auf der mikroskopischen Ebene bestimmt ein repräsentatives Volumenelement (RVE) das Materialverhalten. Für die fibrillare Nanostruktur eines Aerogels lässt sich das RVE als Modell aus verbundenen Balkenelementen darstellen. Diese Balkenstruktur wird unter Verwendung einer Laguerre-Voronoi-Tessellierung auf Basis experimenteller Porengrößenverteilungen konstruiert.

Die FE^2-Methode erfordert das wiederholte Lösen des mikroskopischen Randwertproblems an jedem makroskopischen Integrationspunkt. Besonders für komplexe Nanostrukturen stellt dies einen erheblichen Rechenaufwand dar. Um diesen Aufwand zu reduzieren und die Rechenzeit der Simulationen zu verringern, werden Surrogat-Modelle eingeführt, die den aufwendigen mikroskopischen Löser ersetzen sollen. Diese Modelle werden auf der Grundlage von Ansätzen des maschinellen Lernens darauf trainiert, das Verhalten des Balkenmodells nachzubilden. Hierfür werden neuronale Netze (NN) auf Basis verschiedener Netzwerk-Architekturen entwickelt und ausgewertet. Die betrachteten Ansätze reichen dabei von simplen vorwärtsgerichteten neuronalen Netzen (feed-forward NN), die auf ein spezifisches Materialverhalten trainiert wurden, bis hin zu vielseitigen Graph Neural Networks (GNNs), die in der Lage sind, die Nanostruktur unterschiedlicher RVEs zu verarbeiten. Darüber hinaus wird ein Mixture-of-Experts-Ansatz (MoE) entwickelt, der die Möglichkeit bietet, Materialeigenschaften als zusätzliche Eingabe dem Surrogat-Modell zu übergeben. Das MoE-Modell kann sich dabei für die Simulation an verschiedene offenporige Materialien anpassen. Die numerischen Ergebnisse zeigen, dass die Surrogat-Modelle die Rechenzeit für Simulationen deutlich reduzieren können und gleichzeitig eine hohe Genauigkeit im Vergleich zu dem Balken-Modell erzielen.

Für eine zusätzliche Reduzierung der Rechenzeit wird eine parallele Implementierung der Multiskalen-Methode vorgestellt und diese bezüglich der Skalierbarkeit bewertet. Zur Validierung der Methode werden in der Arbeit Simulationsergebnisse verglichen mit experimentellen Daten, die aus mechanischen Tests und der digitalen Bildkorrelation (DIC) gewonnen wurden. Für die Experimente wurden verschiedene Proben von Biopolymer-Aerogelen verwendet. Die Validierung des Ansatzes bezüglich dieser experimentellen Ergebnisse demonstriert außerdem die Eignung der Multiskalen-Methode für die Anwendung in der Materialforschung.

Acknowledgments

On my journey in science that brought me to writing this PhD thesis, I was influenced and supported by a lot of people. I am grateful for this support and the encouragement to follow this path. First of all, I would like to thank my advisor Professor Axel Klawonn for the opportunity to do my PhD as a part of his research group and to actively contribute to current research topics through my work. I would also like to express my gratitude to my mentor Dr. Ameya Rege, who provided support through his extensive expertise in the field of open-porous materials and their mechanical properties. Furthermore, I am grateful to Professor Oliver Rheinbach for reviewing this thesis.

I would particularly like to highlight the influence of Martin Lanser and Janine Weber-Hamacher. Discussions with both of them on various topics in numerical mathematics and scientific machine learning have always broadened my understanding of the issues concerned. Furthermore, my gratitude goes to my other colleagues in the research group at the University of Cologne, namely Tommaso Bevilacqua, Victor Grimm, Jascha Knepper, Natalie Kubicki, Kyrill Ho, Sabine Musilack-Erle, Lea Saßmannshausen, Colin Schwanke, and Adam Wasiak, who contributed to a vibrant exchange of knowledge and a pleasant working atmosphere.

I owe special thanks to Simon Klaes, whose work provides the basis for many visualizations in this thesis and to Nikita Hust for the development of an automated pipeline for the generation of beam frame structures.

I gratefully acknowledge the funding of my project by the German Aerospace Center (DLR) in Cologne. Here, I would like to highlight the Institute for Frontier Materials on Earth and in Space with the department of Professor Barbara Milow.

Special thanks to Shivangi Aney, who provided the code basis for the creation of beam frame structure based on the Laguerre-Voronoi tessellation. Furthermore, I owe special thanks to Lennart Barth, Eric Breitbarth, Eric Dietrich, Bruno Gonzales, and Max Zinke who provided the experimental data presented in this work. My gratitude goes to my colleagues of the research group working on the modeling and simulation of aerogels, namely Nina Borzecka, Sugan Kanagasenthinathan, Anastasiia Melnyk, Prakul Pandit, and Hemangi Patel.

Over the past four years of my PhD studies, I have received great emotional support from my family and friends. I owe special thanks to my parents Sabine Thelen-Mager and Norbert Mager, who always supported my decisions and stood by my side. Furthermore, I would like to express my special thanks to my girlfriend Hannah Bornhofen, who provided emotional support in good and in difficult times. Especially during stressful times, I knew that I could count on her support.

I gratefully acknowledge the supply of computing capacity of the Center for Data and Simulation Science (CDS) at the University of Cologne as well as of the compute cluster *Fritz* at Friedrich-Alexander-Universität Erlangen-Nürnberg.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the number of rocket launches worldwide steadily on the rise [37], the relevance of materials research in the field of high-performance insulation applications is also growing. The development of materials for aerospace operations is especially challenging due to the extreme conditions to which the objects are exposed. The high velocity of rockets traveling within Earth's atmosphere leads to significant aerodynamic heating due to friction energy. Rockets must be able to withstand these high temperatures without sustaining damage. Furthermore, passenger airplanes are exposed to large temperature fluctuations. Even in low atmospheric layers, where aircraft travel, the temperature usually drops as low as -55°C.

Due to the extreme temperatures to which most aerospace vessels, such as rockets, aircraft, and satellites, are exposed, these units require highly effective insulation. The insulation materials for these applications must primarily combine three important properties. The major feature of the material is evidently the thermal insulation required to withstand high temperatures. In some cases, the material is also expected to possess good acoustic insulation properties. The second important feature is that the material must be lightweight. In aerospace applications, the weight of the vessel is often crucial, and a primary objective for engineers is to use materials of low density. In addition to the already mentioned characteristics, the mechanical properties are also of great importance. The material must be resistant to high pressure and must sustain potential deformation without failing or losing

its insulating properties.

For aerospace applications, aerogel components in particular have emerged as insulation materials with outstanding properties. Aerogels are open-porous solids that feature a nanostructure, which provides the material with particularly good insulating properties combined with very low weight. Various research institutions, such as the German Aerospace Center (DLR) and the National Aeronautics and Space Administration (NASA), have investigated the use of aerogels for aerospace applications for many years [44]. Due to the described advantages, this open-porous material has already been used as insulation in a wide range of applications, including the Mars Rover and space suits for astronauts [63].

Besides the field of aerospace, there are many other applications which require lightweight materials with strong thermal or acoustic insulation. Effective insulation is a key factor in many industrial applications. In addition to industrial use cases, some consumer goods utilizing aerogel-insulated fabric have emerged. In the outdoor sector, jackets use aerogel to protect electronic devices from extreme temperatures. Even footwear and sleeping bags have been developed with aerogel insulation [92]. Additionally, the application of aerogels as insulation material for battery cells is an important research topic regarding the development of electric vehicles [71, 78].

The definition of an aerogel is highly discussed in the field of materials science. However, the term is usually used to refer to a category of open-porous solids produced by sol-gel chemistry and supercritical drying. These materials are characterized by their porous fraction and in most cases, aerogels contain more than 90% air by volume [140]. This property results in a very low density, typically between 0.0011 and 0.5 g/cm^3, and a thermal conductivity of less than 0.020 $W/(m \cdot K)$ [140]. Consequently, the thermal conductivity is lower than that of conventional insulation materials such as polystyrene and polyurethane.

Considering the sol-gel process, various compositions are possible for the synthesis of aerogels. Silica-based aerogels are the most commonly used in applications and this type of aerogel has been well covered in scientific literature. However, it is generally possible to synthesize aerogels from a wide range of polymers. Therefore,

aerogels based on carbon, biopolymers, or metal oxides have also gained scientific popularity. The choice of the base material and variations during the synthesis process, such as different ratios of the suspension used for the sol-gel process, strongly affect the nanostructure of the resulting aerogel. The porous structure of the materials also determines key material properties like thermal conductivity, sound absorption, and mechanical behavior [124]. Regarding the mechanical properties, the behavior of aerogels can range from elastic and spongy to stiff and even brittle. This implies that different materials offer different application possibilities.

As mentioned, the nanostructure of aerogels is strongly influenced by the base material and the synthesis parameters. The characteristic nanostructure of an open-porous material can be analyzed using a scanning electron microscope (SEM) or gas adsorption measurements. Regarding gas adsorption methods, it is possible to determine the specific surface area using the Brunauer-Emmett-Teller (BET) method [19] or to measure the pore size distribution using the Barrett-Joyner-Halenda (BJH) model [7]. The distribution of the pore sizes is an important characteristic of open-porous materials. For aerogels, pore sizes typically range from less than two nanometers to over hundreds of nanometers within the same material [48].

The primary objective of this work is the development of computational methods for the simulation of mechanical material behavior. Although the developed simulation approaches are applicable to a wide range of open-porous materials, the main motivation for developing these methods originates from computing the deformation of aerogel materials based on biopolymers. The backbone of these aerogels is built from molecular chains based on biopolymers like cellulose, carrageenan, pectin, alginate, or chitosan. The respective molecular chains form the fibrillar nanostructure which is typical for these materials. Biopolymers are widely available from natural and renewable sources. Aerogels based on these materials are biodegradable, which reduces the environmental impact of disposal. Food packaging is an example of an everyday application where, in addition to good insulation properties, environmental concerns play an increasingly important role. By reducing the use of plastics, biopolymer aerogels could offer a meaningful alternative for food packaging with good insulating properties. Further approaches to produce

biopolymer aerogels from agricultural and food waste [116] also offer great potential for lowering production costs in an environmentally friendly way.

In addition to the base material used for synthesis, the configuration of the nanoporous structure is decisive for the mechanical properties of an aerogel. Therefore, considering the material's nanostructure is crucial for simulating the mechanical behavior of the open-porous solid. Research into the deformation of open pore structures, such as those found in aerogels, dates back to the mid-20th century. The research conducted by Gent and Thomas in 1959 was among the first to describe the deformation of open-porous materials [54]. This work considered foams, which feature a structure very similar to that of aerogels. In the work of Gibson and Ashby [56, 57], a more robust way of modeling the pore structure was introduced. They described the elastic deformation of the pores by modeling each fibril as an Euler-Bernoulli beam element. The work on plastic deformation of foam structures and the introduced strain energy approach by Dement'ev and Tarakanov [29] provided an important basis for the subsequent research of Rege et al. [125] on a generalized micromechanical constitutive model. In further studies, an extension of the model introduced by Rege showed good validation results for biopolymer aerogels [123]. For describing the mechanical behavior of open-porous materials, each of the mentioned research articles considers the deformation of only a single pore with its pore wall structure. Considering the nanostructure of an aerogel with different combinations of pore sizes ranging from a few to hundreds of nanometers, the interaction of different pores within the full fibrillar structure can be significant for the mechanical properties of the material.

In order to better describe and analyze the structural composition of heterogeneous materials, the representative volume element (RVE) has been established since its introduction by Hill [68]. For a given heterogeneous material, the RVE is a volume usually defined within a domain that is small compared to the respective macroscopic component. The RVE is expected to be a statistically meaningful sample of the heterogeneous structure of the considered medium. Therefore, it is supposed to feature properties similar to those of the material. For the construction of RVEs that model the nanostructure of open-porous materials, the pore structure

can be reconstructed from computed tomography (CT) images [75, 51, 138, 46]. This approach yields highly accurate results, though, for complex nanostructures with small pores, this process becomes very challenging and time-consuming.

For a fast and efficient method of designing the RVE for open-porous materials, a widely used approach is based on the computation of Voronoi cells. This method was introduced in [38] for the modeling of polycrystalline structures. In the case of open-porous materials, the RVE is constructed by applying a sphere packing algorithm [39] on a specific number of sphere diameters following the pore size distribution of a given porous material. Based on the sphere packing, the results of a Laguerre-Voronoi tessellation [73] shape the porous structure of the RVE, with the boundaries of the Voronoi cells forming the respective backbone. For modeling the nanostructure of biopolymer aerogels, the Voronoi approach has shown good results regarding the mechanical behavior [120, 22].

For processes which can be described by partial differential equations (PDEs), the finite element method (FEM) [13, 72] is one of the best-known simulation methods. It is widely used in many scientific fields, including solid mechanics. However, for computing the deformation behavior of heterogeneous materials like aerogels, the application of the FEM is not straightforward. For materials with a heterogeneous nano- or microstructure, it is often not feasible to develop a constitutive model which accurately describes the macroscopic deformation behavior. In the case of aerogels, the nanostructure significantly influences the mechanical behavior. Therefore, the applicability of mechanical models which only consider the macroscopic scale is often limited. On the other hand, fully resolving the microstructure for a large-scale simulation is usually also not feasible, as the required computational effort would be enormous.

Regarding the simulation of materials that feature complex structural properties, multiscale approaches have proven beneficial in a wide range of applications. The most popular multiscale methods include the FE^2 method [45, 108, 135, 139, 134, 90], the Heterogeneous Multiscale Method (HMM) [2], and the Multiscale Finite Element Method (MsFEM) [70]. For the scope of this work, we base our simulation approaches on the FE^2 method, as this method is well suited for the mechanical

simulation of solids and fits our applications.

The FE^2 method is a multiscale simulation approach based on the FEM. It has become a well-established approach for mechanical analyses of heterogeneous materials and has been employed in a wide range of applications such as dual-phase steel [86, 81], lithium-ion battery cells [131], hyperelastic, heterogeneous materials [14], and biological tissue [15]. In its original form, the FE^2 approach considers a macroscopic and a microscopic scale, both of which are discretized using finite elements. On the microscopic scale, an RVE is considered, which is deformed according to the deformation gradient on the macroscopic scale. The coupling of the two scales is usually defined by the Hill-Mandel condition [26, 68, 67]. The constraint induced by the Hill-Mandel condition ensures that the energy computed at the microscopic scale is consistent with the macroscopic energy.

To compute the deformation of open-porous materials for larger workpieces, it is not feasible to fully resolve the porous nanostructure of the geometry. Modeling and solving such a system would not be practicable, even on large, modern computing clusters. Therefore, the application of multiscale approaches is particularly suitable for these problems. To simulate the deformation behavior of open-porous materials such as biopolymer aerogels, the multiscale approach presented in this work adopts the general micro–macro coupling of the well-known FE^2 method. In our approach, the RVE at the microscopic scale is modeled as a beam frame. This modeling of the fibrillar structure with beam elements follows the insights of Gibson and Ashby [56, 57] and the works of Rege et al. [122, 123]. Modeling the fibrillar structure with beam elements also offers the advantage of substantially lower computation costs compared with fully resolving the nanostructure using solid finite elements.

For setting up the multiscale approach for a given open-porous material, the RVE on the microscopic scale is constructed by the Voronoi approach from the respective pore-size distribution and porous fraction as described in [38]. On the macroscopic scale, the FEM is considered for solving the momentum balance equation [69, 105]. However, as the FE^2 method suggests, the relationship between deformation and stress is not determined by a material law, but rather by solving the microscopic beam frame problem and homogenizing the stresses within the microscopic RVE.

This means that for computing a single iteration step of a macroscopic solver, the multiscale method requires solving a beam frame problem and computing the homogenized stress values in each Gaussian quadrature point of the macroscopic finite element mesh.

For large macroscopic problems and complex RVEs, computing the homogenized stresses from the microscopic problems in each Gaussian quadrature point becomes computationally demanding. The high computational workload is a well-known limitation of the FE^2 method and there are different approaches on how to reduce the computational effort. For reducing the computation time, it is possible to run significant parts of the multiscale method in parallel on multiple compute units. Since the microscopic problems in each Gaussian quadrature point are independent of each other, it is possible to run the computations on the microscopic scale fully in parallel. Therefore, the FE^2 method shows very good scalability, although it cannot be considered embarrassingly parallel due to the limitations of parallelizing the macroscopic computations. Consequently, using parallel implementations of the FE^2 method has been a common practice for reducing the computation time for multiscale simulations [111, 86, 81]. For the multiscale approach described in this work, we consider scalability tests for evaluating the performance of an implementation based on the PETSc framework [6, 5].

In addition to parallel implementations of the FE^2 method, the load of the microscopic computations can be reduced by replacing the original microscopic model with a surrogate model. This surrogate model is expected to approximate the outcome of the original model while requiring much less computational effort. The surrogate models used in the multiscale framework are often based on reduced order models (ROM) [96, 50] or based on machine learning (ML) models. For replacing the finite element simulation on the microscopic scale, machine learning-based surrogate models in particular have shown good results with regard to flexibility and accuracy [12, 113]. The various applications of these models cover a wide range of mechanical problems, including elastic material behavior [97, 34, 36], elastoplastic deformations [50, 52, 127, 126], thermomechanics [53], and damage models [30]. Based on the problem definition, different model architectures can be considered for

a machine learning-based surrogate model. This includes architectures such as feed-forward models [62, 52, 36, 42], recurrent neural networks [30], and convolutional neural networks (CNNs) [34].

The field of scientific machine learning (SciML) describes the combination of traditional numerical methods and machine learning techniques. In this regard, the application of machine learning models for reducing the computation time for mechanical simulations is an important research topic. For complex problems, classic numerical methods are often computationally demanding and surrogate models can overcome these limitations. In addition to data-driven ML models for direct approximation of the simulation results, various ML approaches have been established in recent years, such as physics-informed neural networks (PINNs) [32] and operator learning [100, 152, 91], which yield different advantages for applications. In this work, however, we focus on data-driven approaches for directly predicting the solution vector for given mechanical problems.

For the application of machine learning-based surrogate models in the FE^2 framework, different neural network (NN) architectures can be considered. In the scope of this work, we discuss the use of various ML models and compare the results. The straightforward approach for a machine learning-based surrogate model is to use a data-driven, dense feed-forward neural network [110]. A notable approach for simulating open-porous foam structures with the use of data-driven feed-forward neural networks is presented by Korzeniowski and Weinberg [89]. In the mentioned work, the porous foam structure of the underlying numerical model is fully resolved with finite elements. Data-driven feed-forward NNs are relatively easy to construct and to train. As demonstrated for different applications, these models are capable of learning the mechanical behavior of a given material and geometry [89, 12, 97, 50, 62].

In order to test the usability of different materials for applications in materials research, it is necessary to be able to simulate the respective materials in a computationally efficient manner. For the simulation of biopolymer aerogels, the nanostructure of different materials can vary significantly, and with it their mechanical properties. An important goal with regard to materials research is to develop a

flexible and versatile surrogate model that is capable of simulating the mechanical behavior of different open-porous materials. For constructing a general-purpose surrogate model with the described properties, we consider the graph neural network (GNN) architecture [60, 133], which is capable of processing graph-structured data as model inputs. Many interconnected systems can be described using graphs and the analysis of such structures is a well-established field of mathematics. Therefore, GNNs have proven to be useful in a wide range of applications, such as social sciences [149], natural language processing [103], traffic [154], and chemistry [47, 142].

With regard to the use of surrogate models for mechanical simulations, the GNN model yields large potential for reducing the computation time of particle-based simulations [132] and finite element simulations [61, 137]. In the case of the FEM, the finite element mesh structure is interpreted as an undirected graph and the GNN is trained to predict the mechanical response of each node within the mesh. Based on the approach of using GNN models for the FEM, Hendriks et al. [65] presents the use of the GNN architecture in a multiscale framework. Here, the microscopic computations are replaced with a GNN surrogate model. With regard to the fibrillar structure of the open-porous materials, the considered beam frame RVE is explicitly given as an undirected graph. Due to this graph representation considered on the microscopic scale of the multiscale method, the use of GNN models for predicting the mechanical behavior is a natural choice. By using the nanostructure of the material as additional graph input for the NN, the respective surrogate model can be trained on multiple open-porous materials.

In addition to the GNN architecture, we consider the mixture of experts (MoE) approach [74, 79, 35, 40] for setting up a versatile surrogate model capable of predicting the mechanical behavior of multiple materials. The MoE approach describes a machine learning model that features multiple expert NNs. In the MoE framework, the final output of the model is computed as the weighted sum over the outputs of the expert NNs, with the weights determined by an additional gating NN. In the context of physical simulations, the MoE approach is used for predicting complex input-output correlations [25, 112, 10, 141]. As multiple expert NNs are

included, the method allows the specialization of experts in different aspects of the physical problem. For example, each expert can target a different subdomain of the solution. Additionally, for solving PDEs, the MoE approach can be combined with physics loss terms [25, 10, 141]. This method is also referred to as GatedPINN and was introduced in [141].

Considering the homogenization of open-porous materials, we use the MoE architecture to create a single surrogate model that can be trained on the homogenized stress values of various materials. Our approach considers multiple expert NNs, each of which is trained on the stress data resulting from a single RVE. This means each expert model can be associated with a given material and is trained to reproduce its mechanical behavior. The gating model takes a representation of an open-porous nanostructure as an input and predicts, based on this information, the best combination of expert outputs to determine the resulting stress tensor. Each component of the final result is then computed from the weighted sum over the expert outputs.

An important purpose for the development of computational methods for mechanical simulations is to replace time-demanding and expensive mechanical tests with appropriate simulations. To achieve this objective, the considered methods are required to approximate the real-world behavior of the considered material as well as possible. In general, this is difficult to achieve because setting up a simulation requires significant simplifications in many steps. For example, the considered material model requires assumptions that may not always hold in real-world experiments, and the problem must be discretized for numerical solvers to work.

To evaluate the performance of the multiscale simulation method, we compare the simulation results with deformation and stress data obtained from experimental tests. Different biopolymer aerogels are considered for the mechanical testing. Measuring the pore size distribution and the porous fraction of the materials using gas adsorption methods yields the required parameters for creating the open-porous RVE for the multiscale approach. To reproduce the experimental tests, we consider the macroscopic geometry of the sample obtained from digital image correlation (DIC) [117] and the microscopic nanostructure represented by the RVE. The deviation between the experimental data and the simulation results yields the measure

for evaluating the performance of the approach.

The main body of this work is organized as follows. In Chapter 2, we describe the baseline multiscale method which uses the FE^2 as a basis. We introduce the beam frame model and the corresponding homogenization approach that define the computations on the microscopic scale. Additionally, we introduce the macroscopic boundary value problem and describe the algorithmic procedure of solving the full multiscale problem. In the scope of Chapter 3, we describe the development of machine learning models that are capable of replicating the mechanical behavior of the beam frame model. Here, we focus on dense feed-forward neural networks which are trained on the data resulting from a single RVE. The development of surrogate models for the application in the FE^2 approach is also the topic of Chapter 4. In this part, we consider various more sophisticated machine learning models that use the results of Chapter 3 as a basis. In comparison to the models described in Chapter 3, the surrogate models are trained on the data resulting from multiple RVEs. To achieve this, each of the models requires a representation of the material parameters as input. We discuss the performance of the presented surrogate models and evaluate their behavior with regard to the application in multiscale simulation. In Chapter 5, we describe the parallel implementation of the multiscale simulation approach. We discuss opportunities with regard to parallelization and evaluate the performance of the implementation based on a strong scaling test. Chapter 6 presents a comparison of the numerical simulation results with experimental data. Different experimental setups for mechanical tests are considered. We run simulations based on the experimental data and evaluate the performance of the methods based on the difference to the real-world data. In Chapter 7, we focus on the application of the multiscale simulation approach. This part presents a potential use case for aerogel composites and describes the simulation setup for the given example. We simulate the mechanical behavior of the material for the considered application and discuss the results. To conclude this work, Chapter 8 provides a summary of the key findings. Based on the results, we describe possible research questions for future work.

Chapter 2

Multiscale Modeling of Open-Porous Materials Based on the FE^2 Method

Regarding the material design for applications facing extreme temperature conditions, the use of high-performance insulation materials is crucial. Nanoporous materials like aerogels yield high potential with regard to these applications, as their thermal conductivity is extremely low [140]. Generally, the term aerogel refers to a family of nanoporous solids, which are commonly synthesized from sol-gel chemistry and supercritical drying. Depending on the source materials and the given conditions during synthesis, the open-porous nanostructure of aerogels can significantly vary between different samples. Based on the pore composition, the resulting material features very different properties regarding the thermal conductivity but also regarding the mechanical behavior. The typical open-porous nanostructure of an aerogel is shown in Figure 2.1, which presents an SEM image of an aerogel synthesized from carrageenan. As carrageenan is a biopolymer, this material falls into the group of biopolymer aerogels. Furthermore, biopolymer aerogels can be produced from a wide range of materials, including cellulose, pectin, alginate, and chitosan. The resulting aerogels typically feature a fibrillar nanostructure as presented in Figure 2.1.

The development of simulation methods that describe the mechanical behavior

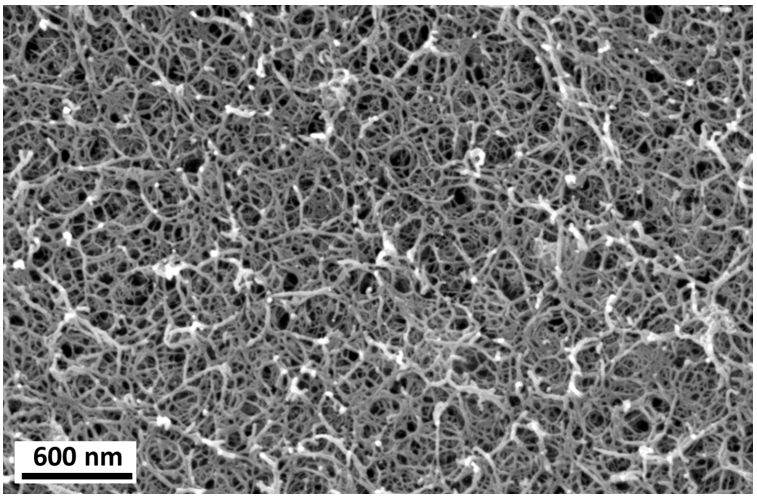


Figure 2.1: SEM image of carrageenan aerogel. The image shows the nanostructure of a biopolymer aerogel synthesized from carrageenan. The figure is taken from [82] and has first been published in [121].

of aerogels and similar open-porous materials is challenging. As the composition of its nanostructure has a crucial impact on the material properties of an aerogel, it is important to incorporate this relationship in an appropriate simulation method. However, if the deformation of aerogel samples is considered, a geometry that fully resolves the nanostructure of the material would be impossible to compute due to the immense computational cost. As the application of traditional numerical approaches is limited, various sophisticated methods can be used to target this challenge. For the simulation of heterogeneous materials like aerogels, multiscale approaches are often considered for the numerical computations.

While different computation schemes can be implemented for multiscale simulations such as the Heterogeneous Multiscale Method (HMM) [2], or the Multiscale Finite Element Method (MsFEM) [70], we use the FE^2 [45, 108, 135] method as a basis for developing our approach regarding the simulation of aerogels. The FE^2 method is a well-known approach for the mechanical simulation of heterogeneous solids. It is well suited for materials with mechanical properties that are strongly influenced by their respective microstructure. Therefore, the FE^2 framework is expected to yield an appropriate basis for the multiscale simulation of aerogels.

In general, the FE^2 approach considers two scales, both of which are discretized with finite elements. Usually, the Hill-Mandel condition [26, 68, 67] is used to prescribe dependencies between the two scales. This constraint is a homogenization principle that ensures that the energy computed at the microscopic scale is consistent with the macroscopic energy. In the homogenization method, which can be derived from the Hill-Mandel condition, boundary value problems on the macroscopic and microscopic scales are described by the balance of linear and angular momentum and suitable boundary conditions on both scales. On the microscopic scale, the energy is homogenized over the domain of a representative volume element (RVE).

As presented in the SEM image in Figure 2.1, the microstructure of biopolymer aerogels is formed as a grid of interconnected fibrils. Each of the fibrils appears relatively thin in relation to its length. Based on this appearance of the material's microstructure, it is natural to model each of the fibrils as beam element. Regarding the mechanical behavior of a single pore, the work of Rege [125, 123] has shown that modeling the fibrillar structure of biopolymer aerogels with beam elements yields a satisfying approximation of the material's real-world behavior. Based on these results, we consider the modeling of the microstructure as a beam frame. This results in an alternative approach based on the FE^2 framework that considers a beam frame model on the microscopic scale.

The scope of this chapter is structured as follows. We provide a general description of the FE^2 method in Section 2.1. Here, we present the general macroscopic and microscopic boundary value problems and describe the coupling of both scales. In Section 2.2, we describe the construction of the beam frame-structures RVE that defines the microscopic problem. The section presents the corresponding system of equations and introduces a homogenization approach for different stress measurements. The focus of Section 2.3 is on the macroscopic scale of the FE^2 method. Here, we present the macroscopic governing equations and we describe different non-linear solver that can be applied to the macroscopic problem. In Section 2.4, we provide an algorithmic representation of the multiscale method that combines the described microscopic and macroscopic computations.

2.1 FE^2 as Multiscale Simulation Approach

The FE^2 method is a computational multiscale approach for the simulation of heterogeneous materials. The application of the FE^2 method is especially beneficial with regard to the simulation of materials with complex microstructures. As it is assumed that the microstructural properties of the materials strongly influence the mechanical behavior on the macroscopic scale, a definition of a constitutive model describing this mechanical behavior of the material is often challenging or impossible.

The term FE^2 refers to the method's key feature, which is the coupling of two finite element problems defined at different length scales. Regarding the mechanical simulation of heterogeneous materials, the macroscopic geometry is considered the structure of interest, as the simulation targets the computation of the deformations on this scale. The macroscopic boundary value problem is usually defined by the governing equations of continuum mechanics, specifically the momentum balance equation [69, chapter 4.3]. With a discretization of the macroscopic geometry using finite elements, it defines the framework for computing the mechanical response on the macroscopic scale. We denote the reference domain of the macroscopic boundary value problem with $\overline{\mathcal{B}}_0$. For each point in the macroscopic geometry that is characterized by its position vector $\overline{\boldsymbol{X}} \in \overline{\mathcal{B}}_0$, we define the deformed position $\overline{\boldsymbol{x}} \in \overline{\mathcal{B}}$. The coordinates of the deformed state are computed by the deformation map $\overline{\varphi} : \overline{\mathcal{B}}_0 \to \overline{\mathcal{B}}$ with $\boldsymbol{X} \mapsto \overline{\varphi}(\overline{\boldsymbol{X}}) = \overline{\boldsymbol{x}}$. The deformed position vector on the macroscopic scale can be expressed as $\overline{\boldsymbol{x}} = \overline{\varphi}(\overline{\boldsymbol{X}}) = \overline{\boldsymbol{X}} + \overline{\boldsymbol{u}}$ in terms of the displacement field $\overline{\boldsymbol{u}}$.

For given macroscopic body force density $\overline{f}_0$ that is applied to the macroscopic geometry, the momentum balance equation is given by the relation

$$\mathrm{Div}_{\overline{\boldsymbol{X}}}\, \overline{P} + \overline{f}_0 = 0 \quad \text{on } \overline{\mathcal{B}_0}$$

with $\overline{P}$ referring to the macroscopic first Piola-Kirchhoff stress tensor. With the macroscopic deformation gradient defined as $\overline{F} = \frac{\partial \overline{\boldsymbol{x}}}{\partial \overline{\boldsymbol{X}}}$, the balance of angular momentum is expressed by $\overline{F} \cdot \overline{P}^{\mathsf{T}} = \overline{P} \cdot \overline{F}^{\mathsf{T}}$.

Based on the momentum balance equation, the macroscopic boundary value problem is commonly expressed in its weak form as

$$\int_{\overline{\mathcal{B}}_0} \delta\overline{\boldsymbol{x}} \left(\mathrm{Div}_{\overline{\boldsymbol{X}}}\, \overline{P} + \overline{f}_0 \right) \mathrm{d}\overline{V} = 0 \tag{2.1}$$

which is obtained by multiplying the strong form by a test function and applying integration by parts. In the weak formulation, $\delta\overline{\boldsymbol{x}} \in H^1(\overline{\mathcal{B}}_0)$ describes an arbitrary test function (virtual displacement) in the Sobolev space for the macroscopic domain [13, chapter II, §1]. The solution of Equation (2.1) is expressed in terms of the macroscopic displacement field $\overline{\boldsymbol{u}}$.

To obtain a well-posed macroscopic problem, Dirichlet boundary conditions are applied as constraint to the deformation variable. For given boundary values $\overline{\boldsymbol{u}}_0$, the conditions are expressed by

$$\overline{\boldsymbol{u}} = \overline{\boldsymbol{u}}_0 \quad \text{on } \partial\overline{\mathcal{B}}_0^*$$

with $\partial\overline{\mathcal{B}}_0^* \subset \partial\overline{\mathcal{B}}_0$ defining the subset of the boundary where Dirichlet boundary conditions are prescribed. To differentiate between macroscopic and microscopic entities, we denote macroscopic variables and domains with a bar above the respective symbol.

As the macroscopic domain is discretized using finite elements, the displacement field $\overline{\boldsymbol{u}}$ which defines the weak solution of the macroscopic boundary value problem is piecewise given by the ansatz functions in each element of the mesh. We denote the discretization of the macroscopic geometry with $\overline{\tau}$. Within a given finite element $\overline{\mathcal{T}} \in \overline{\tau}$, the discretized deformation vector is given by nodal interpolation

$$\overline{\boldsymbol{u}}_h = \overline{\boldsymbol{N}}_{\overline{\mathcal{T}}} \cdot \overline{\boldsymbol{U}}_{\overline{\mathcal{T}}}$$

for a matrix $\overline{\boldsymbol{N}}_{\overline{\mathcal{T}}}$ containing the ansatz functions and $\overline{\boldsymbol{U}}_{\overline{\mathcal{T}}}$ representing the nodal displacements. The subscript h indicates the discretized representation of the respective variable. This expression of the deformation vector leads to the discretized

representation of the deformed macroscopic state

$$\overline{\boldsymbol{x}}_h = \overline{\boldsymbol{X}}_h + \overline{\boldsymbol{u}}_h = \overline{\boldsymbol{X}}_h + \overline{\boldsymbol{N}}_{\overline{\mathcal{T}}} \cdot \overline{\boldsymbol{U}}_{\overline{\mathcal{T}}}$$

with $\overline{\boldsymbol{X}} \in \overline{\mathcal{B}}_0$ given as the reference state corresponding to the deformed position vector $\overline{\boldsymbol{x}} \in \overline{\mathcal{B}}$. Additionally, the arbitrary test function can be written in terms of the ansatz functions as

$$\delta\overline{\boldsymbol{x}}_h = \overline{\boldsymbol{N}}_{\overline{\mathcal{T}}} \cdot \delta\overline{\boldsymbol{U}}_{\overline{\mathcal{T}}}$$

for virtual nodal displacements $\delta\overline{\boldsymbol{U}}_{\overline{\mathcal{T}}}$ for the finite element $\overline{\mathcal{T}} \in \overline{\tau}$. With the derivatives of the ansatz functions given in the matrix $\overline{\boldsymbol{B}}_{\overline{\mathcal{T}}}$, we can express the discretized deformation gradient

$$\overline{F}_h = \left(\frac{\mathrm{d}\,\overline{\boldsymbol{x}}}{\mathrm{d}\,\overline{\boldsymbol{X}}} \right)_h = I + \overline{\boldsymbol{B}}_{\overline{\mathcal{T}}} \cdot \overline{\boldsymbol{U}}_{\overline{\mathcal{T}}}$$

in terms of the nodal displacements $\overline{\boldsymbol{U}}_{\overline{\mathcal{T}}}$. Here, I refers to the identity matrix with appropriate dimensions.

Based on the macroscopic governing equation, a global residual vector $\overline{R}$ is introduced. With the definition of this term, solving the macroscopic problem translates to finding the root of the residual $\overline{R}$. Following standard finite element assembly, the global residual is composed of the local residual terms

$$\overline{r}_{\overline{\mathcal{T}}}(\overline{\boldsymbol{u}}_h) = \int_{\overline{\mathcal{T}}} \overline{\boldsymbol{B}}_{\overline{\mathcal{T}}}^{\mathsf{T}} \ \overline{P}_h \ (I + \overline{\boldsymbol{u}}_h) - \overline{\boldsymbol{N}}_{\overline{\mathcal{T}}}^{\mathsf{T}} \ \overline{f}_0 \ \mathrm{d}\overline{V} \tag{2.2}$$

in each finite element $\overline{\mathcal{T}} \in \overline{\tau}$ based on the weak formulation in Equation (2.1). In general, computing a solution that minimizes the macroscopic residual term requires the application of a non-linear solver. As non-linear solvers like the Newton-Raphson method [114, chapter 11] use the first derivative of the residual in each iteration, the Jacobian of the global residual $\overline{DR}$ is assembled from the local tangent contributions given by

$$\overline{dr}_{\overline{\mathcal{T}}}(\overline{\boldsymbol{u}}_h) = \int_{\overline{\mathcal{T}}} \overline{\boldsymbol{B}}_{\overline{\mathcal{T}}}^{\mathsf{T}} \ \left(\frac{\partial\,\overline{P}}{\partial\,\overline{F}} \right)_h (I + \overline{\boldsymbol{u}}_h) \ \overline{\boldsymbol{B}}_{\overline{\mathcal{T}}} \ \mathrm{d}\overline{V}. \tag{2.3}$$

Here, $\left(\frac{\partial \overline{P}}{\partial \overline{F}}\right)_h$ refers to the discretized representation of the derivative tensor of the first Piola-Kirchhoff stress tensor with respect to macroscopic deformation gradient. To evaluate the local residuals and tangents presented in Equations (2.2) and (2.3), the integrals are numerically computed using Gaussian quadrature [31, chapter 9].

In the classical finite element method, the local stress-strain response is usually given by a constitutive material model. This constitutive law describes the computation of the first Piola-Kirchhoff stress $\overline{P}$ and the modulus $\frac{\partial \overline{P}}{\partial \overline{F}}$. However, as described above, for heterogeneous materials with complex microstructures, it is often not possible to derive such a material model. Therefore, in the FE^2 method, the local stress response resulting from a given deformation in the macroscopic boundary value problem is derived from the results of a microscopic problem.

For defining the microscopic boundary value problem, the microstructure of the heterogeneous material is modeled within a statistically representative region. We denote the scale of the macroscopic geometry with l_{macro} and the scale of the microstructure with l_{micro}. For the FE^2 method, a significant separation of the two scales is considered with

$$l_{\text{macro}} \gg l_{\text{micro}}.$$

In the context of a heterogeneous material, Figure 2.2 presents the microscopic structure as a statistically significant region of the macroscopic geometry and highlights the difference in the two scales. The visualization presents the macroscopic configuration of the heterogeneous material on the right and the representative microstructure on the left. The colors within the structure on the microscopic scale are supposed to emphasize the heterogeneous nature of the material with areas of different color having different mechanical behavior. As the properties of the considered microstructure are expected to be statistically representative for the material, the structure is also referred to as representative volume element (RVE). To define the microscopic boundary value problem, the structure of the RVE is modeled and discretized using finite elements.

As no constitutive material model is defined for computing the resulting stress values for the macroscopic boundary value problem, at each Gaussian quadrature

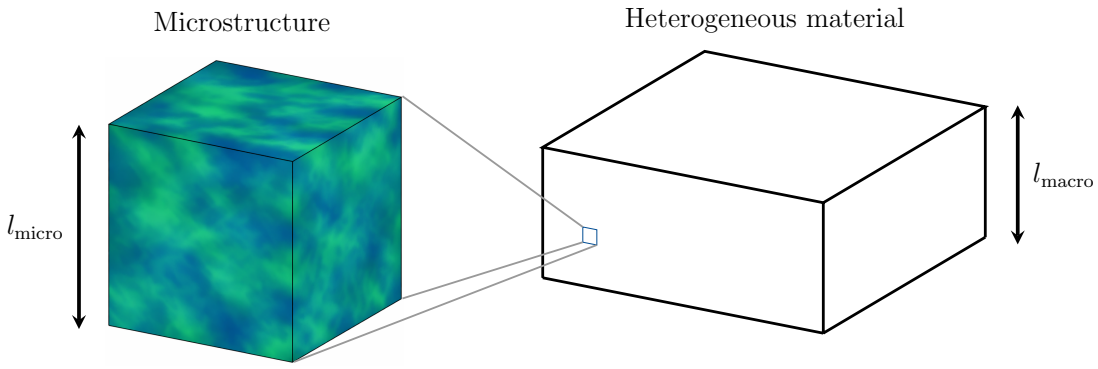


Figure 2.2: Relationship between heterogeneous material and its microstructure. The cuboid on the right shows the macroscopic geometry of a heterogeneous material. A small section of the material is used to determine the microstructural properties of the material. A representation of the material's microstructure is given by the cube on the left.

point of the macroscopic finite element mesh, we consider an individual microscopic boundary value problem. The relationship between the macroscopic and the microscopic problems is described by the Hill-Mandel condition. This relation ensures that the energy on the microscopic scale is consistent with the energy on the macroscopic scale and the condition can be expressed as

$$\overline{P} : \dot{\overline{F}} = \frac{1}{|\mathcal{B}_0|} \int_{\mathcal{B}_0} P : \dot{F} \, \mathrm{d}V. \tag{2.4}$$

Here, $\mathcal{B}_0$ refers to the reference configuration of the microscopic domain, which is defined by the RVE. The variables P and F refer to the microscopic first Piola-Kirchhoff stress tensor and deformation gradient. The operator : denotes the contraction of the both tensors. In Equation (2.4), $\dot{\overline{F}}$ and $\dot{F}$ refer to the time derivatives of the respective deformation gradients.

For a given point $\boldsymbol{X} \in \mathcal{B}_0$ in the reference configuration of the RVE, the deformed position vector $\boldsymbol{x} \in \mathcal{B}$ is derived from the microscopic deformation map $\varphi : \mathcal{B}_0 \to \mathcal{B}$ with $\boldsymbol{X} \mapsto \varphi(\boldsymbol{X}) = \boldsymbol{x}$. On the microscopic scale, we can express the deformed state

$$\boldsymbol{x} = \varphi(\boldsymbol{X}) = \overline{F} \cdot \boldsymbol{X} + \tilde{\boldsymbol{u}} \tag{2.5}$$

in terms of the macroscopic deformation gradient $\overline{F}$ and the fluctuation field $\tilde{\boldsymbol{u}}$. The deformed RVE configuration is denoted by $\mathcal{B}$ with $\boldsymbol{x} \in \mathcal{B}$. For a microscopic point $\boldsymbol{X} \in \mathcal{B}_0$ within the reference configuration, the deformation gradient $F = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{X}}$ is given by the relation

$$F = \overline{F} + \tilde{F}$$

in terms of the macroscopic deformation gradient $\overline{F}$ and the fluctuation term $\tilde{F}$ induced by the microscopic boundary value problem.

Following the relation in Equation (2.5) that describes the microscopic deformation state, the displacement of the RVE is defined by the local deformation gradient $\overline{F}$ given from the macroscopic boundary value problem at the respective macroscopic evaluation point and the fluctuation field $\tilde{\boldsymbol{u}}$. The computations of the fluctuation field are based on the microscopic boundary value problem. For the microscopic problem, we impose periodic boundary conditions on opposite faces of the RVE. The reference configuration of the RVE is assumed to be a square in the two-dimensional case and a cube in the three-dimensional case. Therefore, it is possible to split the microscopic boundary $\partial\mathcal{B}_0$ into two parts, $\partial\mathcal{B}_0 = \partial\mathcal{B}_0^+ \cup \partial\mathcal{B}_0^-$, such that for each node $\boldsymbol{X}^+ \in \partial\mathcal{B}_0^+$ there exists a corresponding node $\boldsymbol{X}^- \in \partial\mathcal{B}_0^-$ with an opposing outer normal vector. For any such pair of periodic nodes $\boldsymbol{X}^+ \in \partial\mathcal{B}_0^+$ and $\boldsymbol{X}^- \in \partial\mathcal{B}_0^-$, we enforce the periodic boundary condition

$$\tilde{\boldsymbol{u}}^+ = \tilde{\boldsymbol{u}}^-$$

where $\tilde{\boldsymbol{u}}^+$ and $\tilde{\boldsymbol{u}}^-$ denote the fluctuation field variables, respectively. Here, $\tilde{\boldsymbol{u}}^+$ refers to the fluctuation field at the node $\boldsymbol{X}^+$ and $\tilde{\boldsymbol{u}}^-$ refers to the fluctuation field at the node $\boldsymbol{X}^-$. Purely periodic boundary conditions still leave rigid-body translation modes. To obtain a regular system, we therefore apply Dirichlet boundary conditions to the corners of the RVE. For the Dirichlet boundary conditions, we denote $\partial\mathcal{B}_0^*$ as the set of nodes located in the corners of the RVE. For the corner nodes $\boldsymbol{X}^* \in \partial\mathcal{B}_0^*$, we enforce the Dirichlet boundary conditions

$$\tilde{\boldsymbol{u}} = \tilde{\boldsymbol{u}}^* = 0 \quad \text{on } \partial\mathcal{B}_0^*$$

for the fluctuation field. This application of periodic and Dirichlet boundary conditions is common practice with regard to the FE^2 method in order to reproduce the conditions of the microstructure inside the material as closely as possible [134].

Based on the microscopic material model, it is possible to compute the microscopic first Piola-Kirchhoff stress tensor $P(F)$ in terms of the respective deformation gradient. To compute the macroscopic first Piola-Kirchhoff stress tensor $\overline{P}$, as it is part of Equation (2.2), we homogenize the microscopic Piola-Kirchhoff stress tensor over the volume of the RVE. This means that the homogenized macroscopic stress tensor is given by the relation

$$\overline{P} = \frac{1}{|\mathcal{B}_0|} \int_{\mathcal{B}_0} P(F)\, \mathrm{d}V \tag{2.6}$$

with the volume of the RVE defined by $|\mathcal{B}_0|$.

Similar to the homogenization of the Piola-Kirchhoff stress tensor, the derivative with respect to the macroscopic deformation gradient is also derived from a homogenization approach. Here, the modulus can be expressed as a derivative in terms of the microscopic stress tensor. Following the chain rule for differentiation, this yields the expression

$$\begin{aligned}
\frac{\partial \overline{P}}{\partial \overline{F}} &= \frac{\partial}{\partial \overline{F}} \left(\frac{1}{|\mathcal{B}_0|} \int_{\mathcal{B}_0} P(F)\, \mathrm{d}V \right) \\
&= \frac{1}{|\mathcal{B}_0|} \int_{\mathcal{B}_0} \frac{\partial P}{\partial F} : \frac{\partial (\overline{F} + \tilde{F})}{\partial \overline{F}}\, \mathrm{d}V \\
&= \frac{1}{|\mathcal{B}_0|} \int_{\mathcal{B}_0} \frac{\partial P}{\partial F}\, \mathrm{d}V + \frac{1}{|\mathcal{B}_0|} \int_{\mathcal{B}_0} \frac{\partial P}{\partial F} : \frac{\partial \tilde{F}}{\partial \overline{F}}\, \mathrm{d}V
\end{aligned}$$

that describes the computation of the derivative of the macroscopic first Piola-Kirchhoff stress tensor with respect to the macroscopic deformation gradient. With this relation, the macroscopic modulus, which is required for the evaluation of the local tangent in Equation (2.3), can be computed from the results of the microscopic boundary value problem.

For evaluating the macroscopic residual and tangent, the homogenization of

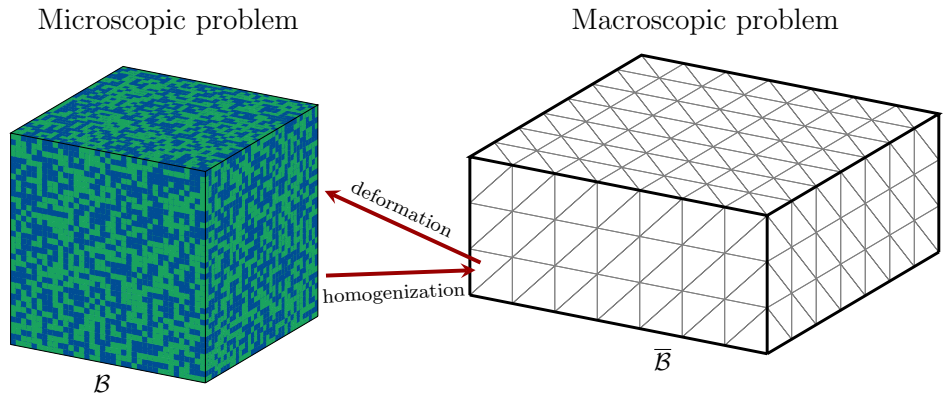


Figure 2.3: Coupling between the macroscopic finite element problem and the microscopic beam frame model. The mesh on the right represents the macroscopic boundary value problem. For each Gaussian quadrature point, the RVE is deformed according to the macroscopic deformation at that point. The RVE is presented as the cube on the left. Based on the deformed RVE, the resulting stress is homogenized over the RVE volume. The homogenized stress defines the mechanical behavior on the macroscopic scale.

the respective microscopic entities is computed for each Gaussian quadrature point of the macroscopic finite element mesh. The coupling of the macroscopic and the microscopic problem is presented in Figure 2.3. In this illustration, the finite element mesh on the right represents the macroscopic problem and the colored cube on the left shows the modeled and discretized RVE which defines the microscopic problem. The red arrows hint that the RVE is deformed based on the local deformation on the macroscopic scale and that the homogenization of the stresses within the RVE yield the macroscopic stress tensor.

A deeper insight regarding the application of the FE^2 method for the simulation of open-porous materials is presented in the following sections of this chapter. Section 2.2 deals with the adjusted definition of the microscopic problem which uses a beam frame to model the RVE. Section 2.3 describes the definition of the macroscopic boundary value problem used for the mechanical simulation.

2.2 Microscopic Problem

As described in the previous section, the FE^2 usually features a finite element discretization on the microscopic scale. In the case of aerogels and similar open-porous materials, we consider a structure of beam elements for modeling the RVE. In the literature, the combination of several beam elements in a structure with fixed or hinged connecting nodes is commonly referred to as frame [115, chapter 3]. In the further course of the work, we refer to this described structure as beam frame to emphasize the physical basis of the model that lies in the Euler–Bernoulli beam theory [144]. In addition to the displacement of the nodes, Euler–Bernoulli beam elements can also be subject to rotation and bending compared to truss elements [115, chapter 2].

Particularly with regard to the simulation of biopolymer aerogels, the modeling based on beam elements is well suited for the representation of the fibrillar nanostructure of the open-porous materials. Depending on the source polymer and the conditions during synthesis, the porous structure of an aerogel can vary significantly. Experimental results have shown that the porous fraction and the pore size distribution can have a direct impact on the mechanical properties of the material [121]. To incorporate the nanostructure of the material, we create an RVE based on its specific porous fraction and pore size distribution, which are measured experimentally from nitrogen desorption isotherms coupled with the Barrett-Joyner-Halenda (BJH) analysis [7].

For the creation of the RVE, we follow the procedure described in [120, 22]. In the first step of the method, a fixed number of random pore sizes is created following the given pore size distribution of the material. The number of pores determines the size and complexity of the resulting RVE. The greater the number of pores, the more accurately the pore sizes of the RVE are expected to follow the given pore size distribution. However, a larger RVE also results in larger computation times for the multiscale simulation. The random pore sizes are then used as a basis for a sphere packing algorithm to arrange the sphere-shaped pores as densely as possible within the volume. To each pore, the algorithm assigns the position of its center point

within the volume. The center points are then used as seeds for a Laguerre-Voronoi tessellation (LVT) [73] with the weights of each Voronoi cell given by the radius of the corresponding pore. The LVT is based on the Voronoi tessellation [147]. As refinement of the Voronoi tessellation, it introduces weight values that affect the metric for computing the distance of a given coordinate to a seed. To describe the algorithm, we consider the d-dimensional case with the number of pores defined by $n_{pores} \in \mathbb{N}$. For each pore with index $i = 1, \dots, n_{pores}$, a center point $\boldsymbol{s}_i \in \mathbb{R}^d$ and a pore radius $r_i > 0$ are given. Based on these entities, the Voronoi cell is defined as a d-dimensional subdomain by the expression

$$V_L(\boldsymbol{s}_i, r_i) = \left\{ \boldsymbol{x} \in \mathbb{R}^d \mid \sqrt{\|\boldsymbol{x} - \boldsymbol{s}_i\|^2 - r_i^2} \leq \sqrt{\|\boldsymbol{x} - \boldsymbol{s}_j\|^2 - r_j^2},\ \forall j = 1, \dots, n_{pores} \right\}.$$

In subdomains where d Voronoi cells overlap at their boundaries, the respective intersections define each a two-dimensional edge. These edges of the given Voronoi cells form the structure of beam elements for the frame model. Due to the nature of the Voronoi tessellation, the beams join at the intersection points of $d + 1$ Voronoi cells and create a fully connected frame structure. For the modeling of the RVE, we consider the two- and three-dimensional case of this creation method.

For modeling the frame structure of the RVE, we consider a circular cross section for each beam element. This assumption follows the insight of SEM images of biopolymer aerogels. The SEM results presented in Figure 2.1 also hint this property of the fibrils. The resulting RVE is then stored as the set of nodes, a list for the adjacencies of all beam elements, and a list containing the diameters of each element. We allow different cross-sectional diameters for the individual beam elements. This assumption follows the observation of SEM images of aerogels where the thickness of the fibrillar nanostructure is not uniform. The diameter of each beam element is computed based on the porous fraction of the material and ratio between volume and perimeter of all Voronoi cells bounded by the respective edge. The exact algorithm used to determine the beam diameters is presented in Figure S4 in the supplementary information of [22]. Despite the assumption of different beam diameters, over the length of a single beam element, we consider a uniform

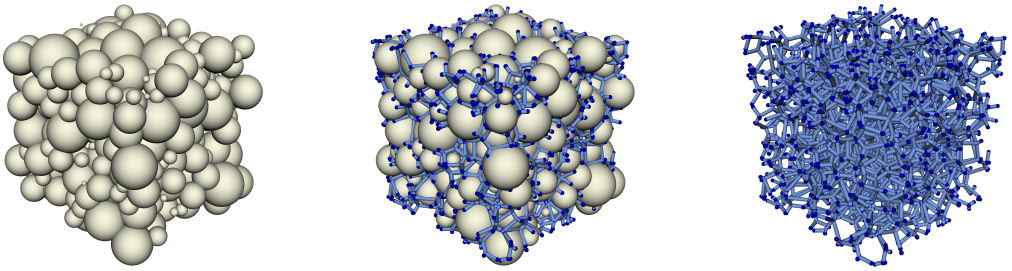


Figure 2.4: Visualization of the method used for creating the beam frame structure for the RVE. The results of the sphere packing algorithm are shown in the left image, the image in the middle presents the packed spheres together with the boundaries of the Voronoi cells, and the image on the right shows the resulting beam frame structure. For visualization, the same diameter is used for all beam elements.

thickness.

The procedure of creating the RVE as a beam frame structure is visualized in Figure 2.4. The figure presents the relation between the sphere packing algorithm and the results of the LVT. The beam frame structure of the RVE resulting from the method is shown in the image on the right. Note that the beam elements in Figure 2.4 are visualized with the same beam diameter. This simplification is supposed to help with the understanding of the approach.

For the FE^2 method, it is a common choice to apply periodic boundary conditions to the microscopic boundary value problem. This is the most natural choice since Dirichlet or Neumann boundary conditions can lead to non-physical behavior [134]. For applying periodic boundary conditions to the RVE, the respective beam structure has to be periodic. This means that for every beam element with a node at a boundary, there should be a corresponding beam element on the opposite boundary. The boundary nodes should share all coordinates except one. However, the frame structures that are created based on sphere packing and LVT generally do not exhibit these periodic properties. To obtain a periodic RVE, we adjust the method after the sphere packing step. The resulting center points of the spheres are copied to create a combination of 2^d sphere packing volumes in which the volumes

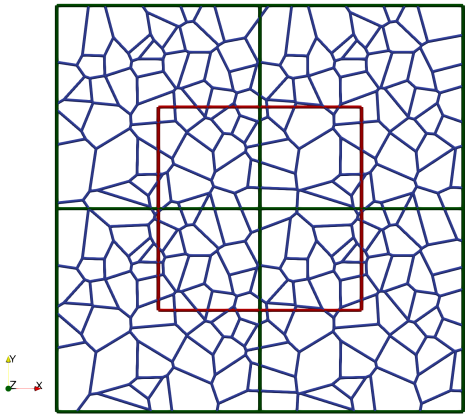


Figure 2.5: Illustration of the method to create a periodic RVE in the two-dimensional case. Prior to the Voronoi tessellation, each of the green squares contains copies of the sphere centers resulting from the sphere packing algorithm. The red square marks the area with periodic boundaries.

are connected to each other at the boundaries. The LVT is then applied to the whole structure, creating a periodic frame in the center with the same dimensions as one of the original volumes. For the two-dimensional case, the method to create a periodic RVE is shown in Figure 2.5. The green squares mark the copied volumes and the red square is the periodic beam frame model which is taken as the resulting RVE. If we focus on the Voronoi cells that are close to the corners of the red square, the periodic properties of the inner frame are apparent, as the arrangement of the Voronoi cells is similar in each of the green squares.

For setting up the microscopic boundary value problem, different approaches can be considered for modeling the beam elements. We consider an RVE consisting of n_B beam elements and n_N nodes in a d-dimensional coordinate system. We define the set of beam elements as $\mathcal{E} = \{e_1, \ldots, e_{n_B}\}$ and the set of nodes as $\mathcal{V} = \{v_1, \ldots, v_{n_N}\}$. For an arbitrary beam element $e \in \mathcal{E}$, the respective radius of the beam is given by $r_e > 0$. The graph built from the nodes and beams is fully connected, meaning that for every two nodes in the RVE, there exists a path over beam elements that connects the two nodes. The modeling of each beam element within a given RVE follows the Euler–Bernoulli beam theory [144], which describes the deflection behavior of

a beam element for an applied load. The theory assumes beam elements with a uniform cross section, which also remains unchanged under bending of the beam. For the full beam frame structure, the deformation of the nodes and the bending of the beams are computed by solving a system of equations of the following form

$$\boldsymbol{K}(\boldsymbol{U}) \cdot \boldsymbol{U} = \boldsymbol{F} \tag{2.7}$$

with the variable of interest $\boldsymbol{U}$ containing, for each node, both the deformation and the rotation of the beam element. The right-hand side of the equation $\boldsymbol{F}$ is given by the force and the bending moments for each vertex. The stiffness matrix $\boldsymbol{K}$ is assembled from element stiffness matrices $\boldsymbol{K}_e(\boldsymbol{U})$ for each beam element e. The element stiffness matrix can be written by the following formula

$$\boldsymbol{K}_e(\boldsymbol{U}) = \boldsymbol{T}_e(\boldsymbol{U})^{\mathsf{T}} \cdot \left(\boldsymbol{K}_e^{material}(\boldsymbol{U}) + \boldsymbol{K}_e^{geometric}(\boldsymbol{U})\right) \cdot \boldsymbol{T}_e(\boldsymbol{U}) \tag{2.8}$$

with a transformation matrix $\boldsymbol{T}_e(\boldsymbol{U})$ and two local stiffness matrices $\boldsymbol{K}_e^{material}(\boldsymbol{U})$ and $\boldsymbol{K}_e^{geometric}(\boldsymbol{U})$. The material element stiffness matrix $\boldsymbol{K}_e^{material}(\boldsymbol{U})$ considers the material properties of the beam element to account for the linear elastic stiffness, while the geometric element stiffness matrix $\boldsymbol{K}_e^{geometric}(\boldsymbol{U})$ accounts for the non-linear geometric effects. For describing the formulas for the stiffness matrices and the transformation matrix we differentiate between the two-dimensional and the three-dimensional case. In both spaces we consider that Young's modulus E and Poisson's ratio ν are given as material properties. The element stiffness matrices follow the descriptions by McGuire et al. in [106], Öchsner [115, chapter 3] and Krenk in [93].

2.2.1 Two-Dimensional Element Stiffness Matrix

First we consider the two-dimensional case with a beam element $e = (\boldsymbol{v}_i, \boldsymbol{v}_j)$. The starting and end point in the two-dimensional coordinate system are given by

$$\boldsymbol{v}_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix} \text{ and } \boldsymbol{v}_j = \begin{pmatrix} x_j \\ y_j \end{pmatrix}.$$

The cross section of the beam is considered circular with a given radius $r_e > 0$. Furthermore, we denote the length of the beam by $L = \|\boldsymbol{v}_j - \boldsymbol{v}_i\|_2$. In the case of a circular cross section, the neutral axis of a beam element follows the center of its cross section. The naming refers to the property of this axis that, when the beam is deformed, the bending stress is zero at the neutral axis. Here, the neutral axis can be described by the set of coordinates

$$\left\{ \boldsymbol{X} \in R^2 \mid \boldsymbol{X} = (1-\gamma) \cdot \boldsymbol{v}_i + \gamma \cdot \boldsymbol{v}_j \quad \text{for } \gamma \in [0,\, 1] \right\}.$$

With the circular cross section, the local second moments of area are equally given by $I_y = I_z = \frac{\pi \cdot r_e^4}{4}$ and the cross-sectional area is obtained from $A = \pi \cdot r_e^2$. Additionally, we have the shear modulus $G = \frac{E}{2 \cdot (1+\nu)}$ and the torsional moment of inertia $J = 2 \cdot I_y$. In a planar Euler–Bernoulli frame model, each node i has two translational degrees of freedom $\boldsymbol{u}_i = \begin{pmatrix} u_i \\ v_i \end{pmatrix} \in \mathbb{R}^2$ and one rotational degree of freedom $\theta_i \in \mathbb{R}$. Consequently, this means that the variable of interest

$$\boldsymbol{U} = \begin{pmatrix} \boldsymbol{u}_1 \\ \theta_1 \\ \boldsymbol{u}_2 \\ \theta_2 \\ \vdots \\ \boldsymbol{u}_{n_N} \\ \theta_{n_N} \end{pmatrix} \in \mathbb{R}^{3 \cdot n_N}$$

has three degrees of freedom per node and each element stiffness matrix is a square matrix of size 6×6. In the two-dimensional case of the Euler–Bernoulli frame model, the material stiffness matrix $\boldsymbol{K}_e^{material}(\boldsymbol{U}) \in \mathbb{R}^{6\times 6}$ is defined as

$$\boldsymbol{K}_e^{material}(\boldsymbol{U}) = \begin{pmatrix} M_1 & M_2 \\ M_2^{\mathsf{T}} & M_3 \end{pmatrix}$$

with the matrices M_1, M_2, and M_3 given by

$$M_1 = \begin{pmatrix} \frac{EA}{L} & 0 & 0 \\ 0 & \frac{12EI_z}{L^3} & \frac{6EI_y}{L^2} \\ 0 & \frac{6EI_y}{L^2} & \frac{4EI_y}{L} \end{pmatrix}, \; M_2 = \begin{pmatrix} \frac{-EA}{L} & 0 & 0 \\ 0 & \frac{12EI_z}{L^3} & \frac{6EI_y}{L^2} \\ 0 & \frac{-6EI_y}{L^2} & \frac{2EI_y}{L} \end{pmatrix}, \; M_3 = \begin{pmatrix} \frac{EA}{L} & 0 & 0 \\ 0 & \frac{12EI_z}{L^3} & \frac{-6EI_y}{L^2} \\ 0 & \frac{-6EI_y}{L^2} & \frac{4EI_y}{L} \end{pmatrix}.$$

A more detailed description and a derivation for the stiffness matrices can be found in the publication of Öchsner [115, chapter 3]. Analogous to the material stiffness matrix, the geometric element stiffness matrix $\boldsymbol{K}_e^{geometric}(\boldsymbol{u}) \in \mathbb{R}^{6\times 6}$ accounts for the second-order contribution induced by the current axial force N in the beam element. The local matrix is commonly expressed as

$$\boldsymbol{K}_e^{geometric}(\boldsymbol{U}) = \begin{pmatrix} G_1 & G_2 \\ G_2^{\mathsf{T}} & G_3 \end{pmatrix}$$

with the submatrices G_1 and G_2 given by

$$G_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \frac{6N}{5L} & \frac{N}{10} \\ 0 & \frac{N}{10} & \frac{2NL}{15} \end{pmatrix}, \; G_2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & -\frac{6N}{5L} & \frac{N}{10} \\ 0 & -\frac{N}{10} & -\frac{2NL}{30} \end{pmatrix}.$$

Here, the axial force N is defined as $N = \frac{E \cdot A}{L} \cdot (u_j^e - u_i^e)$, with

$$\boldsymbol{u}_i^e = \begin{pmatrix} u_i^e \\ v_i^e \end{pmatrix} \quad \text{and } \boldsymbol{u}_j^e = \begin{pmatrix} u_j^e \\ v_j^e \end{pmatrix}$$

referring to the deformation vectors of the beam's start and end node relative to

the local coordinate system of the beam element e. Note that a positive axial force $N > 0$ denotes tension of the beam element and a negative axial force denotes $N < 0$ compression. The local representations of the deformation vectors $\boldsymbol{u}_i^e$, $\boldsymbol{u}_j^e$ are derived from a linear transformation that is described in the following.

The local coordinate system has its origin at the start node of the respective beam element and is aligned with the x-axis parallel to the beam direction. With the transition matrix $T_{sub} \in \mathbb{R}^{2\times 2}$ given by

$$T_{sub}(\boldsymbol{U}) = \begin{pmatrix} \frac{(x_j+u_j)-(x_i+u_i)}{L} & \frac{(y_j+v_j)-(y_i-v_i)}{L} \\ \frac{(y_i-v_i)-(y_j-v_j)}{L} & \frac{(x_j+u_j)-(x_i+u_i)}{L} \end{pmatrix},$$

the global to local transformation of the deformation vectors can be computed by $\boldsymbol{u}_i^e = T_{sub}^{\mathsf{T}} \cdot \boldsymbol{u}_i$ and $\boldsymbol{u}_j^e = T_{sub}^{\mathsf{T}} \cdot \boldsymbol{u}_j$. Note that the matrix T_{sub} is orthonormal and it changes if the beam deforms. With the transition matrix, we can also describe the block diagonal matrix $\boldsymbol{T}_e(\boldsymbol{U}) \in \mathbb{R}^{6\times 6}$ in the form

$$\boldsymbol{T}_e(\boldsymbol{U}) = \begin{pmatrix} 1 & & & & & \\ & T_{sub}(\boldsymbol{U}) & & & & \\ & & & 1 & & \\ & & & & T_{sub}(\boldsymbol{U}) & \end{pmatrix}.$$

Here, the matrix entries outside the diagonal blocks are all zero.

2.2.2 Three-Dimensional Element Stiffness Matrix

To describe the element stiffness matrix in the three-dimensional case, a single beam element $e = (\boldsymbol{v}_i, \boldsymbol{v}_j)$ in a three-dimensional coordinate system is considered with starting point

$$\boldsymbol{v}_i = \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} \text{ and end point } \boldsymbol{v}_j = \begin{pmatrix} x_j \\ y_j \\ z_j \end{pmatrix}.$$

The circular cross section is described by its radius $r_e > 0$. The length of the beam is given by $L = \|\boldsymbol{v}_j - \boldsymbol{v}_i\|_2$. Similar to the two-dimensional case, the neutral axis of the beam element is described by the set of coordinates

$$\left\{ \boldsymbol{X} \in R^3 \mid \boldsymbol{X} = (1-\gamma) \cdot \boldsymbol{v}_i + \gamma \cdot \boldsymbol{v}_j \quad \text{for } \gamma \in [0,\, 1] \right\}.$$

As a circular cross section is assumed for each beam element, the local second moments of area are equally given by $I_y = I_z = \frac{\pi \cdot r_e^4}{4}$ and the cross-sectional area is obtained from $A = \pi \cdot r_e^2$. Additionally, we have the shear modulus $G = \frac{E}{2 \cdot (1+\nu)}$ and the torsional moment of inertia $J = 2 \cdot I_y$.

For each node of the three-dimensional frame structure, we have three degrees of freedom regarding the displacement $\boldsymbol{u} \in \mathbb{R}^3$ and three degrees of freedom for the rotation $\boldsymbol{\theta} \in \mathbb{R}^3$ of the node. Consequentially, with the variable of interest

$$\boldsymbol{U} = \begin{pmatrix} \boldsymbol{u}_1 \\ \boldsymbol{\theta}_1 \\ \boldsymbol{u}_2 \\ \boldsymbol{\theta}_2 \\ \vdots \\ \boldsymbol{u}_{n_N} \\ \boldsymbol{\theta}_{n_N} \end{pmatrix} \in \mathbb{R}^{6 \cdot n_N},$$

each element stiffness matrix is a square matrix with 12 rows. As this stiffness matrix features material and geometric components, we describe the composition of both matrices. Here, the material element stiffness matrix $\boldsymbol{K}_e^{material}(\boldsymbol{U}) \in \mathbb{R}^{12 \times 12}$ is defined as

$$\boldsymbol{K}_e^{material}(\boldsymbol{U}) = \begin{pmatrix} M_1 & M_2 & -M_1 & -M_2 \\ M_2^\mathsf{T} & M_3 & -M_2^\mathsf{T} & M_4 \\ -M_1 & -M_2 & M_1 & -M_2 \\ -M_2^\mathsf{T} & M_4 & -M_2^\mathsf{T} & M_3 \end{pmatrix}$$

with the matrices M_1, M_2, M_3, and M_4 given by

$$M_1 = \begin{pmatrix} \frac{EA}{L} & 0 & 0 \\ 0 & \frac{12EI_z}{L^3} & 0 \\ 0 & 0 & \frac{12EI_y}{L^3} \end{pmatrix}, \qquad M_2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & \frac{6EI_z}{L^2} \\ 0 & -\frac{6EI_y}{L^2} & 0 \end{pmatrix},$$

$$M_3 = \begin{pmatrix} \frac{GJ}{L} & 0 & 0 \\ 0 & \frac{4EI_y}{L} & 0 \\ 0 & 0 & \frac{4EI_z}{L} \end{pmatrix}, \qquad M_4 = \begin{pmatrix} -\frac{GJ}{L} & 0 & 0 \\ 0 & \frac{2EI_y}{L} & 0 \\ 0 & 0 & \frac{2EI_z}{L} \end{pmatrix}.$$

See [115, chapter 3] for more insight regarding the derivation of the stiffness matrices. The geometric element stiffness matrix $\boldsymbol{K}_e^{geometric}(\boldsymbol{U}) \in \mathbb{R}^{12\times 12}$ can be written as

$$\boldsymbol{K}_e^{geometric}(\boldsymbol{U}) = \begin{pmatrix} G_1 & G_2 & -G_1 & G_2^\mathsf{T} \\ G_2^\mathsf{T} & 4\cdot G_3 & G_2 & -G_3 \\ -G_1 & G_2^\mathsf{T} & G_1 & G_2^\mathsf{T} \\ G_2 & -G_3 & G_2 & 4\cdot G_3 \end{pmatrix} \in \mathbb{R}^{12\times 12}$$

with the submatrices G_1, G_2, and G_3 given by

$$G_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \frac{5N}{6L} & 0 \\ 0 & 0 & \frac{5N}{6L} \end{pmatrix}, G_2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & \frac{N}{10} \\ 0 & -\frac{N}{10} & 0 \end{pmatrix}, G_3 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \frac{NL}{30} & 0 \\ 0 & 0 & \frac{NL}{30} \end{pmatrix}.$$

Similar to the two-dimensional case, the axial force N is defined as $N = \frac{E\cdot A}{L} \cdot (u_j^e - u_i^e)$. The local deformation vectors

$$\boldsymbol{u}_i^e = \begin{pmatrix} u_i^e \\ v_i^e \\ w_i^e \end{pmatrix} \quad \text{and } \boldsymbol{u}_j^e = \begin{pmatrix} u_j^e \\ v_j^e \\ w_j^e \end{pmatrix}$$

corresponding to the local coordinate system of the beam element are derived from a linear transformation described by the matrix $T_{sub}(\boldsymbol{U}) \in \mathbb{R}^{3\times 3}$.

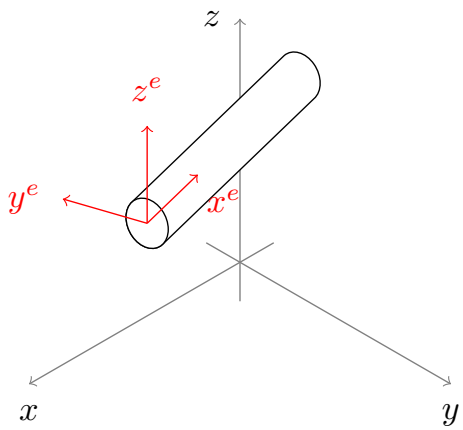


Figure 2.6: Relationship between global and local coordinates for a single beam element in three dimensions.

The transformation matrix $\boldsymbol{T}_e(\boldsymbol{U}) \in \mathbb{R}^{12\times 12}$ is a block diagonal matrix of the form

$$\boldsymbol{T}_e(\boldsymbol{U}) = \begin{pmatrix} T_{sub}(\boldsymbol{U}) & \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} \\ \mathbf{0}_{3\times 3} & T_{sub}(\boldsymbol{U}) & \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} \\ \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} & T_{sub}(\boldsymbol{U}) & \mathbf{0}_{3\times 3} \\ \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} & T_{sub}(\boldsymbol{U}) \end{pmatrix}$$

with the zero matrix $\mathbf{0}_{3\times 3} \in \mathbb{R}^{3\times 3}$ and the transition matrix $T_{sub}(\boldsymbol{U}) \in \mathbb{R}^{3\times 3}$. The matrix $T_{sub}(\boldsymbol{U})$ changes the basis from the local coordinate system, which is aligned with the beam element e, to the global coordinate system. The transition matrix is determined by the start and end nodes of the beam element but can also change depending on the deformation. The relationship between global and local coordinates is shown in Figure 2.6 for an arbitrary beam element in three-dimensions.

2.2.3 Non-Linearity and Alternative Formulations

The beam frame formulation in Equation (2.7) with the stiffness matrix given by Equation (2.8) is in general non-linear, since each element stiffness matrix depends on the current displacement of the corresponding beam element. However, it is

also a common approach to use a linear formulation of Equation (2.7) in which the element stiffness matrices do not consider the deformation but use only the undeformed reference configuration of the beam frame model for assembling $\boldsymbol{K}$. With the terms in Sections 2.2.1 and 2.2.2, this leads to the simplified form of the element stiffness matrix

$$\boldsymbol{K}_e(\boldsymbol{0}) = \boldsymbol{T}_e(\boldsymbol{0})^{\mathsf{T}} \cdot \left(\boldsymbol{K}_e^{material}(\boldsymbol{0}) + \boldsymbol{K}_e^{geometric}(\boldsymbol{0})\right) \cdot \boldsymbol{T}_e(\boldsymbol{0}) = \boldsymbol{T}_e(\boldsymbol{0})^{\mathsf{T}} \cdot \boldsymbol{K}_e^{material}(\boldsymbol{0}) \cdot \boldsymbol{T}_e(\boldsymbol{0}).$$

We denote the vector $\boldsymbol{0}$ as the vector of the same size as $\boldsymbol{U}$ that contains only entries equal to zero. With the described relation, the full stiffness matrix $\boldsymbol{K}$ is not dependent on the deformation or rotation of the nodes. This leads to a system of linear equations

$$\boldsymbol{K}(\boldsymbol{0}) \cdot \boldsymbol{U} = \boldsymbol{F} \tag{2.9}$$

that describes the microscopic beam frame model. In the following sections of this work, we refer to the problem described by Equation (2.9) as the linear beam frame model.

The inclusion of non-linear geometric effects with the matrix $\boldsymbol{K}_e^{geometric}(\boldsymbol{U})$ to Equation (2.9) leads to the element stiffness matrix

$$\boldsymbol{K}_e(\boldsymbol{U}) = \boldsymbol{T}_e(\boldsymbol{0})^{\mathsf{T}} \cdot \left(\boldsymbol{K}_e^{material}(\boldsymbol{0}) + \boldsymbol{K}_e^{geometric}(\boldsymbol{U})\right) \cdot \boldsymbol{T}_e(\boldsymbol{0}). \tag{2.10}$$

We refer to the beam frame problem which derives from assembling Equation (2.7) from the non-linear element stiffness matrices given in Equation (2.10) as the geometrically non-linear beam frame model. This is distinguished from the fully non-linear beam frame problem derived from the element stiffness matrices given by Equation (2.8).

In addition to the beam frame models described in this chapter, it is common to model frame structures using truss elements. This leads to a simpler problem compared to the beam frame, as truss models only consider the deformation of the joints, neglecting the rotation. This results in a linear microscopic problem with the significantly smaller stiffness matrix $\boldsymbol{K} \in \mathbb{R}^{2 \cdot n_N \times 2 \cdot n_N}$ in the two-dimensional case

and $\boldsymbol{K} \in \mathbb{R}^{3 \cdot n_N \times 3 \cdot n_N}$ in the three-dimensional case [115, chapter 2]. However, the truss model requires stronger connectivity within the frame structure in order for a given problem to be well-defined, whereas the described beam approach allows well-defined problems to be constructed even with lower connectivity. For the rather sparsely connected aerogel nanostructures we are interested in, we observed that for most RVEs the connectivity has not been high enough for applying the truss model. Due to this limitation in terms of usability for the simulation of open-porous materials, we focus our work on the beam frame approaches described in the previous passage.

2.2.4 Stress Homogenization

In the multiscale framework, macroscopic quantities are obtained by homogenizing the stresses generated by the deformation of the microscopic RVE. After solving the beam frame problem based on the system of equations described in Equation (2.7), the nodal solution vector $\boldsymbol{U}$ that contains the displacements and rotations of all RVE nodes is used to evaluate the stress components in each beam element. The homogenized stress tensor is then obtained by integrating these stresses over the RVE volume, resulting in a single representative stress tensor for the entire RVE.

Regarding applications in solid mechanics, the Cauchy stress tensor commonly describes the stress state within a material in a deformed state [105, chapter 3]. In the two-dimensional case, it is described by a symmetric matrix, given by $\sigma \in \mathbb{R}^{2\times2}$, and in three dimensions, by the symmetric matrix $\sigma \in \mathbb{R}^{3\times3}$. In addition to the Cauchy stress tensor, the first Piola-Kirchhoff stress tensor enjoys great popularity with regard to finite element simulation. In the scope of this work, we consider both stress measurements for the multiscale approach. The differences regarding the application of both stress tensors are discussed in Section 2.3. As both stress measurements are relevant for the further scope of this work, we derive homogenization schemes for the Cauchy stress tensor and the first Piola-Kirchhoff stress tensor.

In general, the homogenization of stresses within the FE^2 framework is computed

by integrating over the RVE volume, which is denoted by $\mathcal{B}_0$. For the homogenized Cauchy stress tensor $\overline{\sigma}$, this leads to the equation

$$\overline{\sigma} = \frac{1}{V} \int_{\mathcal{B}_0} \sigma \, \mathrm{d}V$$

and similarly, the first homogenized Piola-Kirchhoff stress tensor $\overline{P}$ is given by

$$\overline{P} = \frac{1}{V} \int_{\mathcal{B}_0} P \, \mathrm{d}V$$

with $V := |\mathcal{B}_0|$ referring to the volume of the RVE. Regarding the special case of the beam frame structure, the stresses within the RVE volume can only be non-zero on the beam elements. As the considered materials are open-porous and the pores represented in the beam frame structure are not enclosed, it is not possible for stress to arise inside the pores. Therefore, the integration over the full volume of the RVE is reduced to an integration over the beam elements. This means that the integral for computing the homogenized Cauchy stress tensor reduces to

$$\overline{\sigma} = \frac{1}{V} \sum_{e \in \mathcal{E}} \int_{V_e} \sigma \, \mathrm{d}V$$

and in the case of the first Piola-Kirchhoff stress tensor, the formula reduces to

$$\overline{P} = \frac{1}{V} \sum_{e \in \mathcal{E}} \int_{V_e} P \, \mathrm{d}V.$$

Here, $\mathcal{E}$ denotes the set of beam elements within the RVE and the volume of the beam element e is given by V_e. As the cross section of the beam elements is assumed to be circular, the volume V_e describes a cylinder in the three-dimensional space. With the representation of the homogenized stresses as a sum over the beam elements, it is sufficient to describe the integration of the stress tensors over the volume of a single beam element e.

For describing the procedure of computing the stress within a single beam, we consider a beam element $e = (\boldsymbol{v}_i, \boldsymbol{v}_j)$ with the radius $r_e > 0$ and the length

$L = \|\boldsymbol{v}_j - \boldsymbol{v}_i\|_2$. We assume that $\boldsymbol{U}$ is the solution vector to a given beam frame problem and the deformations of the nodes $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ are given by $\boldsymbol{u}_i$ and $\boldsymbol{u}_j$. The corresponding rotation of each node is described by $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_j$.

The approach for computing stresses within one beam element is based on the results of [8] by Becker and Sokolow. The authors focus on truss elements and do not account for bending moments in their computations. We are aware that this is generally not aligned with the Euler–Bernoulli beam theory, which we considered in the previous section. The incorporation of bending moments into the homogenization method is expected to produce more accurate results, especially with regard to large deformations. However, since we are considering relatively small deformations regarding the mechanical behavior of aerogels, we consider this simplification to be sufficient for our computations. For small deformations, the bending moments are less significant and we expect the approach of Becker and Sokolow to be sufficiently accurate.

First, we focus on the Cauchy stress tensor as a measurement and derive the integral over the volume of the beam element $e \in \mathcal{E}$. As described in [8], the integral over the Cauchy stress tensor can be expressed as

$$\int_{V_e} \sigma \mathrm{d}V = \int_{V_e} \frac{L}{2} \cdot (\boldsymbol{Q} \otimes (\boldsymbol{v}_j - \boldsymbol{v}_i) + (\boldsymbol{v}_j - \boldsymbol{v}_i) \otimes \boldsymbol{Q}) \, \mathrm{d}V \tag{2.11}$$

with the force vector $\boldsymbol{Q}$ and the operator $\otimes$ describing the dyadic product. In the three-dimensional case, the force vector can be written as

$$\boldsymbol{Q} = \begin{pmatrix} Q_x \\ Q_y \\ Q_z \end{pmatrix} \in \mathbb{R}^3$$

and the components of the Cauchy stress tensor

$$\sigma = \begin{pmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{xy} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{xz} & \sigma_{yz} & \sigma_{zz} \end{pmatrix} \in \mathbb{R}^{3\times 3}$$

can then be explicitly expressed by the relations

$$\sigma_{xx} = L \cdot Q_x(x_j - x_i), \quad \sigma_{xy} = \frac{L}{2}(Q_x(y_j - y_i) + Q_y(x_j - x_i)),$$

$$\sigma_{yy} = L \cdot Q_y(y_j - y_i), \quad \sigma_{xz} = \frac{L}{2}(Q_x(z_j - z_i) + Q_z(x_j - x_i)),$$

$$\sigma_{zz} = L \cdot Q_z(z_j - z_i), \quad \sigma_{yz} = \frac{L}{2}(Q_y(z_j - z_i) + Q_z(y_j - y_i)).$$

The force components Q_x, Q_y, and Q_z can be expressed with respect to the derivative of the local deformation of the beam element. Let x^e be the local variable of the coordinate axis that is aligned with the beam element e. Then, the starting point of the beam element $\boldsymbol{v}_i = (x_i, y_i, z_i)^\mathsf{T}$ converts in the local coordinate system to $\boldsymbol{v}_i^e = (0, 0, 0)^\mathsf{T}$ and the end node $\boldsymbol{v}_j = (x_j, y_j, z_j)^\mathsf{T}$ converts to $\boldsymbol{v}_j^e = (1, 0, 0)^\mathsf{T}$. This relation between global and local coordinates is also represented in Figure 2.6 for an exemplary beam element in three dimensions. Additionally, we introduce a local representation of the deformation variables for the given beam element. The deformation vector $\hat{\boldsymbol{u}}$ describes the global deformation with respect to the local coordinate variables of the beam element e.

By following the relations for the Euler-Bernoulli beam presented in [115, section 3.2], the shear force components can be expressed in terms of the derivatives with respect to the local coordinates as

$$\begin{aligned} Q_x &= -\frac{E \cdot I_y}{L^3} \cdot \frac{\mathrm{d}^3 \hat{u}}{\mathrm{d}(x^e)^3}, \\ Q_y &= -\frac{E \cdot I_y}{L^3} \cdot \frac{\mathrm{d}^3 \hat{v}}{\mathrm{d}(y^e)^3}, \text{ and} \\ Q_z &= -\frac{E \cdot I_y}{L^3} \cdot \frac{\mathrm{d}^3 \hat{w}}{\mathrm{d}(z^e)^3}. \end{aligned} \tag{2.12}$$

To compute the integrated Cauchy stress tensor as given in Equation (2.11),

we need to derive the third derivatives in Equation (2.12) for computing the force components. For framing an explicit expression for the derivatives of the deformation components, we use the basic assumptions of a cubic relationship between the deformation of the beam element $\boldsymbol{u}$ and the local coordinate variable x^e. This cubic relationship is a fundamental premise for the development of the system of equations that defines the beam frame problem [115, chapter 3], as it is given by Equation (2.7). To describe this relationship mathematically, we denote the deformation vector with respect to the local coordinate variable as $\hat{\boldsymbol{u}}(x^e)$. Let $\boldsymbol{X} \in \mathcal{B}_0$ be a global position vector that lies on the neutral axis of beam element e. For the given coordinate vector, the corresponding local variable x^e can be expressed by $x^e = \frac{\|\boldsymbol{X}-\boldsymbol{v}_i\|_2}{L}$. Here, x^e can be interpreted as the distance of the respective coordinate $\boldsymbol{X}$ from the starting point $\boldsymbol{v}_i$ of the beam relative to its length L. In this case, we obtain a compatible expression for the displacement $\boldsymbol{u}(\boldsymbol{X}) = \hat{\boldsymbol{u}}(x_e)$. Based on the cubic relation, this means that there exists a constant matrix $A \in \mathbb{R}^{3\times 4}$ such that the deformation over the length of the beam can be represented with respect to x^e by the equation

$$\hat{\boldsymbol{u}}(x^e) = A \cdot \begin{pmatrix} (x^e)^3 \\ (x^e)^2 \\ x^e \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} \cdot (x^e)^3 + a_{12} \cdot (x^e)^2 + a_{13} \cdot x^e + a_{14} \\ a_{21} \cdot (x^e)^3 + a_{22} \cdot (x^e)^2 + a_{23} \cdot x^e + a_{24} \\ a_{31} \cdot (x^e)^3 + a_{32} \cdot (x^e)^2 + a_{33} \cdot x^e + a_{34} \end{pmatrix} \in \mathbb{R}^3. \quad (2.13)$$

Note that for a given beam element $e \in \mathcal{E}$, the specific entries of the corresponding matrix A can be derived from displacement and rotation of the respective start and end nodes. This cubic representation of the deformation vector results in the

expressions

$$\frac{d^3\,\hat{u}(x^e)}{d\,(x^e)^3} = 6\,a_{11},$$

$$\frac{d^3\,\hat{v}(x^e)}{d\,(x^e)^3} = 6\,a_{21}, \text{ and} \tag{2.14}$$

$$\frac{d^3\,\hat{w}(x^e)}{d\,(x^e)^3} = 6\,a_{31}$$

for the third derivatives used in Equation (2.12).

With the explicit formulation of the force components, it is possible to integrate the Cauchy stress tensor over the volume of a given beam element. Equation (2.14) shows that the third derivative of the deformation vector with respect to the local coordinate x^e is constant. Based on this representation, the shear force components in Equation (2.12) as well as the components of the Cauchy stress tensor σ are also constant over the volume of the beam element. This results in the simple representation

$$\int_{V_e} \sigma \; \mathrm{d}V = L \cdot \pi \cdot r^2 \cdot \sigma$$

for the Cauchy stress integral over a single beam element.

In the following, we consider the homogenization of the first Piola-Kirchhoff stress tensor. We can do this by regarding the integral over a single beam element e. In general, the first Piola-Kirchhoff stress tensor is given by the relation

$$P = J\,\sigma\,(F^{-1})^{\mathsf{T}}$$

which depends on the Cauchy stress σ, the transposed inverse of the deformation gradient F, and the determinant of the deformation gradient $J := \det(F)$ [105, section 5.5]. By describing each of the given components required for the computation of P in terms of the deformation variable $\boldsymbol{u}$, we are able to derive an explicit computation method for the integration of the first Piola-Kirchhoff stress tensor.

Based on the representation of the deformation vector in terms of the local position variable x^e in Equation (2.13), it is also possible to derive an explicit formulation for the Jacobian of $\boldsymbol{u}$ in terms of local coordinates. Generally, the Jacobian can be written as

$$\nabla \boldsymbol{u}(X) = \begin{pmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} & \frac{\partial u}{\partial z} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} & \frac{\partial v}{\partial z} \\ \frac{\partial w}{\partial x} & \frac{\partial w}{\partial y} & \frac{\partial w}{\partial z} \end{pmatrix}$$

and the partial derivatives can be expressed in terms of the beam length variable x^e by the relations

$$\frac{\partial u}{\partial x} = \frac{d\,\hat{u}(x^e)}{d\,x^e} \cdot \frac{\partial x^e}{\partial x}, \qquad \frac{\partial u}{\partial y} = \frac{d\,\hat{u}(x^e)}{d\,x^e} \cdot \frac{\partial x^e}{\partial y}, \qquad \frac{\partial u}{\partial z} = \frac{d\,\hat{u}(x^e)}{d\,x^e} \cdot \frac{\partial x^e}{\partial z},$$

$$\frac{\partial v}{\partial x} = \frac{d\,\hat{v}(x^e)}{d\,x^e} \cdot \frac{\partial x^e}{\partial x}, \qquad \frac{\partial v}{\partial y} = \frac{d\,\hat{v}(x^e)}{d\,x^e} \cdot \frac{\partial x^e}{\partial y}, \qquad \frac{\partial v}{\partial z} = \frac{d\,\hat{v}(x^e)}{d\,x^e} \cdot \frac{\partial x^e}{\partial z},$$

$$\frac{\partial w}{\partial x} = \frac{d\,\hat{w}(x^e)}{d\,x^e} \cdot \frac{\partial x^e}{\partial x}, \qquad \frac{\partial w}{\partial y} = \frac{d\,\hat{w}(x^e)}{d\,x^e} \cdot \frac{\partial x^e}{\partial y}, \qquad \frac{\partial w}{\partial z} = \frac{d\,\hat{w}(x^e)}{d\,x^e} \cdot \frac{\partial x^e}{\partial z}$$

following the chain rule. Based on the cubic relation, the partial derivatives of the components of $\hat{\mathbf{u}}$ are given by

$$\frac{d\,\hat{u}(x^e)}{d\,x^e} = 3a_{11}(x^e)^2 + 2a_{12}x^e + a_{13},$$

$$\frac{d\,\hat{v}(x^e)}{d\,x^e} = 3a_{21}(x^e)^2 + 2a_{22}x^e + a_{23}, \text{ and}$$

$$\frac{d\,\hat{w}(x^e)}{d\,x^e} = 3a_{31}(x^e)^2 + 2a_{32}x^e + a_{33}$$

and the partial derivatives of x^e can be expressed as

$$\frac{\partial x^e}{\partial x} = \frac{x_j - x_i}{L},$$

$$\frac{\partial x^e}{\partial y} = \frac{y_j - y_i}{L}, \text{ and}$$

$$\frac{\partial x^e}{\partial z} = \frac{z_j - z_i}{L}.$$

Following the expressions of the partial derivatives, the Jacobian $\nabla \boldsymbol{u}$ can be written as

$$\nabla \boldsymbol{u} = \frac{d\,\hat{\boldsymbol{u}}(x^e)}{d\,x^e} \otimes \nabla x^e$$

with the dyadic product of the vectors $\frac{d\,\hat{\boldsymbol{u}}(x^e)}{d\,x^e} \in \mathbb{R}^3$ and $\nabla x^e \in \mathbb{R}^3$ being defined by the scalar derivative components.

The representation of the Jacobian directly yields the explicit formulation of the deformation gradient

$$F = I + \nabla \boldsymbol{u} = I + \frac{d\,\hat{\boldsymbol{u}}(x^e)}{d\,x^e} \otimes \nabla x^e \tag{2.15}$$

and with this term we are able to derive a formula for its determinant $J = \det(F)$. Following the Leibniz formula for the computation of determinants of 3×3 matrices and some simplification steps, the Jacobian determinant is given by

$$\begin{aligned}
J = \det(F) = &\left(1 + \frac{d\,\hat{u}}{d\,x^e}\frac{\partial x^e}{\partial x}\right) \cdot \left(1 + \frac{d\,\hat{v}}{d\,x^e}\frac{\partial x^e}{\partial y}\right) \cdot \left(1 + \frac{d\,\hat{w}}{d\,x^e}\frac{\partial x^e}{\partial z}\right) \\
&+ 2 \cdot \frac{d\,\hat{u}}{d\,x^e}\frac{d\,\hat{v}}{d\,x^e}\frac{d\,\hat{w}}{d\,x^e} \cdot \frac{\partial x^e}{\partial x}\frac{\partial x^e}{\partial y}\frac{\partial x^e}{\partial z} \\
&- \left(1 + \frac{d\,\hat{u}}{d\,x^e}\frac{\partial x^e}{\partial x}\right) \cdot \frac{d\,\hat{v}}{d\,x^e}\frac{d\,\hat{w}}{d\,x^e} \cdot \frac{\partial x^e}{\partial y}\frac{\partial x^e}{\partial z} \\
&- \left(1 + \frac{d\,\hat{v}}{d\,x^e}\frac{\partial x^e}{\partial y}\right) \cdot \frac{d\,\hat{u}}{d\,x^e}\frac{d\,\hat{w}}{d\,x^e} \cdot \frac{\partial x^e}{\partial x}\frac{\partial x^e}{\partial z} \\
&- \left(1 + \frac{d\,\hat{w}}{d\,x^e}\frac{\partial x^e}{\partial z}\right) \cdot \frac{d\,\hat{u}}{d\,x^e}\frac{d\,\hat{v}}{d\,x^e} \cdot \frac{\partial x^e}{\partial x}\frac{\partial x^e}{\partial y} \\
= &1 + \frac{d\,\hat{u}}{d\,x^e}\frac{\partial x^e}{\partial x} + \frac{d\,\hat{v}}{d\,x^e}\frac{\partial x^e}{\partial y} + \frac{d\,\hat{w}}{d\,x^e}\frac{\partial x^e}{\partial z}
\end{aligned}$$

$$= 1 + \operatorname{trace}(\nabla \boldsymbol{u}) = 1 + \left\langle \frac{d\,\hat{\boldsymbol{u}}(x^e)}{d\,x^e}, \nabla x^e \right\rangle .$$

Given the equations for F and J with respect to the local beam coordinate variable x^e, it is also possible to derive an explicit expression for the inverse F^{-1} of the deformation gradient. As shown in Equation (2.15), the deformation gradient can be represented as the sum of the identity matrix and a dyadic product of two vectors. For this special case, the Sherman-Morrison formula yields an explicit formulation of the inverse matrix as described in [58, chapter 2]. Following this relation, the inverse of the deformation gradient can be expressed as

$$F^{-1} = I - \frac{1}{J} \cdot \nabla \boldsymbol{u}.$$

With the derivation of explicit formulations of J and F^{-1}, the integral over the volume of a beam element e regarding the first Piola-Kirchhoff stress tensor can be expressed as

$$\begin{aligned}
\int_{V_e} P \,\mathrm{d}V &= \int_{V_e} J \cdot \sigma \cdot (I - \frac{1}{J} \nabla \boldsymbol{u})^{\mathsf{T}} \,\mathrm{d}V \\
&= \int_{V_e} J \cdot \sigma - \sigma \cdot \nabla \boldsymbol{u}^{\mathsf{T}} \,\mathrm{d}V \\
&= \int_{V_e} J \cdot \sigma \,\mathrm{d}V - \int_{V_e} \sigma \cdot \nabla \boldsymbol{u}^{\mathsf{T}} \,\mathrm{d}V \\
&= \int_{V_e} \left(1 + \left\langle \frac{d\,\hat{\boldsymbol{u}}(x^e)}{d\,x^e}, \nabla x^e \right\rangle \right) \cdot \sigma \,\mathrm{d}V - \int_{V_e} \sigma \cdot \left(\nabla x^e \otimes \frac{d\,\hat{\boldsymbol{u}}(x^e)}{d\,x^e} \right) \mathrm{d}V
\end{aligned}$$

in two separate parts. Using this representation together with the derivation of the integrated Cauchy stress, the integral can be directly computed for each beam element within the RVE and the homogenized stress tensor results from the sum over all beam elements as described above.

2.3 Macroscopic Problem

The macroscopic domain of the given FE^2 problem is denoted by $\overline{\mathcal{B}}_0$ in the undeformed state. For any given point within the macroscopic domain, we note $\overline{\boldsymbol{X}} \in \overline{\mathcal{B}}_0$ as the undeformed configuration and the deformed state is given by $\overline{\boldsymbol{x}} := \overline{\boldsymbol{X}} + \overline{\boldsymbol{u}} \in \overline{\mathcal{B}}$ with $\overline{\boldsymbol{u}}$ being the macroscopic deformation vector. The domain is partitioned with a finite element discretization [13, 72].

For the macroscopic boundary value problem, we consider the momentum balance equation, which is an essential equation in continuum mechanics. Further insight to the equation can be found in the work of Holzapfel [69, chapter 4.3] or in the work of Mase et al. [105, chapter 5]. It follows Newton's second law of motion and it is possible to express the relation in terms of the Cauchy stress

$$\mathrm{Div}_{\overline{\boldsymbol{x}}}\, \overline{\sigma} + \overline{f} = \overline{\rho} \cdot \overline{a} \quad \text{on } \overline{\mathcal{B}}$$

and in terms of the first Piola-Kirchhoff stress tensor

$$\mathrm{Div}_{\overline{\boldsymbol{X}}}\, \overline{P} + \overline{f}_0 = \overline{\rho}_0 \cdot \overline{a} \quad \text{on } \overline{\mathcal{B}}_0$$

with $\overline{a}$ referring to the acceleration. The two formulations differ in that the Cauchy stress formulation uses the current force and density states of the solid $\overline{f}$, $\overline{\rho}$, while the formulation regarding the Piola-Kirchhoff stress uses the reference force and density $\overline{f}_0$, $\overline{\rho}_0$ of the undeformed state. Due to this description in terms of the reference configuration $\overline{\mathcal{B}}_0$, the formulation based on the first Piola-Kirchhoff stress tensor is commonly used in mechanical simulations like in the FEM.

In the static form of the momentum balance equation, the acceleration term is neglected. This eliminates the right side of the equations. Regarding the formulation based on the first Piola-Kirchhoff stress tensor, the weak form of the static equation is given by

$$\int_{\overline{\mathcal{B}}_0} \delta\overline{\boldsymbol{x}} \left(\mathrm{Div}_{\overline{\boldsymbol{X}}}\, \overline{P} + \overline{f}_0 \right) \mathrm{d}\overline{V} = 0$$

for a test function $\delta\overline{\boldsymbol{x}} \in H^1(\overline{\mathcal{B}}_0)$ satisfying the macroscopic boundary conditions.

Here, $H^1(\overline{\mathcal{B}}_0)$ describes the Sobolev space for the macroscopic domain [13, chapter II, §1]. By neglecting the volume force $\overline{f}_0$, the weak form simplifies to

$$\int_{\overline{\mathcal{B}}_0} \delta\overline{\boldsymbol{x}} \, \mathrm{Div}_{\overline{\boldsymbol{X}}} \, \overline{P} \, \mathrm{d}\overline{V} = 0. \tag{2.16}$$

To obtain a regular system for the macroscopic problem that can be solved for the displacement field $\overline{\boldsymbol{u}}$, we apply Dirichlet boundary conditions to given subdomains of the boundary of the macroscopic geometry $\partial\overline{\mathcal{B}}_0$. The application of these boundary conditions is described in Section 2.1.

2.3.1 Coupling of Microscopic and Macroscopic Scale

In the FE^2 method, the macroscopic stress measurements are considered to result from the homogenization of the stresses on the microscopic scale. This coupling of the macroscopic and microscopic deformations follows the Hill-Mandel condition [26, 68, 67]. As it is common practice with the FEM, the weak form of the momentum balance equation is computed using Gaussian quadrature [31, chapter 9] in each element of the macroscopic finite element grid. Therefore, in each Gaussian quadrature point of the macroscopic grid, the stress tensor required for the macroscopic equation results from the microscopic homogenization procedure described in Section 2.2.4.

For a given macroscopic evaluation point $\overline{\boldsymbol{X}} \in \overline{\mathcal{B}}_0$, the RVE on the microscopic scale is transformed according to the macroscopic deformation gradient

$$\overline{F} = \frac{\mathrm{d}\overline{\boldsymbol{x}}}{\mathrm{d}\overline{\boldsymbol{X}}}$$

at that given point. On the microscopic scale, the deformation of each node within the RVE can be described as the sum of the deformation component that is determined by the macroscopic deformation gradient $\overline{F}$ and a fluctuation field $\tilde{\boldsymbol{u}}$. For a given microscopic evaluation point $\boldsymbol{X} \in \mathcal{B}_0$, this results in the formulation

$$\boldsymbol{x} = \overline{F} \cdot \boldsymbol{X} + \tilde{\boldsymbol{u}}$$

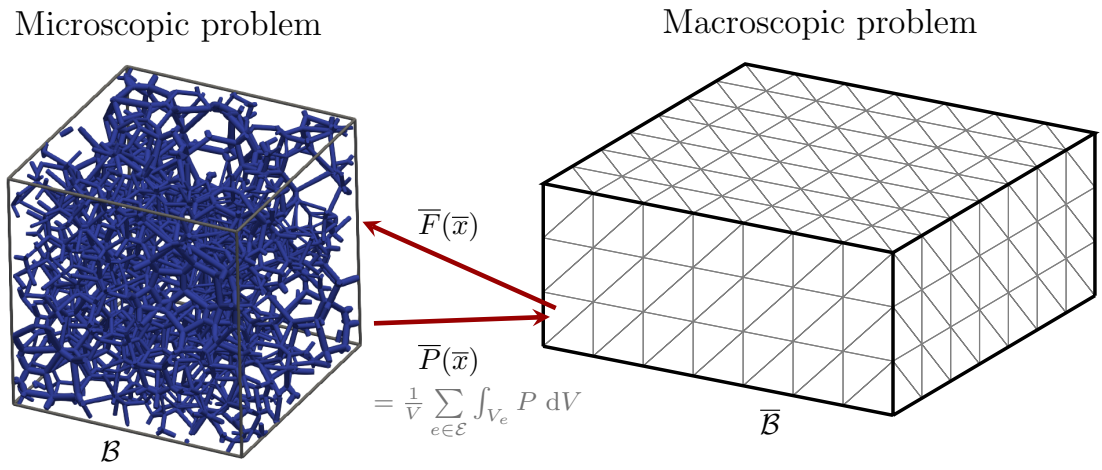


Figure 2.7: Coupling between the macroscopic and microscopic problems in the FE^2 method. Microscopic RVE is deformed according to the deformation gradient at the given macroscopic evaluation point $\overline{\boldsymbol{X}} \in \overline{\mathcal{B}}_0$. Based on the homogenization approach, the macroscopic stress tensor $\overline{P}$ is computed from the deformed RVE.

for the deformed state $\boldsymbol{x} \in \mathcal{B}$. In the multiscale framework described in this chapter, the computation of the fluctuation field $\tilde{\boldsymbol{u}}$ is dependent on the properties of the beam frame structure. A visualization of the dependencies regarding the macroscopic and microscopic boundary value problems is given in Figure 2.7. The image shows the domain of the microscopic problem $\mathcal{B}$ on the left and the domain of the macroscopic problem $\overline{\mathcal{B}}$ on the right.

It is a common approach for the FE^2 method to apply periodic boundary conditions to the boundary points of the microscopic problem. As explained in Section 2.2, the beam frame is designed to feature periodic boundaries. This means that for every node on one boundary of the microscopic domain, there exists a corresponding node on the opposite boundary with the coordinates of both nodes differing only in one entry. With the periodic properties of the RVE, it is possible to directly apply periodic boundary conditions to the respective nodes on the microscopic scale. If we neglect nodes that are in the exact corners of the RVE, it is possible to partition the set of boundary nodes $\partial\mathcal{B}_0$ into two non-overlapping sets of nodes $\partial\mathcal{B}_0 = \partial\mathcal{B}_0^+ \cup \partial\mathcal{B}_0^-$ in a way that for each node in ∂B_0^+ there exists an

associated node in $\partial\mathcal{B}_0^-$ that differs only in one coordinate. For the fluctuation field $\tilde{\boldsymbol{u}}$, we apply periodic boundary conditions

$$\tilde{\boldsymbol{u}}^+ = \tilde{\boldsymbol{u}}^- \quad \text{for each pair of periodic nodes } \boldsymbol{X}^+ \in \partial\mathcal{B}_0^+,\, \boldsymbol{X}^- \in \partial\mathcal{B}_0^-.$$

Additionally, Dirichlet boundary conditions need to be applied to define a regular beam frame problem. Since arbitrary beam frame structures created from the LVT usually do not feature nodes directly in the corners, we consider for each corner of the microscopic domain $\mathcal{B}_0$ the nodes closest to the corner and define the set of nodes as $\partial B_0^* \subset \partial\mathcal{B}_0$. For each respective corner node, we apply Dirichlet boundary conditions by setting the fluctuation to zero. This results in the condition

$$\tilde{\boldsymbol{u}} = \tilde{\boldsymbol{u}}^* = 0 \quad \text{for all boundary nodes } \boldsymbol{X}^* \in \partial\mathcal{B}_0^*.$$

By applying the boundary conditions as described, we obtain a well-defined microscopic problem. This is solved independently in each Gaussian quadrature point of the macroscopic finite element grid. Subsequently, homogenization of the stresses within the RVE yields the respective macroscopic stress values. This procedure describes the relationship of the deformation gradient $\overline{F}$ and the stresses on the macroscopic scale.

2.3.2 Macroscopic Solver

In general, the relationship between the deformation gradient $\overline{F}$ and the macroscopic stress tensor, which is computed from the homogenization approach for the microscopic beam frame problem, is non-linear. Due to this non-linear relationship, the integration of the microscopic problem into the FE^2 method yields a non-linear macroscopic problem. Consequently, a non-linear solver is required to solve the macroscopic boundary value problem. For this application, different non-linear numerical solvers can be considered, and in the scope of this work, we apply various solvers and compare their performance. Regarding the macroscopic boundary value problem as described in the previous section, we use the notation $\overline{R}(\overline{\boldsymbol{U}})$ for the

residual of the macroscopic boundary values problem. This macroscopic residual depends on the deformation state of the nodes within the finite element grid and results from the weak formulations given by Equation (2.16). It is assembled from the local residual terms in each macroscopic finite element as described in Equation (2.2). Based on this definition,

$$\overline{R}(\overline{\boldsymbol{U}}) = 0$$

describes the non-linear relation that is to be solved for the deformation vector $\overline{\boldsymbol{U}}$. This means that solving the macroscopic problem translates to finding the root of the residual $\overline{R}$. The residual function resulting from the finite element discretization is differentiable.

Despite its early development in the 17th century, the Newton-Raphson method [114, chapter 11] is still a state-of-the-art optimization method. In this work, we simply refer to the method as Newton's method as it is common practice in literature. This iterative method requires an initial guess $\overline{\boldsymbol{U}}_0$ and computes the next iterate from the relation

$$\overline{\boldsymbol{U}}_{k+1} = \overline{\boldsymbol{U}}_k - \left(\overline{DR}(\overline{\boldsymbol{U}}_k)\right)^{-1} \cdot \overline{R}(\overline{\boldsymbol{U}}_k)$$

for $k \in \{0, 1, 2, \dots\}$. Here, the Jacobian matrix of the residual with respect to the deformation vector $\overline{\boldsymbol{U}}$ is given by $\overline{DR}$. For each Newton iteration, this tangent is assembled from local tangent terms in each macroscopic finite element as described in Equation (2.3). If the Jacobian of the residual is non-zero at the root and the initial guess $\overline{\boldsymbol{U}}_0$ is within a neighborhood of the solution, Newton's method converges quadratically [114, chapter 11].

Each iteration of Newton's method requires the computation of the Jacobian of the residual $\overline{DR}(\overline{\boldsymbol{U}}_k)$. In the FE^2 multiscale approach, assembling the Jacobian for a given finite element discretization requires the computation of the derivative of the homogenized stress tensor with respect to the deformation gradient $\frac{\partial \overline{P}}{\partial \overline{F}}$ in each Gaussian quadrature point. This dependence is shown in Equation (2.3). Regarding

the beam frame problem on the microscopic scale, we have no explicit formula for computing the derivative of the homogenized Piola-Kirchhoff stress tensor. Therefore, we use central differences for computing an approximation of this modulus for assembling the Jacobian. Depending on the accuracy of the approximation of the Jacobian, the convergence behavior of the solver can be worse than for the regular Newton's method [20].

The term for computing the next Newton iteration includes the inverse of the Jacobian $\overline{DR}(\overline{\boldsymbol{U}}_k)$. In most applications, it is common to not compute this inverse directly, since matrix inversion is considered slow and computationally expensive. For a general square matrix, the number of operations increases in cubic proportion to the size of the matrix. Therefore, the iteration rule is usually reformulated in a way that the inverse of the Jacobian is replaced by solving a system of linear equations. Regarding the Jacobian $\overline{DR}(\overline{\boldsymbol{U}}_k)$ computed from finite element discretizations, the resulting matrix is usually very sparse and solving a system of linear equations for this sparse Jacobian matrix is expected to be much less computationally demanding and more numerically stable than computing the dense inverse matrix. Therefore, we reformulate the iteration rule for each Newton step by the expressions

$$\begin{aligned} \text{solve} \quad \overline{DR}(\overline{\boldsymbol{U}}_k) \cdot \overline{\boldsymbol{S}}_k &= -\overline{R}(\overline{\boldsymbol{U}}_k) \\ \overline{\boldsymbol{U}}_{k+1} &= \overline{\boldsymbol{U}}_k + \overline{\boldsymbol{S}}_k \end{aligned}$$

where the system of linear equations is solved for $\overline{\boldsymbol{S}}_k$ by using a linear solver.

Depending on the size of the macroscopic problem, the computation of the approximate Jacobian $\overline{DR}$ for the non-linear solver can become computationally expensive. Especially for multiscale methods with complex microscopic problems, the computation of the Jacobian often requires the majority of the computation time. To avoid computing the Jacobian, quasi-Newton methods can offer an appropriate alternative. These non-linear solvers are based on similar iteration rules but replace the exact computation of the Jacobian with different approximation techniques for the tangent in the iteration steps.

One of the most popular quasi-Newton methods is the BFGS method [18, 114],

which is named for its discoverers Broyden, Fletcher, Goldfarb, and Shanno. The update rule of the BFGS method is given by the expression

$$\overline{\boldsymbol{U}}_{k+1} = \overline{\boldsymbol{U}}_k + \alpha \cdot p_k$$

for a step size $\alpha > 0$ and the direction of descent p_k in iteration $k \in \{0, 1, 2, \dots\}$ [114, chapter 6]. The direction of descent is computed in each iteration as

$$p_k = -H_k \cdot \overline{R}(\overline{\boldsymbol{U}}_k).$$

Here, the matrix H_k is used as the approximate inverse of the Jacobian for the iteration step k. It is computed successively from its previous iteration with the relation

$$H_{k+1} = \left(I - \frac{\overline{\boldsymbol{S}}_k \overline{\boldsymbol{Y}}_k^{\mathsf{T}}}{\overline{\boldsymbol{Y}}_k^{\mathsf{T}} \overline{\boldsymbol{S}}_k} \right) \cdot H_k \cdot \left(I - \frac{\overline{\boldsymbol{Y}}_k \overline{\boldsymbol{S}}_k^{\mathsf{T}}}{\overline{\boldsymbol{Y}}_k^{\mathsf{T}} \overline{\boldsymbol{S}}_k} \right) + \frac{\overline{\boldsymbol{S}}_k \overline{\boldsymbol{S}}_k^{\mathsf{T}}}{\overline{\boldsymbol{Y}}_k^{\mathsf{T}} \overline{\boldsymbol{S}}_k}.$$

In this formulation, $\overline{\boldsymbol{S}}_k = \overline{\boldsymbol{U}}_{k+1} - \overline{\boldsymbol{U}}_k$ refers to the update of the deformation vector and $\overline{\boldsymbol{Y}}_k = \overline{R}(\overline{\boldsymbol{U}}_{k+1}) - \overline{R}(\overline{\boldsymbol{U}}_k)$ refers to the update of the residual in the respective iteration.

To determine the proper step size α for each iteration of the BFGS method, we apply a line search algorithm based on the Wolfe conditions [148]

1. $\overline{R}(\overline{\boldsymbol{U}}_k + \alpha \cdot p_k)^{\mathsf{T}} \cdot \overline{R}(\overline{\boldsymbol{U}}_k + \alpha \cdot p_k) \leq \overline{R}(\overline{\boldsymbol{U}}_k)^{\mathsf{T}} \cdot \overline{R}(\overline{\boldsymbol{U}}_k) + c_1 \alpha \cdot p_k^{\mathsf{T}} \cdot \overline{R}(\overline{\boldsymbol{U}}_k)$
2. $-p_k^{\mathsf{T}} \cdot \overline{R}(\overline{\boldsymbol{U}}_k + \alpha \cdot p_k) \leq -c_2 \cdot p_k^{\mathsf{T}} \cdot \overline{R}(\overline{\boldsymbol{U}}_k)$

for $0 < c_1 < c_2 < 1$. The Wolfe conditions are commonly used as requirements for controlling the step size in the BFGS method to ensure a sufficient reduction of the residual norm. The first rule is also referred to as Armijo condition [4] and the second rule is referred to as curvature condition. As the Armijo condition alone would ensure an appropriate descent for the residual, the application of the Wolfe conditions often yields better convergence behavior for the non-linear solver. In practice, this means that the step size is initially set to $\alpha = 1$ and it is subsequently reduced until both Wolfe conditions are satisfied. Note that as the curvature condition can

reject small step sizes, it is possible that the valid region for a step size regarding Wolfe conditions is skipped using a simple reduction method for α. Due to this issue, more sophisticated reduction methods are often applied for finding the step size α. More insight with regard to these methods and further studies regarding the zoom algorithm can be found in [114, chapter 3]. However, as the issue of skipping the valid region with Wolfe conditions is considered rare, we use a simple reduction method for deriving the step size α. With regard to our applications for the simulation of open-porous materials, the used method did not induce any undesirable convergence behavior concerning the step size control. However, for different applications, it is generally recommended to use the zoom algorithm when implementing the line search method based on the Wolfe conditions [114, chapter 3].

For the first iteration of the BFGS method, an initial guess of the approximate inverse of the Jacobian H_0 is required. There are different approaches for defining this initial matrix and the performance of each approach often depends on the specifications of the given problem. In many applications, it can be sufficient to use the identity matrix as an initial guess. However, in some cases, defining the initial matrix as an approximation of the inverse Jacobian yields significantly better performance.

For large problem sizes, storing the square matrix H_k in each iteration can take up large amounts of memory storage and slow down the method if the matrix does not fit into the fast cache of the used processor. One solution regarding this issue is provided by the limited-memory BFGS method [99], which is also referred to as L-BFGS. This quasi-Newton method is based on the BFGS approach but omits the explicit computation of the approximate inverse of the Jacobian H_k in each iteration step. Instead of the full square matrix H_k the L-BFGS method stores the last m states $\overline{\boldsymbol{S}}_k, \overline{\boldsymbol{S}}_k - 1, \ldots, \overline{\boldsymbol{S}}_{k-m+1}$ and $\overline{\boldsymbol{Y}}_k, \overline{\boldsymbol{Y}}_k - 1, \ldots, \overline{\boldsymbol{Y}}_{k-m+1}$ for computing the update in each iteration for a given parameter $m \in \mathbb{N}$. Since m is usually set significantly smaller than the problem size, this results in much lower memory requirements.

The computation of the descent direction from the previous m states of $\overline{\boldsymbol{S}}_k$ and $\overline{\boldsymbol{Y}}_k$ uses a two-loop-recursion approach. A detailed description of this algorithm is

presented in [114, section 7.2]. Depending on the application, the L-BFGS method often performs similar or slightly worse than the BFGS method regarding its convergence rate. Specially, for ill-conditioned problems, the L-BFGS method converges slowly [114, section 7.2]. However, this can be a reasonable trade-off considering the faster computation of each update and the lower amount of memory required for storing the iteration variables.

As mentioned above, Newton's method converges quadratically for an initial value that lies within a neighborhood of the optimum. Note that convergence behavior is not guaranteed for initial values outside of this neighborhood. In practice, we observe that, for certain values, the solver does not converge at all. This behavior is similar in the case of quasi-Newton methods that also require an initial value close to the optimum. To overcome this limitation, load stepping schemes are often applied when non-linear solvers fail to compute the solution for large deformations. These methods apply the load of large deformations in multiple small steps. In each load step, the solution is computed using the non-linear solver and the obtained solution is used as the initial value for the next load step. This ensures that the initial value for the non-linear solver is always close to the optimum when the load steps are small enough. For the multiscale approach, we apply an adaptive load stepping scheme that automatically reverses the solver updates and reduces the load step if the solver does not converge. Conversely, the step size increases if fast convergence is observed for previous load steps.

2.4 Algorithmic Description

In the previous sections of this chapter, we covered the different aspects of the multiscale approach on the macroscopic and microscopic levels. With the given introduction to the method, it is possible to formulate a coherent algorithm for solving a given multiscale problem with beam frame microstructure. The pseudocode given in Algorithm 1 describes the required steps for computing the macroscopic deformation vector $\overline{\boldsymbol{U}}$ for one load step considered in the adaptive load stepping scheme.

The algorithm requires a macroscopic finite element discretization of a given boundary value problem, a beam frame model that defines the microscopic problem as introduced in Section 2.2, and a non-linear solver as input. We consider Newton's method, BFGS, or L-BFGS, which are described in Section 2.3.2, as non-linear solvers.

Algorithm 1 FE^2 multiscale method with a beam frame model on the microscopic scale. The algorithm describes the computations for a single load step.

Input: macroscopic boundary value problem, microscopic beam frame model, non-linear solver with tolerance $\bar{\varepsilon}$ and maximum number of iteration k_{max}

Output: macroscopic deformation vector $\overline{\boldsymbol{U}}$

Set initial guess $\overline{\boldsymbol{U}}_0$
for $k = 0, 1, 2, \ldots$ **do**
  / iteration of macroscopic solver */*
  for all macroscopic integration points $\overline{\boldsymbol{X}}$ **do**
    / microscopic computations */*
    Compute $\overline{F}(\overline{\boldsymbol{X}})$ according to deformation state $\overline{\boldsymbol{U}}_k$
    Solve beam frame problem resulting from deformation gradient $\overline{F}(\overline{\boldsymbol{X}})$
    Compute homogenized stress $\overline{P}$ from deformed RVE
  end for
  Assemble residual vector $\overline{R}(\overline{\boldsymbol{U}}_k)$ from stress values $\overline{P}$
  if $\left\|\overline{R}(\overline{\boldsymbol{U}}_k)\right\|_2 \,/\, \left\|\overline{R}(\overline{\boldsymbol{U}}_0)\right\|_2 < \bar{\varepsilon}$ **then**
    / stopping criterion of non-linear solver is satisfied */*
    Break for loop
  end if
  Compute update of non-linear solver $\overline{\boldsymbol{S}}_k$
  Update $\overline{\boldsymbol{U}}_{k+1} = \overline{\boldsymbol{U}}_k + \overline{\boldsymbol{S}}_k$
  if $k \geq k_{max}$ **then**
    / macroscopic solver does not converge */*
    Repeat computations with reduced load
  end if
end for
Return solution $\overline{\boldsymbol{U}} = \overline{\boldsymbol{U}}_k$

Chapter 3

Machine Learning-Based Surrogates for Microscale Homogenization

In the scope of Chapter 2, a modification of the FE^2 approach is described that considers a beam frame model on the microscopic scale. The beam frame structure is well suited for modeling the open-porous nanostructure of aerogels, though solving the beam frame problem in each Gaussian quadrature point of the macroscopic mesh can be computationally expensive. The computational effort also increases with the size of the considered RVE. For extensive microscopic RVEs, which yield an appropriate representation of the material's pore size distribution, the related system of equations becomes very large and even in the linear case solving the microscopic problem becomes computationally demanding. Replacing an expensive microscopic problem by a surrogate model is an established approach in the recent literature regarding multiscale methods like FE^2.

In general, a surrogate model describes an approach to approximate the outcome of a complex model which cannot be easily computed. The surrogate model is used to replace the complex original model with the purpose of reducing the computational effort. Machine learning-based surrogate models are widely used. These surrogate models show good versatility regarding different applications and they can be fully data driven which makes them relatively easy to implement. For

many applications, dense feed-forward neural networks (NN) are used as basis for the surrogate model. These networks usually feature a relatively simple architecture and process the data only in one direction from input to output. In the literature, the dense feed-forward neural network is also referred to as multi-layer perceptron (MLP).

The application of dense feed-forward neural networks (NN) in the FE^2 framework is a common approach to reduce the computational effort of the multiscale simulations as presented in [97, 50, 62, 34, 52, 127, 36, 126, 42, 113, 30]. There are two main requirements for a surrogate model to be applied in the FE^2 framework. It is expected to provide the solution to a given microscopic problem with a significantly reduced computational effort compared to the original model. Furthermore, the solution computed by the surrogate model needs to be an accurate approximation of the reference solution. Since in many cases the exact solution to a given microscopic problem is not known, the surrogate solution is usually compared to the solution computed with the original finite element solver.

Our work differs from the approaches in [97, 50, 62, 34, 52, 127, 36, 126, 42, 113, 30] by the nanoporous structure of the material, which we model with the beam frame structure described in Section 2.2. Regarding surrogate models for open-porous materials, a data-driven machine learning approach is applied in [89] for learning the mechanical properties of nanoporous material on the microscale. However, in comparison to our work, the authors consider an extensive finite element discretization of the RVE. This leads to a significantly higher computational effort for generating the training data and performing macroscopic comparative computations with the original model also becomes computationally very expensive.

For this chapter, we consider different machine learning models for replacing the beam frame model on the microscopic scale of the FE^2 approach described in Chapter 2. The general approach of using a machine learning-based surrogate model to replace the beam frame problem on the microscopic scale is presented in Figure 3.1. The figure illustrates the coupling of the macroscopic finite element problem and the surrogate model on the microscopic scale. The macroscopic deformation gradient is used as an input of the microscopic model and based on this input, the

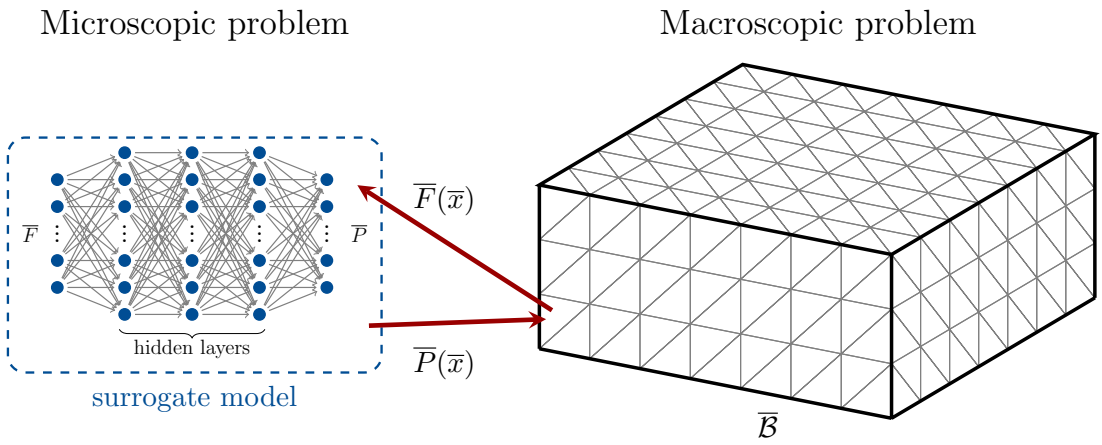


Figure 3.1: Coupling between the macroscopic finite element problem and the machine learning-based surrogate model on the microscopic scale. The macroscopic deformation gradient is given as the input of the surrogate model. Based on the input, the homogenized stress tensor is computed by the surrogate model and used for the finite element computations on the macroscopic scale.

surrogate model computes the homogenized stress tensor. Similar to the case with the beam frame solver on the microscopic scale, the stress resulting from the microscopic problem is used for the finite element computations on the macroscopic scale. Since the surrogate model is applied to the microscopic problem of the multiscale method, this approach is generally independent of the macroscopic geometry. This means that the approach is applicable to any macroscopic boundary value problem and in each case, it is expected to yield similar performance. To evaluate this generalization property, we compute the error induced by the surrogate models for different macroscopic problem sizes and compare the results.

In the scope of Section 3.1, we give a brief introduction regarding the general structure of a muli-layer perceptron (MLP) and the training process. Sections 3.2 and 3.3 deal with the application of MLPs for predicting the resulting homogenized first Piola-Kirchhoff stress tensor for a given RVE. In Section 3.2, we discuss the application of the surrogate model in the two-dimensional case and Section 3.3 is focused on the three-dimensional case. To evaluate the quality of given surrogate models, we regard both the microscopic and the macroscopic performance. On

the microscopic scale we compute the model's deviation from the original beam frame model. Additionally, we apply the surrogate model within the multiscale FE^2 method and compare the results with the same macroscopic setup that uses the beam frame approach on the microscopic scale. Here, we evaluate the performance of a surrogate model for different test cases. Additionally, this yields a proper measurement for the model's accuracy in real-world applications. Even though the main focus of this chapter is the prediction of the first Piola-Kirchhoff stress tensor, Section 3.4 discusses the reduction of the computational effort for the homogenization of the Cauchy stress tensor.

3.1 Multi-Layer Perceptron

The multi-layer perceptron (MLP) [59, chapter 6] is one of the simplest forms of a NN. Due to its feed-forward processing of the data from one layer to the next, the implementation of the model is relatively straightforward compared to other machine learning models. Despite its simplicity with regard to its structure, with a possibly great number of layers and neurons per layer, the MLP can still prove beneficial for complex applications with large problem sizes. Because of these properties the MLP architecture is a common choice for developing a machine learning model. Sophisticated machine learning approaches are often based on MLPs and incorporate these structures within their architecture.

3.1.1 Model Structure

In general, an MLP can be interpreted as a function $f_{MLP} : \mathbb{R}^{n_{in}} \to \mathbb{R}^{n_{out}}$ that takes an input vector $x \in \mathbb{R}^{n_{in}}$ and computes the corresponding output vector $f_{MLP}(x) = y \in \mathbb{R}^{n_{out}}$. For computing the output y of an MLP, the input information x is processed in one or more iterative steps, called layers. The general structure of an arbitrary MLP with different layers is illustrated in Figure 3.2. The arrows represent the processing of data from one layer to another and indicate the typical property of an MLP that information is strictly processed in forward di-

rection from the input to the output layer. The processing layers between input and output are referred to as *hidden layers*. This naming indicates that the output of these hidden layers is usually not observable in the evaluation of an MLP. In Figure 3.2, each dot represents a single value within the output vectors of a layer and in the framework of NNs, they are also referred to as neurons. For describing the computation rules for the evaluation of an MLP, we consider an arbitrary MLP architecture with the number of layers given by $n_{layers} \in \mathbb{N}$. For every layer with the index $i \in \{0, 1, \ldots, n_{layers} - 1\}$, the neurons $x^{(i+1)} \in \mathbb{R}^{n^{(i+1)}}$ in the layer with index $i + 1$ are computed from the configuration of the neurons $x^{(i)} \in \mathbb{R}^{n^{(i)}}$ in the previous layer i with the formula

$$x^{(i+1)} = \phi^{(i+1)} \left(W^{(i+1)} \cdot x^{(i)} + b^{(i+1)} \right). \tag{3.1}$$

Here, $W^{(i+1)} \in \mathbb{R}^{n^{(i+1)} \times n^{(i)}}$ refers to the weight matrix, $b^{(i+1)}$ to the bias and $\phi^{(i+1)}$ to the activation function of layer $i + 1$. For the layer with index $i = 0$, the configuration of the neurons is defined by the input vector with $x^{(0)} = x$. Evaluating the full MLP for computing the values of the final output layer from a given input requires the successive computing entries of each layer of the MLP. This results in iterative matrix-vector multiplications, addition of the bias, and evaluation of the respective activation functions. Note that it is a common convention to express the multiplication of the weight matrix and the addition of the biases as a single matrix-vector product by

$$W^{(i+1)} \cdot x^{(i)} + b^{(i+1)} = \hat{W}^{(i+1)} \cdot \hat{x}^{(i)}.$$

Here, the weights and biases are combined in the matrix $\hat{W}^{(i+1)} \in \mathbb{R}^{n^{(i+1)} \times (n^{(i)}+1)}$ and the vector $\hat{x}^{(i)} \in \mathbb{R}^{n^{(i)}+1}$ contains the state of layer i extended by an entry equal to one. This results in

$$\hat{W}^{(i+1)} = \left(W^{(i+1)}, b^{(i+1)} \right) \in \mathbb{R}^{n^{(i+1)} \times (n^{(i)}+1)} \quad \text{and} \quad \hat{x}^{(i)} = \begin{pmatrix} x^{(i)} \\ 1 \end{pmatrix} \in \mathbb{R}^{n^{(i)}+1}.$$

In many applications, the activation functions of an MLP add non-linearity to the model. Since multiplying the weights and adding the bias only yields linear operations, a non-linear activation function is required so that the network is capable of replicating non-linear relationships. Regarding the training of an MLP, for most optimizers, it is important that the weak derivative of all activation functions can be directly computed. The following list of common activation functions that are used in many applications provides the mathematical description of each function:

- linear: $\phi(x) = x$
- ReLU: $\phi(x) = \max(0, x)$
- tanh: $\phi(x) = \tanh(x)$
- sigmoid: $\phi(x) = \frac{1}{1+exp(-x)}$
- GELU: $\phi(x) = x \cdot \Phi(x)$ (with the Normal Cumulative Density Function Φ).

The respective activation functions can play a significant role for the performance of an MLP. For a given application, the output range and behavior of the activation functions regarding large positive and negative values need to be considered.

3.1.2 Training Process

The machine learning approach which is focused on in this section is fully data-driven. This means that, for the training of the MLP, data sets are provided for the input and also for the target output. For training the model, a regression problem of minimizing the distance between the target output data and the output of the MLP for the given input needs to be solved. The distance between the output data and the models output is commonly defined using a loss function

$$\ell : \mathbb{R}^{n_{out}} \times \mathbb{R}^{n_{out}} \to \mathbb{R}_{\geq 0},\ (y, y^*) \mapsto \ell(y, y^*),$$

and for a given input value x and target output y^*, the loss is computed from $\ell(f_{MLP}(x), y^*)$. In many data-driven cases, the loss function is defined from a

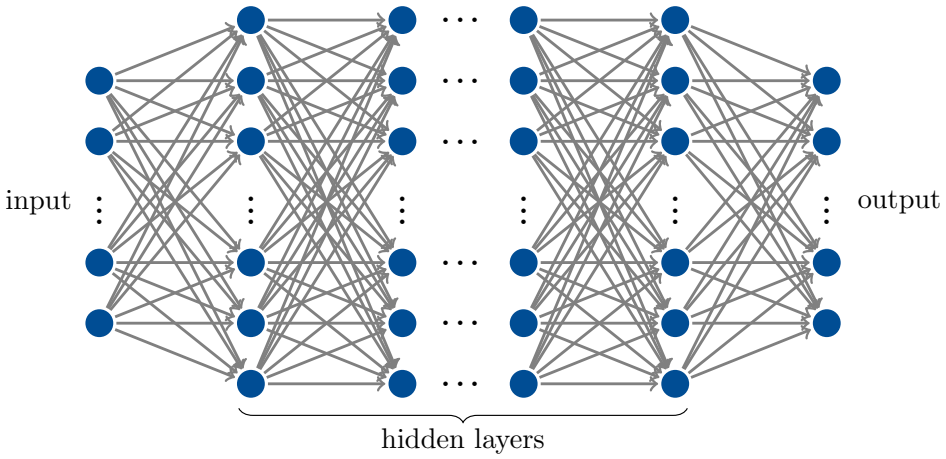


Figure 3.2: General structure of an arbitrary multi-layer perceptron. Vertically aligned dots refer to layers within the MLP structure and each gray arrow connecting two dots refers to a weight value.

metric like for example the metric induced by the L_1 or the L_2 norm. For regression problems, the mean squared error (MSE) is a common choice for training machine learning models. Given an input $x \in \mathbb{R}^{n_{in}}$ and a corresponding target output vector $y^* \in \mathbb{R}^{n_{out}}$, this loss term is defined by the expression

$$\ell_{MSE}(f_{MLP}(x), y^*) = \sum_{i=0}^{n_{out}-1} \left(f_{MLP}(x) - y^*\right)^2 .$$

In the scope of this work, the MSE is commonly applied as the loss function regarding the training of the machine learning models. Unless otherwise specified, the MSE loss is optimized in each of the described training processes.

For training the MLP, the loss function ℓ is minimized using iterative optimization methods. In the training process, the weights $W^{(i)}$ and the biases $b^{(i)}$ for each layer i are optimized. In the field of machine learning, each iteration of the optimization process in which the weights and biases are adjusted is referred to as an epoch. For most MLP applications, the iterative solver is based on stochastic gradient descent. Optimization methods for these applications typically include stochastic gradient descent, with and without momentum [118], Adagrad [33], and Adam [80].

These solvers require the derivative of the loss function ℓ in each iteration. If the derivatives of the activation functions of an MLP are known, the derivative of the loss function can be computed directly using backpropagation [128]. In some applications, the use of L-BFGS [99] as an optimizer has also proven to be beneficial. As explained in Section 2.3.2, this Quasi-Newton method approximates the derivative of the loss function and does not require its computation in each iteration.

As the training approach for the surrogate models is data-driven, input and output data sets are required for the optimization of the models. For the microscopic beam frame problem described in Section 2.2, this means that we generate data from solving several microscopic problems based on the original beam frame model. The target output data for the training is then given by the average stress computed from the deformed RVE as described in Section 2.2.4. For generating the training data, we consider randomized macroscopic sample problems which we solve using the FE^2 with the beam frame solver for the respective RVE. In each step of the iterative macroscopic solver we store the deformation gradients that are applied in each Gaussian quadrature point as well as the resulting stress tensor which is computed from averaging the stresses in the deformed beam frame RVE. This method ensures that the generated training data contains deformations that are relevant for the macroscopic solver. By computing a large number of random, macroscopic sample cases, the approach yields diverse training data. Providing training data with a wide range of microscopic deformations is crucial since it improves the models ability to generalize to unseen deformations.

3.2 Surrogate for Two-Dimensional Microstructure

We first consider the MLP approach for a given two-dimensional beam frame RVE. For most applications, it is much more realistic to consider a three-dimensional nanostructure. For the beam frame RVE modeling the open-porous nanostructure of aerogels, the difference between two-dimensional and three-dimensional computations is especially evident, since the three-dimensional heterogeneity of the porous

material cannot be directly transferred to the two-dimensional case. Therefore, it is expected that the two-dimensional and three-dimensional RVEs show different mechanical responses. Despite the consideration that the two-dimensional case offers few opportunities for practical application in real-world examples, it can still be helpful to consider such examples as proof of concept. Two-dimensional RVEs consist of a much lower number of beam elements and nodes which implicates a significantly reduced computational effort compared to the three-dimensional case. Additionally, the deformation gradient as well as the stress tensor each contain only four values each instead of nine values in the three-dimensional case. This reduces the input and output dimensions of the MLP that is to be trained on the relationship between deformation gradient and the resulting stress values. Formally, the MLP for the two-dimensional case can be described as the function

$$f_{MLP} : \mathbb{R}^{2\times 2} \to \mathbb{R}^{2\times 2}, \quad \overline{F} \mapsto f_{MLP}\left(\overline{F}\right) = \overline{P}$$

with the macroscopic deformation gradient $\overline{F} \in \mathbb{R}^{2\times 2}$ and the homogenized first Piola-Kirchhoff stress tensor $\overline{P} \in \mathbb{R}^{2\times 2}$.

Large parts of this section are based on the results of [82]. We consider a linear beam frame problem as given by Equation (2.9) and develop a surrogate model to replace the beam frame solver as well as the computations for the homogenization on the microscopic scale. The considered beam frame structure consists of 225 beam elements and 172 vertices. For Equation (2.9), this results in a solution vector $\boldsymbol{U} \in \mathbb{R}^{516}$, which contains two deformation components and one rotation variable for each node. The beam frame structure of this specific RVE in the two-dimensional case is presented in Figure 3.3.

3.2.1 Training Data

For creating the training data set for the multi-layer perceptron, we run several FE^2 simulations with the described beam frame problem considered on the microscopic scale. As described above, this procedure is supposed to ensure that the data set

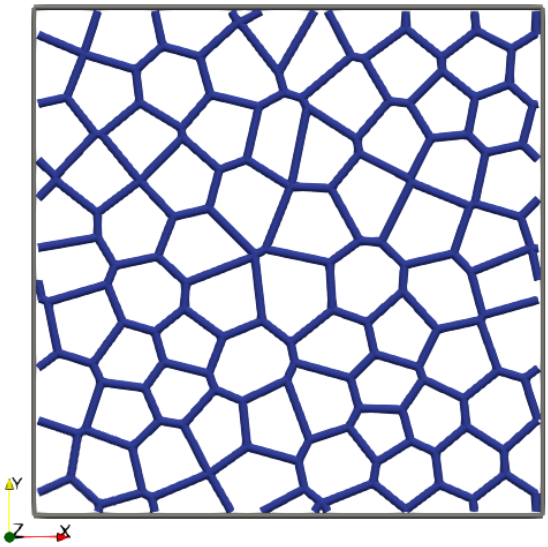


Figure 3.3: Two-dimensional beam frame-structured RVE.

contains relevant deformation examples for the macroscopic simulations and by running a significant number of test cases, the data set is expected to feature a variety of microscopic deformations.

For the two-dimensional multiscale simulations, we consider a rectangular geometry that is discretized with a coarse finite element mesh of 40 elements. Linear, triangular finite elements, which are also referred to as P1 elements, are used for this macroscopic discretization. This results in a total of 60 degrees of freedom (DOF) on the macroscopic scale. The training data set is created using seven different macroscopic test cases, each of which is defined by individual boundary conditions. Results for the seven test cases computed with the microscopic beam frame solver are presented in Figure 3.4. By defining Dirichlet boundary conditions for a different selection of boundary nodes in each instance, the deformation is specified for each of the test cases shown. To obtain a wider range of data points in the training data set, multiple simulations are computed for each test case that vary with regard to the orientation and the magnitude of the macroscopic deformation. The boundary values for each specific run are randomly defined. During the computation of the multiscale simulations, in each iteration of the macroscopic solver, the macroscopic

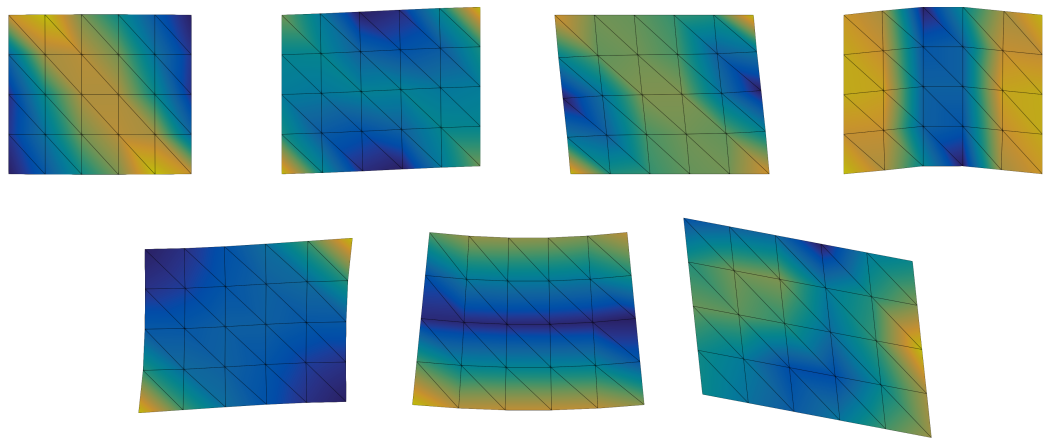


Figure 3.4: Deformation cases used for creating the training data set with respect to the two-dimensional RVE. Seven different deformation cases are considered on the macroscopic scale. The orientation and the magnitude of the deformation are randomly modified for a larger variety of data points. The images have first been published in [82].

deformation gradient and the first Piola-Kirchhoff stress tensor resulting from the homogenization method are stored for the input and output data sets.

Overall, we end up with a data set consisting of about 110,000 distinct pairs of macroscopic deformation gradients $\overline{F}$ and stress tensors $\overline{P}$. Any duplicate data points which could have emerged from the data acquisition process are removed from the data in a preprossessing step. The resulting data set is used as the training data set for optimizing the weights of the MLP.

In addition to the described training data set, we create a validation data set using the same procedure of running multiscale simulations. The validation data set consists of about 12,000 deformation and stress pairs, which are all distinct from the data points represented in the training data set. The creation process of the validation data ensures that the data set contains no deformation sample which is also part of the training data set. During NN training, the loss values based on the NN's current state are tracked for both the training and validation data sets in each training epoch. Since the weights of the NN are only adjusted based on the training data, the loss values regarding the validation data can be useful to indicate how the

NN performs on unknown data which has not been included in the training data. During training, it is desired that the validation loss progresses the same as the training loss. If the NN performs significantly worse on the validation data than on the training data, this indicates that the model is overfitting the training data. This is an important issue in machine learning approaches. Overfitting describes how models adjust their weights in order to precisely replicate training data points, yet fail to predict outputs that were not included in the training data. Monitoring this behavior with the validation data is crucial during the training process. Therefore, regarding the evaluation of a machine learning model, the validation loss is usually the more important metric than the training loss. Additionally, the validation loss can often indicate the model's general behavior regarding real applications.

The distribution of the training data is presented in Figure 3.5. The figures show histograms for each of the input and output variables. The four histograms on the left show the distribution of the components of the deformation gradient $\overline{F}$. For each of the input variables, a relatively even distribution around the mean of zero can be observed. The standard deviation is similar for each of the input variables and varies in the range between 0.19 and 0.24. Similar to the visualization of the input data, the distribution of the output data is presented in the histograms on the right. The figures show the arrangement of the output components, which are each distributed around the mean of zero. For each of the components of the Piola-Kirchhoff stress tensor, the standard deviation is in the range between 0.04 and 0.09. Despite the apparent accumulation of output data around the mean values, the figures indicate the presence of few outliers.

Note that the presented distributions refer to the unprocessed data sets. For training the MLP, the training and validation data sets are normalized in an additional processing step. In the field of machine learning, the normalization of the training and validation data sets is a common tool to improve the training behavior of the NN.

The correlation of the input and output data is presented in Table 3.1. The values in the table refer to the respective Pearson correlation coefficients resulting from the correlation matrix with respect to the input and output data. The val-

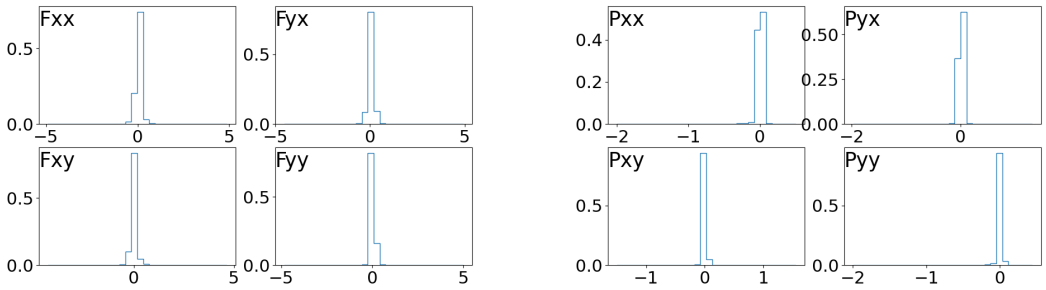


Figure 3.5: Distribution of the data created for the training of the two-dimensional MLP. Histograms on the left show the distribution of the input data and histograms on the right show the distribution of the target output data. Images and data have first been published in [82].

	$\overline{P}_{xx}$	$\overline{P}_{yx}$	$\overline{P}_{xy}$	$\overline{P}_{yy}$
$\overline{F}_{xx}$	**0.39**	0.09	0.20	0.10
$\overline{F}_{yx}$	0.07	**0.50**	**0.46**	0.05
$\overline{F}_{xy}$	0.07	**0.35**	**0.52**	0.16
$\overline{F}_{yy}$	0.00	0.18	0.22	**0.28**

Table 3.1: Correlation coefficients between the input and output variables for the data set resulting from the two-dimensional RVE. Data has first been published in [82].

ues show a significant correlation between the deformation gradient and the first Piola-Kirchhoff stress tensor. For each component of the deformation gradient, the correlation with the stress component that refers to the same coordinates is highlighted in the table. These variables also show the most significant positive correlation.

3.2.2 Training and Microscopic Behavior

For defining the architecture and training the NN, a set of hyperparameters needs to be selected by the user. These hyperparameters typically feature values like the number of hidden layers, the size of each hidden layer, and the respective activation function, which defines the model's architecture. Additionally, values like the batch size used for partitioning the training data or the learning rate used by the opti-

mizer are important hyperparameters required for the training process. For some of these hyperparameters, it is sufficient to set the values manually. However, for some parameters with a wider range of sensible values, hyperparameter optimization can be useful. This concept and a number of well-known algorithms used for the optimization of hyperparameters in machine learning are described by Goodfellow [59, chapter 11] and by Bergstra et al. [9]. Notable algorithms for hyperparameter tuning are grid search, random search, and Bayesian optimization.

To determine which NN architecture fits the two-dimensional problem, we consider a grid search algorithm [59, chapter 11] which compares the performance of different model architectures with each other. The grid search algorithm is a well-known approach for optimizing the hyperparameters regarding the training of a machine learning model. For the NN model considered for the two-dimensional RVE, we investigate the number of hidden layers, the number of neurons per layer, and the activation function used in the hidden layers as hyperparameters for the grid search. The activation of the output layer is generally defined to be linear. With manual testing, the batch size is set to 256 and the optimizer used for training is defined as the Adam optimizer [80]. The mean squared error defines the loss term that is to be minimized during training. The learning rate of the Adam optimizer is set with a learning rate scheduler which exponentially reduces the learning rate from $1e-3$ to $1e-5$ over all training epochs. A gradual reduction of the learning rate is expected to prevent large jumps in the loss curve towards the end of the training process. Each model is trained for a total number of 1500 epochs. With regard to the implementation of training and grid search algorithms, we use the function provided by the open-source package TensorFlow [1].

The results of the grid search are presented in Table 3.2. As the table shows, up to three hidden layers are considered for the model and the set of possible activation functions is defined to feature *sigmoid*, *tanh*, or *GELU*. Furthermore, the possible numbers of neurons for each layer are 64, 128, 256, or 512. In general, the lowest loss values are obtained with the *GELU* activation function. Within the section of the *GELU* activation function, the lowest value is obtained from the configuration with three layers and 128 neurons per layer. The value is highlighted in Table 3.2. The

activation	neur./layer	one layer	two layers	three layers
sigmoid	64	2.24e-04	8.99e-06	5.06e-06
	128	3.36e-04	3.37e-06	9.68e-07
	256	7.95e-04	1.18e-06	6.46e-07
	512	8.13e-04	4.52e-07	6.23e-07
tanh	64	3.20e-05	1.66e-06	6.68e-07
	128	3.36e-05	2.65e-06	1.03e-06
	256	9.12e-05	2.42e-06	4.20e-06
	512	1.26e-04	1.41e-06	3.62e-06
gelu	64	2.19e-07	1.69e-08	7.13e-09
	128	1.27e-07	2.58e-08	**5.76e-09**
	256	8.11e-08	8.53e-08	8.36e-09
	512	5.33e-08	1.85e-07	1.08e-08

Table 3.2: Results of the grid search to optimize the model architecture of the MLP for the two-dimensional RVE. Data has first been published in [82].

corresponding NN achieved a final loss value of $5.76e-9$ after training. Similar loss values are also achieved by other models with $GELU$ activation and three hidden layers. Both neural network with 64 and 256 neurons per layer reduced the loss below $1e-8$. With further manual testing, we decided to use a mixture of the hyperparameters of these models that showed the best reduction in loss. Thus, the final model for the two-dimensional case, which we use for further testing, is built from three hidden layers with 128, 256, and 128 neurons in the respective layers.

The progression of the training and validation loss during training the model for the two-dimensional RVE is presented in Figure 3.6. Based on the grid search results and further manual testing, the model's architecture is set to feature three hidden layers with $GELU$ activation and 128, 256, and 128 neurons. The model is trained for 2000 epochs with Adam optimizer and a learning rate scheduler as described above. The training and validation losses decrease significantly and end up both in the order of $1e-9$. The validation loss in Figure 3.6 behaves very similar to the training loss. This indicates that the model is able to generalize with regard to different deformation data and it is not much affected by overfitting.

Regarding the considered data, the low loss values presented in Figure 3.6 indi-

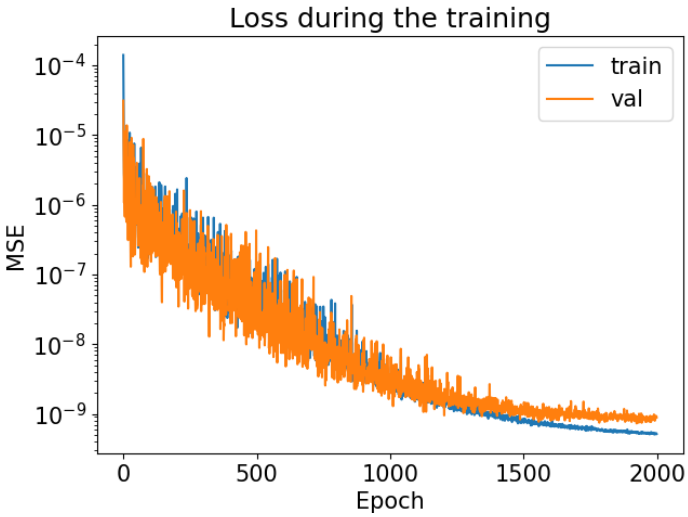


Figure 3.6: Loss curves for the MLP fitted to stress data from the two-dimensional RVE. Blue curve shows the progression of the training loss and orange curve shows the progression of the validation loss. Image and loss data have first been published in [82].

cate that the trained model shows only minor deviation from the stresses resulting from the beam frame model. However, in addition to the evaluation of the loss regarding the deviation to the beam frame model, the required computation time is also significant for the evaluation of the surrogate model. The evaluation of the computation times is especially crucial, as the main purpose of developing a surrogate model, which replaces the beam frame problem on the microscopic scale, is the reduction of the computational effort. Since the computational effort with regard to the full multiscale simulation is topic of Section 3.2.3, we consider in this part only the difference between the computational effort on the microscopic scale.

For a given beam frame model, the computation of the homogenized first Piola-Kirchhoff stress tensor from the macroscopic deformation gradient mainly includes two steps. In the first step, the microscopic beam frame problem is solved to obtain the deformation and rotation values for each node within the RVE and in the second step which refers to the homogenization process, for each beam element the integral over the stress tensor is computed based on the global deformation

vector. As described in Section 2.2.4, the sum over the integrated stress tensors divided by the RVE volume yields the homogenized stress tensor. Note that the computational effort of solving the beam frame problem is mainly dependent on the number of nodes within the RVE, as this defines the dimensions of the matrix. On the other hand, the computational effort regarding the homogenization process is mainly dependent on the number of beam elements within the RVE.

Regarding the multiscale simulation, the microscopic beam frame problem requires solving a system of equations at each Gaussian quadrature point. Each of these systems of equations is defined by the same stiffness matrix but a different right-hand side. In the case of the linear beam frame problem, it can be beneficial to explicitly compute the LU decomposition of the stiffness matrix [58, chapter 3]. This yields a significant potential of reducing the computation time for the microscopic computations in each quadrature points. With the given LU decomposition of the stiffness matrix

$$\boldsymbol{K} = \boldsymbol{L}_{\boldsymbol{K}} \cdot \boldsymbol{U}_{\boldsymbol{K}},$$

computing the global deformation vector for the RVE requires only a forward and backward substitution with the triangular matrices $\boldsymbol{L}_{\boldsymbol{K}}, \boldsymbol{U}_{\boldsymbol{K}} \in \mathbb{R}^{n_N \times n_N}$ instead of the application of a linear solver. In case of the LU decomposition with pivoting, an additional matrix multiplication with the sparse pivot matrix is required. This reduces the complexity of the computation from cubic complexity $\mathcal{O}(n_N^3)$ in the case of solving the linear problem from the ground up with Gaussian elimination [58, chapter 3] to quadratic complexity $\mathcal{O}(n_N^2)$ for the forward and backward substitution. Here, $\mathcal{O}$ refers to the Landau notation. A detailed description of the notation can be found in [64, chapter 1].

On the one hand, the explicit computation of the triangular matrices resulting from LU decomposition can reduce the computation time for solving each microscopic problem. However, on the other hand, this approach is only applicable in case of the linear beam frame problem as an iterative solver would be used in case of the non-linear beam frame problem. Here, the stiffness matrix would not be constant. This means that the computation time of the linear solver for computing

the microscopic deformations is closer to the general case of a microscopic problem. As a result, for the comparison with the surrogate model, we consider both the approach of solving each system of linear equations separately and the approach of explicitly computing the LU decomposition once for all microscopic problems.

We consider a MATLAB [143] implementation of the beam frame solver and also of the evaluation of the MLP. For solving the system of linear equations resulting from the linear beam frame problem, we use MATLAB's backslash operator, which itself uses the UMFPACK solver [27] for solving linear systems of equations. The UMFPACK solver is considered an efficient and fast implementation for solving linear systems of equations. For the computation of the homogenized first Piola-Kirchhoff stress tensor based on MATLAB's backslash operator, we see a 400 times higher computation time compared to the evaluation of the MLP. Here, solving the system of linear equations with the backslash operator required the clear majority of the computation. If we explicitly compute the LU decomposition of the stiffness matrix and consider only the forward-backward substitution for solving the system of linear equations, the computation time is significantly lower, though still ten times higher than the computation time required for evaluating the MLP.

The difference between the computation time required for computing the homogenized stress tensor based on the beam frame problem and the computation time required for evaluating the MLP is expected to be even larger if more complex beam frame structures are considered. As the computational effort for computing the stresses based on the beam frame model increases with an increasing number of beam elements and nodes in the RVE, the computation time of evaluating the MLP does not depend on the size of the RVE. It is to be expected that this difference will become even more pronounced in the three-dimensional case, as the beam frame structures generally show higher complexity.

3.2.3 Application in the Multiscale Framework

The main purpose of developing the surrogate model is to use it as a replacement of the beam frame model within the FE^2 framework. To evaluate the performance

of the MLP with regard to the application in the multiscale simulation method, we consider macroscopic test problems and compare the behavior of the FE^2 method with the MLP on the microscopic scale to the behavior of the original FE^2 method with the beam frame problem on the microscopic scale as described in Chapter 2. On the macroscopic scale, we regard different discretizations for the finite element mesh and compare the behavior for smaller and larger problem sizes.

For the multiscale simulations computed for the comparison of beam frame model and surrogate approach, a square domain is considered on the macroscopic scale with Dirichlet boundary conditions applied to both opposing sides at the boundaries of the x-dimension. The nodes on the left and right boundary are deformed in x-direction depending on the corresponding y-coordinate. This means that the nodes with higher y-coordinate value are deformed toward the middle of the geometry and the nodes with lower y-coordinate value are deformed away from the middle of the geometry. This results in a tilting of each of the boundaries affected by the Dirichlet boundary conditions. The results of this simulation computed with the beam frame solver are presented in Figure 3.7. The colors in the figure illustrate the distribution of the von Mises stress [109] within the finite element mesh. For a point in the solution of the two-dimensional problem, the von Mises stress measurement is defined as

$$\overline{\sigma}_{vM} = \sqrt{\overline{\sigma}_{xx}^2 + \overline{\sigma}_{yy}^2 - \overline{\sigma}_{xx} \cdot \overline{\sigma}_{yy} + 3\,\overline{\sigma}_{xy}^2}$$

based on the respective Cauchy stress tensor $\overline{\sigma} \in \mathbb{R}^{2\times 2}$. In mechanical simulations, the von Mises stress is a popular tool to illustrate in which regions of a geometry high stress values occur. However, the value does not give any information about the direction of the stress. As the part of the geometry close to the horizontal centerline is the least affected by the deformation, we see the lowest von Mises stress values in this region. Higher stresses are observed closer to the boundaries with regard to the y-coordinates. The considered boundary conditions have also been used for the creation of the training data set. However, the magnitude of the deformation at the boundaries and the macroscopic geometry are different to the

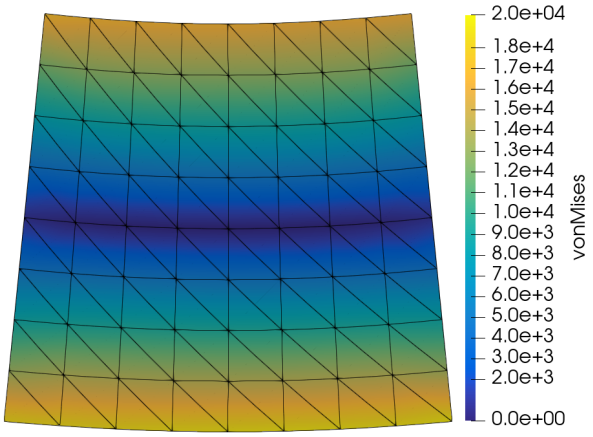


Figure 3.7: Macroscopic solution of the considered test problem for the two-dimensional case. The presented solution refers to the problem size with 162 DOF. Colors in the image refer to the resulting von Mises stress at the respective point in the geometry. Image has first been published in [82].

setups considered for creating the training data.

As described in Section 2.3.2, the macroscopic problem considered within the FE^2 framework can be solved using Newton's method or quasi-Newton methods like BFGS. As the consistent tangent is not known for the beam frame problem on the microscopic scale, we approximate the required derivatives using central differences in case of Newton's method. This approximation is not needed in case of the microscopic computations based on the MLP. For the surrogate model, the derivative of the output with respect to the input vector is well known, as the considered activation functions are differentiable. Based on the description in Equation (3.1), the derivative of a single layer update of an MLP with respect to the state vector $x^{(i)}$ can be written as

$$\frac{\partial x^{(i+1)}}{\partial x^{(i)}} = \operatorname{diag}\left(\phi'\left(W^{(i+1)} \cdot x^{(i)} + b^{(i+1)}\right)\right) \cdot W^{(i+1)}$$

with ϕ' denoting the weak derivative of the activation function and diag referring to the transformation of a given vector into a diagonal matrix. With the given

	DOF	50	162	578	2,178	8,450	33,282
BF	Newton iterations	2	3	3	3	4.3	4
	BFGS iterations	4	4.5	5	7.5	12.3	12.6
	Newton comp. time [min]	0.37	2.01	4.72	15.01	138.25	709.21
	BFGS comp. time [min]	0.16	0.69	2.59	9.06	126.08	458.75
	Newton w. backslash solver [min]	1.25	10.36	45.35	247.05	2,080.12	>48h
	BFGS w. backslash solver [min]	0.39	1.86	7.01	252.94	2,738.93	>48h
MLP	Newton iterations	2	3	3	3	4.3	4
	BFGS iterations	4	4.5	5	7.5	12.3	12.6
	Newton comp. time [min]	3e-3	0.01	0.05	0.21	4.90	402.70
	BFGS comp. time [min]	4e-3	0.01	0.04	0.18	4.78	179.71

Table 3.3: Macroscopic convergence behavior for beam frame solver or MLP on the microscopic scale. The table show the number of iterations per load step and the total computation times. Each macroscopic problem is discretized using P1 elements. Data has first been published in [82].

derivative for each layer update and iterative application of the chain rule, it is possible to directly compute the modulus $\frac{\partial \overline{P}}{\partial \overline{F}} = \frac{\partial f_{MLP}(\overline{F})}{\partial \overline{F}}$ from the surrogate model.

For solving the macroscopic boundary value problem that defines the two-dimensional test case, we apply the BFGS method as well as Newton's method with both beam frame solver and surrogate model considered for the microscopic computations. The convergence behavior and the computation times are presented in Table 3.3. For the macroscopic problems used for creating the results shown in this table, the discretization of the macroscopic geometry is based on P1 elements. Additionally, for each macroscopic solver, we use a tolerance of $1e-10$ and stop the computations when the relative norm of the residual becomes smaller than the given tolerance. The relative norm of the macroscopic residual in iteration k is defined as

$$\frac{\|\overline{R}\left(\overline{\boldsymbol{U}}_k\right)\|_2}{\|\overline{R}\left(\overline{\boldsymbol{U}}_0\right)\|_2}$$

with $\overline{\boldsymbol{U}}_k$ given as the macroscopic solution vector in iteration k and $\overline{\boldsymbol{U}}_0$ as the macroscopic initial guess for the solution vector.

In Table 3.3, the first six lines of results refer to the solutions computed with the beam frame solver on the microscopic scale. The macroscopic test case is computed for different problem sizes with regard to the discretizations of the finite element

mesh. The discretization that we considered with the most coarse grid features 50 DOF and the finest discretization features 33,282 DOF. The upper two rows in Table 3.3 refer to the average number of iterations of Newton's or BFGS method required until the stopping criterion is satisfied for one load step. The results show that for both non-linear solvers, the average number of iterations increases with an increasing problem size. Newton's method generally requires significantly less iterations than the BFGS method. This result is to be expected, as it aligns with the theoretically better convergence behavior of Newton's method. With regard to the computation times, we differentiate between the implementation of the beam frame solver that utilizes the given LU decomposition and computes only the forward-backward substitution and the implementation of the beam frame solver which uses MATLAB's backslash operator for computing the deformation vector. Note that we do not differentiate between both approaches for the number of iterations, since both implementations of the beam frame solver yield the same results and therefore, the macroscopic solvers show exactly the same convergence behavior. The previous analyses of the beam frame solver on the microscopic scale have shown that the microscale computations are significantly faster if the LU decomposition is computed explicitly. With regard to the macroscopic computation times, we see again a strong advantage of using the forward and backward substitution for computing the deformations. The respective computation times are significantly smaller than the computation times of the solvers that utilize the backslash solver. This difference in the computation times is apparent for Newton's method as well as for the BFGS method. Although the BFGS method requires more iterations until convergence, for most considered problem sizes, computing the solution with the BFGS solver takes less computation time than computing the solution with Newton's method. Here, the reduced computational effort of approximating the Jacobian yields greater benefit than the lower number of iterations for Newton's method. Note that this behavior is very much problem dependent and does not generally apply to other non-linear problems.

The second section of Table 3.3 refers to multiscale simulations computed with the MLP on the microscopic scale. Similar to the results shown for the beam

frame solver, the average number of iterations per load step are presented as well as the computation times required for computing the macroscopic solutions. Both Newton's and BFGS method are considered as macroscopic solvers for the same discretizations of the macroscopic mesh from 50 DOF to 33,282 DOF. With regard to the average number of iterations, we see exactly the same results as for the beam frame solver on the microscopic scale. These results show that exchanging the beam frame problem on the microscopic scale with the MLP surrogate model does not affect the convergence behavior of the non-linear solver. This behavior is the same for Newton's method as well as for the BFGS method. Regarding the computation times, we see that the multiscale simulations that use the MLP on the microscopic scale compute significantly faster than the simulations based on the beam frame solver. However, the macroscopic results do not show the same extreme difference that the comparison of the microscopic computation times have shown. While the microscopic computations scale linearly with the DOF, the computations on the macroscopic scale show a superlinear scaling behavior. As a result, if no parallelization is considered, the macroscopic computations dominate the computation times for larger problem sizes. Since the macroscopic computations are the same for the beam frame problem or the surrogate model on the microscopic scale, the computation times between both approaches do not differ as greatly as when we solely consider the microscopic computations.

To evaluate the accuracy of the multiscale method with the surrogate model on the microscopic scale, we compute the deviation from the results based on the microscopic beam frame model. For testing, we consider the same geometry and boundary conditions as described above. The levels of refinement for the macroscopic finite element mesh are the same that are also considered for the results of Table 3.3. Additionally, we use different types of elements for discretization, namely bilinear square elements (Q1), linear triangle elements (P1), and quadratic triangle elements (P2). The norm values resulting from the deviation of the MLP based solutions and the solutions based on the beam frame model are presented in Table 3.4. The first two sections of the table refer to the relative norm values with regard to the difference between the deformation vectors. The first section shows the values

DOF		50	162	578	2,178	8,450	33,282
$\frac{\|\overline{\boldsymbol{U}}_{MLP}-\overline{\boldsymbol{U}}_{BF}\|_2}{\|\overline{\boldsymbol{U}}_{BF}\|_2}$	Q1	$1.28e-3$	$1.58e-3$	$1.73e-3$	$1.80e-3$	$1.84e-3$	$1.85e-3$
	P1	$1.24e-3$	$1.56e-3$	$1.72e-3$	$1.80e-3$	$1.83e-3$	$1.85e-3$
	P2	$1.30e-3$	$1.61e-3$	$1.74e-3$	$1.81e-3$	$1.84e-3$	$1.85e-3$
$\frac{\|\overline{\boldsymbol{U}}_{MLP}-\overline{\boldsymbol{U}}_{BF}\|_\infty}{\|\overline{\boldsymbol{U}}_{BF}\|_\infty}$	Q1	$1.38e-3$	$1.52e-3$	$1.56e-3$	$1.58e-3$	$1.58e-3$	$1.58e-3$
	P1	$1.33e-3$	$1.49e-3$	$1.55e-3$	$1.57e-3$	$1.58e-3$	$1.58e-3$
	P2	$1.38e-3$	$1.54e-3$	$1.57e-3$	$1.58e-3$	$1.58e-3$	$1.58e-3$
$\frac{\|\overline{\sigma}_{vM,MLP}-\overline{\sigma}_{vM,BF}\|_2}{\|\overline{\sigma}_{vM,BF}\|_2}$	Q1	$1.19e-3$	$1.26e-3$	$1.37e-3$	$1.43e-3$	$1.46e-3$	$1.46e-3$
	P1	$1.11e-3$	$1.28e-3$	$1.39e-3$	$1.44e-3$	$1.45e-3$	$1.46e-3$
	P2	$1.54e-3$	$1.47e-3$	$1.46e-3$	$1.47e-3$	$1.47e-3$	$1.46e-3$
$\frac{\|\overline{\sigma}_{vM,MLP}-\overline{\sigma}_{vM,BF}\|_\infty}{\|\overline{\sigma}_{vM,BF}\|_\infty}$	Q1	$1.43e-3$	$1.59e-3$	$1.90e-3$	$2.39e-3$	$2.42e-3$	$3.13e-3$
	P1	$1.66e-3$	$1.80e-3$	$1.86e-3$	$2.42e-3$	$2.55e-3$	$3.13e-3$
	P2	$1.82e-3$	$1.81e-3$	$2.45e-3$	$3.16e-3$	$3.41e-3$	$2.88e-3$

Table 3.4: Deviations between the two-dimensional multiscale solutions based on the MLP and based on the beam frame problem. Different ansatz functions and different discretizations are considered for the comparison. Data has first been published in [82].

for the relative L^2-norm and the second section refers to the relative L^∞-norm. For both norm measurements, we see small deviations with each mesh discretization. The relative norm increases slightly with a greater mesh refinement, though the values remain small. Based on the values, there is also no difference regarding the MLPs behavior with different types of elements. Although the training data has been generated on computations based on P1 discretizations, the results for other element types are very similar. This shows that the trained MLP is expected to be applicable to different finite element mesh discretizations without expected increase in the macroscopic error. The last two sections of Table 3.4 refer to the deviations with respect to the von Mises stress measurements. Once again, we consider the relative L^2-norm and the relative L^∞-norm for the evaluation. The deviation behavior for different types of elements and different levels of refinement is very similar to the behavior with regard to the deformation data. Although the L^∞-norm has a slightly stronger increase with an increase in DOF, the norm values are all in the same range.

For the discretization of the test geometry with P1 elements and 33,282 DOF,

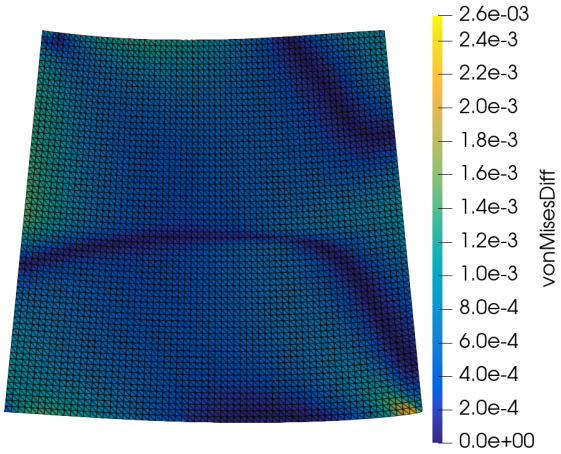


Figure 3.8: Error distribution induced by the surrogate model regarding the test problem for the two-dimensional case. The colors represent the absolute difference between the von Mises stress resulting from the MLP and the von Mises stress resulting from the beam frame model. Image has first been published in [82].

the distribution of the deviation between the MLP based solution and the solution based on the beam frame model is presented in Figure 3.8. Here, the macroscopic solution of the test problem is shown with the colors representing the absolute difference in the von Mises stress at the respective point. With regard to the von Mises stress, the figure shows that the macroscopic error induced by the surrogate model is relatively evenly distributed over the macroscopic domain. In regions with slightly higher differences like the bottom right corner, the error is also relatively low.

To conclude the findings of this section regarding the application of the MLP surrogate model within the FE^2 framework, the results show that the surrogate model on the microscopic scale yields a meaningful reduction of the computational effort for multiscale simulations. For each considered problem size, the MLP based simulations require significantly reduced computation times compared to the multiscale simulations based on the beam frame model. Additionally, the accuracy of the simulations does not deteriorate when the beam frame model is replaced by the surrogate model on the microscopic scale. The macroscopic solvers show exactly the same convergence behavior for both considered microscopic models, while the

deviations between the results of both approaches are minor. The behavior does not change, if different problem sizes are considered. Based on these findings, we consider the application of surrogate models to replace the beam frame problem within the FE^2 framework as beneficial for reducing the computational effort of multiscale simulations.

3.3 Surrogate for Three-Dimensional Microstructure

Similar to the two-dimensional case described in the previous section, we consider an analog machine learning approach for the three-dimensional case. In this section, we describe the development of an MLP that is trained to replicate the relation between the macroscopic deformation gradient and the homogenized first Piola-Kirchhoff stress for a three-dimensional beam frame RVE. In the three-dimensional case, the MLP can be described as the function

$$f_{MLP} : \mathbb{R}^{3\times3} \to \mathbb{R}^{3\times3}, \quad \overline{F} \mapsto f_{MLP}\left(\overline{F}\right) = \overline{P}$$

with the macroscopic deformation gradient $\overline{F} \in \mathbb{R}^{3\times3}$ and the homogenized first Piola-Kirchhoff stress tensor $\overline{P} \in \mathbb{R}^{3\times3}$.

For the three-dimensional case, we consider a beam frame RVE created by the method described in Section 2.2 using sphere packing and Voronoi tessellation. An artificial pore size distribution has been used as basis for creating the RVE. The beam frame structure consists of 1,482 beam elements connected at 1,005 nodes. Based on the linear beam frame problem described in Section 2.2, the deformation and rotation of each node is computed by solving a system of linear equations. With the RVE featuring 1005 nodes, the microscopic problem results in 6,030 degrees of freedom. The described beam frame RVE considered for the three-dimensional case is presented in Figure 3.9.

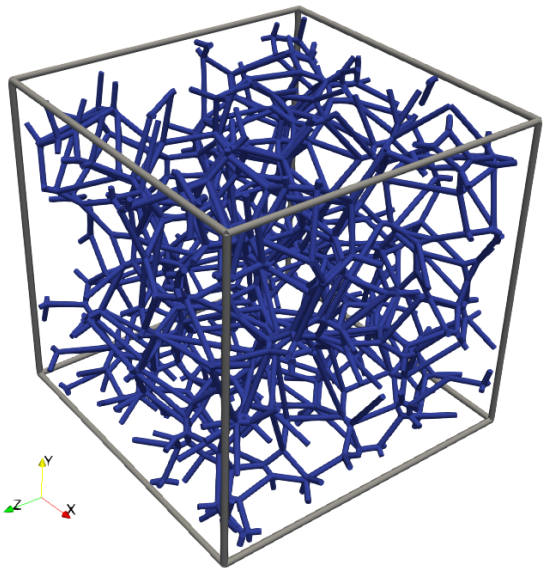


Figure 3.9: Three-dimensional beam frame-structured RVE.

3.3.1 Training Data

Similar to the two-dimensional case, we run several multiscale simulations with different macroscopic setups to create the training data set. In each iteration of the macroscopic solver, the macroscopic deformation gradients and the homogenized stress values resulting from the microscopic problem are stored for every evaluation point of the macroscopic geometry. For the multiscale simulations, we consider a cube geometry discretized with P1 elements resulting in 192 degrees of freedom on the macroscopic scale. In contrast to the generation method for the two-dimensional case, we do not consider various test cases for the boundary conditions. Instead, we apply Dirichlet boundary conditions regarding the deformation at all macroscopic boundary nodes. The deformation values considered for the boundary conditions result from a deformation gradient $\overline{F}_d \in \mathbb{R}^{3\times3}$ which is randomly generated for every multiscale simulation. The deformation for a given macroscopic boundary value $\overline{\boldsymbol{X}} \in \overline{\mathcal{B}}_0$ is then computed by $\overline{\boldsymbol{u}} = \overline{F}_d \cdot \overline{\boldsymbol{X}}$. Manual testing with different generation methods has shown that this training data yields good results regarding

the model's ability to generalize with regard to unknown deformations. Note that the described macroscopic test cases feature only linear transformations. However, this does not restrict the possible applications of the model. Since the NN predicts the local stress strain relationship of the material, the approach is expected to show good results for more complex macroscopic test cases. With the small macroscopic geometry, it is possible to compute a large amount of multiscale simulations with relatively small computational effort, which results in a highly diverse training data set.

Based on the described approach, we create a training data set with a total of about 118,000 data pairs of deformation gradients and corresponding first Piola-Kirchhoff stress tensors. In a preprocessing step, it is ensured that all data pairs are distinct from each other. The validation data set for the NN in the three-dimensional case is created analog to the training data set with different deformations. The validation data results in about 13,000 input and output pairs.

The distribution of the input and output data is presented with the histograms in Figure 3.10. The histograms on the left illustrate the distribution of the components of the deformation gradients. For each of the input features, the mean is relatively close to zero, with small deviations of 0.02 at most. The standard deviation is also very similar for all input features with values of 0.18 or 0.19. The distribution of the components of the target Piola-Kirchhoff stress tensor are presented in the histograms on the right in Figure 3.10. Visually, a slight difference between the main components P_{xx}, P_{yy}, and P_{zz} and the secondary components $P_{xy}, P_{xz}, P_{yx}, P_{yz}, P_{zx}, P_{zy}$ can be observed. This is also attested by the mean values, which are -0.16 for P_{xx}, -0.13 for the components P_{yy} and P_{zz}, and zero for all secondary components. Similar differences are observed regarding the standard deviations of the output data. The secondary components feature standard deviations in the range of 0.14 and 0.16, while the main components show all the same standard deviation of 0.27. The difference between the data distribution of the input and output components is expected to result from the random nature of the microscopic RVE and also of the data generation process.

As in the two-dimensional case, the presented distributions refer to the raw data.

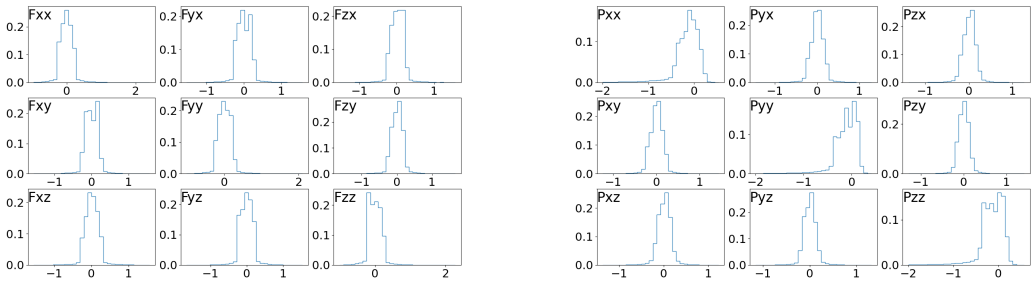


Figure 3.10: Distribution of the data created for the training of the three-dimensional MLP. Histograms on the left show the distribution of the input data and histograms on the right show the distribution of the target output data. Images and data have first been published in [82].

For training the MLP, the data sets are normalized in a processing step to improve the training behavior.

The correlations of the input and output features are presented in Table 3.5. Similar to the correlations in the two-dimensional case, the table shows the Pearson correlation coefficients of various pairs of input and output components. As we pointed out in the description of the correlation regarding the training data of the two-dimensional case, the strongest correlation is observed between the input and output components which refer to the same coordinates. These variables are highlighted in the table. The strongest positive correlation can be observed between the input and output pairs (F_{xx}, P_{xx}), (F_{yy}, P_{yy}), and (F_{zz}, P_{zz}), with close to perfect positive correlation with coefficients greater than 0.9.

3.3.2 Training and Microscopic Behavior

The NN considered in this chapter is intended to replicate the mechanical behavior of the three-dimensional beam frame RVE. With the higher dimensional RVE, the shapes of the input and output layers are unequal to the NN for the two-dimensional case. With an input and output size of $3 \cdot 3 = 9$ and different correlations between the features, we expect a different behavior of the model with respect to the NN architecture. To evaluate the influence of different hyperparameters like layer num-

	$\overline{P}_{xx}$	$\overline{P}_{yx}$	$\overline{P}_{zx}$	$\overline{P}_{xy}$	$\overline{P}_{yy}$	$\overline{P}_{zy}$	$\overline{P}_{xz}$	$\overline{P}_{yz}$	$\overline{P}_{zz}$
$\overline{F}_{xx}$	**0.95**	-0.04	0.03	-0.04	-0.04	-0.04	0.03	-0.04	0.12
$\overline{F}_{yx}$	-0.03	**0.73**	0.03	**0.74**	-0.03	-0.01	0.03	-0.01	-0.05
$\overline{F}_{zx}$	-0.00	0.02	**0.69**	0.02	0.03	-0.06	**0.70**	-0.06	0.02
$\overline{F}_{xy}$	-0.01	**0.74**	-0.05	**0.73**	0.08	-0.04	-0.04	-0.04	-0.07
$\overline{F}_{yy}$	-0.02	0.05	-0.04	0.05	**0.93**	-0.01	-0.04	-0.00	-0.00
$\overline{F}_{zy}$	-0.04	0.02	-0.03	0.02	0.02	**0.69**	-0.03	**0.70**	-0.06
$\overline{F}_{xz}$	0.04	-0.04	**0.69**	-0.04	-0.09	0.04	**0.68**	0.04	0.09
$\overline{F}_{yz}$	0.01	-0.02	0.01	-0.02	-0.05	**0.74**	0.01	**0.73**	-0.03
$\overline{F}_{zz}$	0.12	-0.08	0.08	-0.08	-0.00	-0.09	0.08	-0.09	**0.94**

Table 3.5: Correlation coefficients between the input and output variables for the data set resulting from the three-dimensional RVE. Data has first been published in [82].

ber, layer size, and the activation function used in the hidden layers, we consider a grid search algorithm. As described in the previous section, with the grid search approach, different models are trained for each possible combination of hyperparameters. The performance of each model is evaluated by the resulting loss value after training. The set of hyperparameters which need to be taken into account by the algorithm is previously defined by the user. As we saw good results for the grid search in the two-dimensional case, we consider the same set of hyperparameters also for the model in the three-dimensional case. For training each of the models considered in the grid search, we use a batch size of 256 and Adam optimizer with learning rate scheduler for training. Analog to the two-dimensional case, the learning rate scheduler reduces the learning rate from $1e-3$ to $1e-5$ within 1500 epochs of training. The activation of the output layer is linear for every considered model. With regard to the implementation of training and grid search algorithms, we use the open-source package TensorFlow [1].

The results of the grid search are presented in Table 3.6. Regarding the activation functions, the NNs using *GELU* show the smallest loss values. This behavior is similar to the two-dimensional case. The smallest loss results from the model with three hidden layers, 256 neurons per layer and *GELU* activation function. Because this architecture shows the best results, this model is also used for further

activation	neur./layer	one layer	two layers	three layers
sigmoid	64	1.99e-04	3.31e-05	2.49e-05
	128	4.55e-04	9.18e-06	6.43e-06
	256	5.50e-04	4.37e-06	2.93e-06
	512	1.37e-03	2.00e-06	2.61e-06
tanh	64	5.22e-05	1.88e-05	8.03e-06
	128	5.81e-05	1.52e-05	3.64e-06
	256	1.47e-04	1.56e-05	4.21e-06
	512	4.17e-04	1.16e-05	7.02e-06
gelu	64	1.61e-06	8.66e-07	5.07e-07
	128	3.10e-07	1.24e-07	1.34e-07
	256	1.88e-07	5.38e-08	**3.95e-08**
	512	3.38e-07	3.30e-07	4.24e-08

Table 3.6: Results of the grid search to optimize the model architecture of the MLP for the three-dimensional RVE. Data has first been published in [82].

testing in the three-dimensional case. The final model is trained for 2000 epochs using Adam optimizer with the described learning rate scheduler. The progression of the loss terms resulting from the training of the considered model is presented in Figure 3.11. The graph shows that the loss is reduced significantly during training. In this regard, the validation loss behaves very similar to the training loss. This indicates that the model is not strongly affected by overfitting.

The main goal of this section is the development of a surrogate model to reduce the computational effort for multiscale simulations. Therefore, it is crucial to evaluate the difference in computation time between the developed MLP and the computations of the homogenized Piola-Kirchhoff stress tensor based on the beam frame model. In the case of the beam frame model, the computations which yield the homogenized stress tensor can be described in two main steps. The first step is the computation of the deformation and rotation values for each node within the RVE and in the second step, the first Piola-Kirchhoff stress is integrated over all beam elements to obtain the homogenized stress tensor. For the linear beam frame problem, solving a system of linear equations provides the microscopic stress data. As described for the two-dimensional case in Section 3.2, in each Gaussian quadra-

ture point, the system of linear equations is defined by the same stiffness matrix. Therefore, it can be beneficial to compute the LU decomposition for the stiffness matrix explicitly. This reduces the computations for each Gaussian quadrature point to a forward-backward substitution with the triangular matrices resulting from the LU decomposition and thus also reduces the computation time on the microscopic scale. Alternatively, the microscopic deformations and rotations are directly computed using a linear solver. Both approaches of computing the homogenized stress tensor based on the beam frame model as well as the evaluation of the MLP are implemented in MATLAB [143]. For solving the system of linear equations directly, MATLAB's backslash operator is used. Given these implementations, computing the homogenized stress tensor based on the beam frame model with MATLAB's backslash operator requires about 1,000 times more computation time than evaluating the MLP. Here, the backslash operator requires the majority of the time. For the beam frame problem, computing the homogenized stress tensor from a given LU decomposition and forward-backward substitution is significantly faster. However, this execution requires still about 80 times the computation times compared to the evaluation of the MLP. Additionally, the computational effort of computing the homogenized stress based on the beam frame model increases with an increasing complexity of the RVE, as the complexity of solving the system of linear equations and the homogenization process are both dependent on the number of nodes and beam elements within the RVE. In contrast, the computation time required for the evaluation of the MLP is independent of the size of the RVE.

3.3.3 Application in the Multiscale Framework

Similar to the results described in Section 3.2.3, we evaluate the performance of the multiscale simulation method based on the derived MLP on the microscopic scale. For the discussion of the macroscopic performance, we consider the torsion of a cube geometry as three-dimensional test case. To apply the deformation to the cube, Dirichlet boundary conditions are applied to the opposing upper and lower surfaces of the cube in x-direction. For the points included in the respective planes,

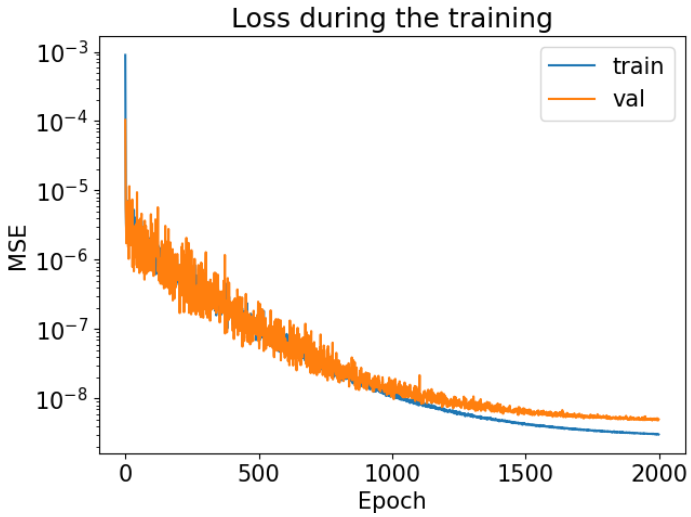


Figure 3.11: Loss curves for the MLP fitted to stress data from the three-dimensional RVE. Blue curve shows the progression of the training loss and orange curve shows the progression of the validation loss. Image and loss data have first been published in [82].

the deformation in x-direction is set to zero and the deformation in y and z-direction is set based on the distance of the regarded point to the center of the surface. As the deformation is higher for the macroscopic points that are further away from the center of the surface, the surfaces are rotated around its center points. With the upper surface rotated in the opposite direction of the lower surface, a torsion deformation is applied to the full geometry. The macroscopic results of this three-dimensional test case are presented in Figure 3.12. Here, the macroscopic geometry is discretized using tetrahedral $P1$ elements and the solution is computed using the beam frame model on the microscopic scale. The colors shown in Figure 3.12 present the distribution of the von Mises stress [109]. This illustration helps to highlight the region where high stress values occur. In the three-dimensional case, the von Mises stress is defined by

$$\overline{\sigma}_{vM} = \sqrt{\overline{\sigma}_{xx}^2 + \overline{\sigma}_{yy}^2 + \overline{\sigma}_{zz}^2 - \overline{\sigma}_{xx} \cdot \overline{\sigma}_{yy} - \overline{\sigma}_{xx} \cdot \overline{\sigma}_{zz} - \overline{\sigma}_{yy} \cdot \overline{\sigma}_{zz} + 3 \cdot \left(\overline{\sigma}_{xy}^2 + \overline{\sigma}_{xz}^2 + \overline{\sigma}_{yz}^2\right)}$$

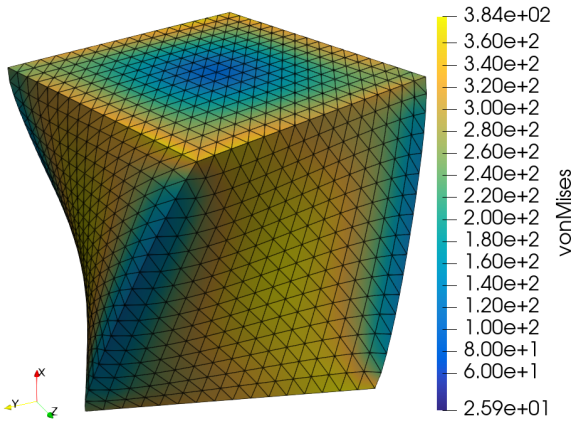


Figure 3.12: Solution for the three-dimensional test case regarding the torsion of a cube geometry. The macroscopic mesh is discretized using P1 elements and the solution is computed based on the beam frame model. Colors in the figure represent the resulting von Mises stress within the geometry.

for a given macroscopic Cauchy stress tensor $\overline{\sigma} \in \mathbb{R}^{3\times 3}$. In Figure 3.12, the highest von Mises stress values are apparent in the region close to the edges and the corners of the cube. These are also the regions in which the nodes feature the greatest deformation values in the final solution.

For evaluating the macroscopic performance of the machine learning-based surrogate model in the three-dimensional case, we compute the multiscale simulations for several levels of refinement based on the beam frame model and based on the MLP on the microscopic scale. Additionally, we consider different ansatz functions for the discretization of the macroscopic finite element mesh. Here, the macroscopic geometry is discretized using trilinear cube elements (Q1), linear tetrahedral elements (P1), and quadratic tetrahedral elements (P2). For each combination of the mesh refinement level and ansatz function, we compute the difference between the solution computed with the beam frame problem on the microscopic scale and the solution computed with the MLP on the microscopic scale. The deviation between the results is evaluated based on the macroscopic deformation vector $\overline{U}$ and the von Mises stress $\overline{\sigma}_{vM}$. The values of the relative L^2 error and the relative L^∞ error are

presented in Table 3.7. Here, variables using NN in the index refer to the solutions computed with the MLP on the microscopic scale and variables using BF in the index refer to the solutions computed based on the beam frame problem on the microscopic scale. The first two sections of Table 3.7 refer to the measurements based with regard to the deformation vectors. For all considered mesh refinements and ansatz functions, the measured error is relatively small. Although the loss values increase slightly with the problem size, each of the error values is relatively minor and below $1e-3$. There is no significant difference regarding the norm values of different ansatz functions. Although the training data for the MLP has been generated using P1 ansatz functions on the macroscopic scale, the MLP approach for different ansatz functions does not show worse behavior overall. The values in the two last sections of Table 3.7 refer to the von Mises stress measurements. The behavior of the relative error is very similar to the behavior of the norm values based on the deformation vectors. Generally, we see a slight increase of the error with finer finite element meshes. However, the relative error remains small for all considered discretizations.

To conclude the findings of this section, the MLP model shows decent results as a replacement for the beam frame problem on the microscopic scale. The FE^2 multiscale simulations based on the surrogate model yield the same convergence behavior with a reduced computational effort. Additionally, the accuracy of the computed results is very close to the accuracy of the computations based on the original beam frame model. The quality of the simulation results does not change significantly, as different ansatz functions or mesh refinements are considered. This shows the versatility of the MLP approach with regard to different macroscopic problems.

DOF		108	525	3,159
$\frac{\|\overline{\boldsymbol{U}}_{MLP}-\overline{\boldsymbol{U}}_{BF}\|_2}{\|\overline{\boldsymbol{U}}_{BF}\|_2}$	Q1	$2.36e-4$	$3.56e-4$	$4.13e-4$
	P1	$1.72e-4$	$2.91e-4$	$3.77e-4$
	P2	$1.65e-4$	$3.52e-4$	$4.22e-4$
$\frac{\|\overline{\boldsymbol{U}}_{MLP}-\overline{\boldsymbol{U}}_{BF}\|_\infty}{\|\overline{\boldsymbol{U}}_{BF}\|_\infty}$	Q1	$4.95e-4$	$6.45e-4$	$7.03e-4$
	P1	$3.84e-4$	$5.68e-4$	$6.26e-4$
	P2	$3.20e-4$	$6.83e-4$	$7.27e-4$
$\frac{\|\bar{\sigma}_{vM,MLP}-\bar{\sigma}_{vM,BF}\|_2}{\|\bar{\sigma}_{vM,BF}\|_2}$	Q1	$2.98e-4$	$4.56e-4$	$5.72e-4$
	P1	$2.32e-4$	$3.59e-4$	$4.80e-4$
	P2	$2.44e-4$	$3.40e-4$	$4.83e-4$
$\frac{\|\bar{\sigma}_{vM,MLP}-\bar{\sigma}_{vM,BF}\|_\infty}{\|\bar{\sigma}_{vM,BF}\|_\infty}$	Q1	$6.05e-4$	$1.11e-3$	$1.66e-3$
	P1	$4.33e-4$	$9.07e-4$	$1.56e-4$
	P2	$3.85e-4$	$6.24e-4$	$1.79e-4$

Table 3.7: Deviations between the three-dimensional multiscale solutions based on the MLP and based on the beam frame problem. Different ansatz functions and different discretizations are considered for the comparison. Data has first been published in [82].

3.4 Efficient Homogenization of Cauchy Stress Tensor

As described in Section 2.3, the momentum balance equation can be described in terms of the first Piola-Kirchhoff stress tensor or the Cauchy stress tensor. Although it is common to use the Piola-Kirchhoff stress tensor with regard to finite element simulation, we also discuss the combination of the Cauchy stress tensor and machine learning-based surrogate models.

As the results in Section 3.2 and Section 3.3 have shown, the relationship between the macroscopic deformation gradient and the homogenized first Piola-Kirchhoff stress tensor can be reproduced by a trained MLP. A similar approach could also be considered for predicting the relationship between the deformation gradient and the Cauchy stress tensor with an MLP. Due to its symmetry the Cauchy stress tensor features only four mutually different entries in the two-dimensional case and six mutually different entries in the three-dimensional case. Therefore, this would

result in a smaller output layer than in the case of the MLP trained to predict the first Piola-Kirchhoff stress tensor.

Regarding the Cauchy stress tensor, in the case of the linear beam frame problem, all computation steps can be described as linear operations. This includes solving the linear problem and computing homogenization of the stress components. If we consider the linear beam frame problem in the three-dimensional space with n_B beam elements and n_N nodes as described by Equation (2.9) in Section 2.2, the vector containing the deformation and rotation values for each node can be computed from a simple matrix multiplication

$$\boldsymbol{U} = \boldsymbol{K}^{-1} \cdot \boldsymbol{RHS} \cdot \mathrm{vec}(\overline{F}) \tag{3.2}$$

with the inverse stiffness matrix $\boldsymbol{K}^{-1} \in \mathbb{R}^{18 \cdot n_N \times 18 \cdot n_N}$ defined from the given RVE, the matrix $\boldsymbol{RHS} \in \mathbb{R}^{18 \cdot n_N \times 9}$ defined to compute the right-hand side of Equation (2.9), and $\mathrm{vec}(\overline{F}) \in \mathbb{R}^9$ the column vector that results from vectorization of the deformation gradient matrix $\overline{F} \in \mathbb{R}^{3\times3}$.

As the homogenization approach with respect to the Cauchy stress tensor described in Section 2.2.4 shows, for each beam element, the computations yielding the integral of the Cauchy stress tensor over the volume of the beam are linear. Therefore, it exists a matrix $\boldsymbol{C} \in \mathbb{R}^{6 \times 18 \cdot n_N}$ that defines the linear relation between the global deformation vector $\boldsymbol{U}$ and the homogenized Cauchy stress tensor σ by the equation

$$\mathrm{vec}(\sigma) = \boldsymbol{C} \cdot \boldsymbol{U}. \tag{3.3}$$

Here, the vector $\mathrm{vec}(\sigma) \in \mathbb{R}^6$ contains the mutually different entries of the symmetric Cauchy stress tensor $\sigma \in \mathbb{R}^{3\times3}$. Based on Equation (3.2) and Equation (3.3), we can derive a matrix combining the described multiplications

$$\boldsymbol{M} = \boldsymbol{C} \cdot \boldsymbol{K}^{-1} \cdot \boldsymbol{RHS}$$

and with this matrix $\boldsymbol{M} \in \mathbb{R}^{6\times9}$, the linear relationship of the macroscopic defor-

mation gradient and the homogenized Cauchy stress tensor is given by the equation

$$\mathrm{vec}(\sigma) = \boldsymbol{M} \cdot \mathrm{vec}(\overline{F}).$$

Regarding the computation of the matrix $\boldsymbol{M}$, it is not recommended to explicitly derive the inverse of the stiffness matrix $\boldsymbol{K}^{-1}$, as it is numerically more stable to use a linear solver for this computation.

Note that although we derived the linear relation only for the three-dimensional Cauchy stress tensor, similar equations can also be described in the two-dimensional case. Since the symmetric Cauchy stress tensor has only three distinct components in the two-dimensional case, the respective matrix $\boldsymbol{M} \in \mathbb{R}^{3\times4}$ is even smaller.

To conclude the findings with regard to the linear relationship between the macroscopic deformation gradient and the Cauchy stress tensor, as all matrices required for the explicit computation of $\boldsymbol{M}$ are known, the solving of the linear beam frame model and the homogenization process to compute the homogenized Cauchy stress tensor can be reduced to a small matrix multiplication. The dimensions of the matrix $\boldsymbol{M}$ are independent of the number of beam elements and the number of nodes of the considered beam frame model. This means that for every possible RVE, the computation of the homogenized Cauchy stress tensor requires the multiplication of the deformation gradient with the matrix with the dimensions $\boldsymbol{M} \in \mathbb{R}^{6\times9}$. With regard to the computation of the homogenized Cauchy stress in a multiscale simulation, the computational effort required for obtaining the inverse of the stiffness matrix and the other required matrices can be neglected, as these matrices need only to be computed once for a given RVE.

As computing the homogenized Cauchy stress tensor requires only the computation of a small matrix multiplication, it is not expected that the application of a machine learning-based surrogate model could offer any opportunity of reducing the computation time. Even an MLP model without any hidden layers would require at least a matrix multiplication of the same scale to predict the homogenized components of the Cauchy stress tensor based on the given macroscopic deformation gradient. Although it is expected that developing an MLP that is capable of

predicting the homogenized Cauchy stress tensor would be possible, it would not yield any benefit with regard to the multiscale simulations. Therefore, we omit the analysis of surrogate models with regard to the computation of the Cauchy stress tensor and consider the matrix multiplication described above as the fastest method of computing the stress components.

Chapter 4

Advanced Architectures for Material-Dependent Homogenization

The models described in Chapter 3 have in common that the input is always considered to be the deformation gradient. With the deformation gradient as the sole input, the MLPs can only be trained to replicate the mechanical properties of a single RVE. In the scope of this chapter, we consider different approaches which use additional input features describing the structural properties of the RVE. With these approaches, we investigate the possibilities of training a single NN that is capable of predicting the mechanical properties of various RVEs. Achieving this goal eliminates the need of retraining a model if a different material is considered for simulation that has not been included in previous training.

For taking into account the RVE structure, we consider three different approaches, which are described in this chapter. Before we introduce the first surrogate model, Section 4.1 introduces the approach of generating a training data set that covers stress data for multiple RVEs. We analyze the properties of the generated RVE data, as the resulting data sets are relevant for all following sections in this chapter. Section 4.2 deals with the surrogate approach based on the MLP architecture. Here, the used NN architecture is very similar to the models presented in Chapter 3. In addition to the deformation gradient, we incorporate material in-

put features into the training of the MLP. In Section 4.3, we introduce the Mixture of Experts (MoE) model architecture as a surrogate model. This model combines the output of multiple expert NNs in a weighted sum to predict the homogenized stress tensor for the FE^2 method. Here, the composition of the weighted sum is determined by a separate gating NN. For this surrogate approach we evaluate the performance of multiple MoE models with various numbers of experts and compare different training approaches. Additionally, for validation of the surrogate model, we compute the macroscopic deviation from the multiscale simulations that are based on the beam frame model. The third surrogate approach in Section 4.4 covers graph neural networks (GNNs) which process the graph structure of the frame-structured RVE. Here, we introduce a MoE approach that utilizes the GNN architecture for the definition of the gating network. We evaluate the performance of this specific gating network and discuss the application of GNN models for homogenization.

For implementing and evaluating the different surrogate approaches including the material input features, we focus on the three-dimensional beam frame model as reference. In Chapter 3, the results based on the two-dimensional and the three-dimensional beam frame problems show a very similar behavior of both surrogate models. Based on these findings, we expect that the development of a surrogate model for the beam frame problem with consideration of RVE based input features will also allow conclusions to be drawn about the two-dimensional case. As the three-dimensional behavior is more general and also more realistic with regard to the real-world behavior of open-porous materials, we consider the analyses of this approach as preferred for our study.

4.1 Training Data for Multiple Microstructures

As the models considered in this chapter are expected to predict the stresses for various open-porous materials, the data set used for training the models needs to contain data resulting from multiple RVEs. For each considered beam frame model, the computation of homogenized stress values is required based on the respective

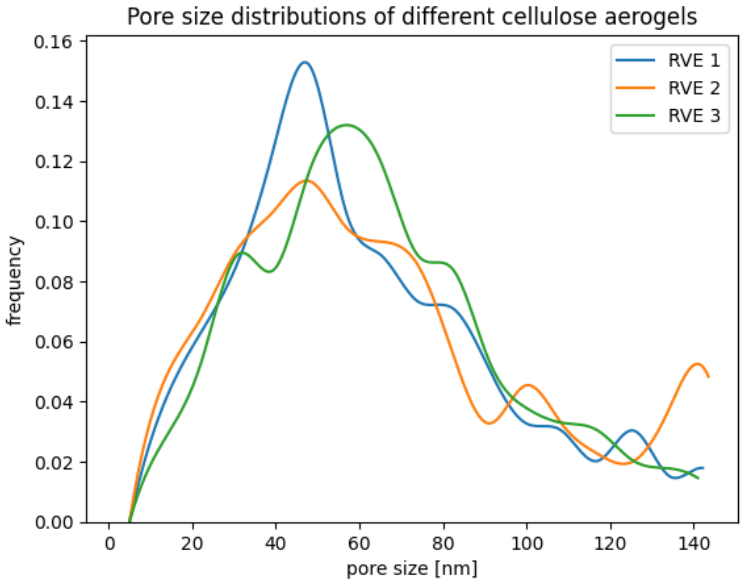


Figure 4.1: Pore size distributions for three random RVEs from the training data set. The value on the x-axis refers to the pore diameter and the y-axis shows the respective frequency for a given diameter.

mechanical behavior of the RVE. To obtain this data set, we consider 250 different RVEs constructed as beam frame models. The creation method for each of the RVEs is applied based on sphere packing and Voronoi tessellation as described in Section 2.2. As initial input of the sphere packing algorithm, we consider 400 pore diameters in every case. For each of the RVEs, we use a different pore size distribution and a different porous fraction as a basis for creating the RVE. The pore size distribution as well as the porous fraction are selected randomly but the range of possible values is determined to resemble the shape of experimental data obtained from the analysis of cellulose aerogels. The graphs in Figure 4.1 illustrate the pore size distributions for three randomly created RVEs. The curves show the density functions for each considered material. For illustration purposes, the curves are slightly smoothed to emphasize the differences between the pore size distributions for the reader. As the shape of the density functions is based on experimental data, the differences between the distributions are rather subtle. The porous fractions considered for creating the RVEs are equally distributed in the range of [0.82, 0.94].

For each of the considered RVEs, we generate an individual data set of deformation gradients and the corresponding stress tensors. Similar to the training data generated for the MLPs in Chapter 3, we run multiscale simulations based on the linear beam frame solver and store the deformation gradients and the resulting homogenized stress values for the data sets. For each of the RVEs, we consider a data set containing a total of 100,000 pairs of deformation gradients and first Piola-Kirchhoff stress tensors. With 250 RVEs, this results in a combined data set of 25,000,000 data points.

In addition to deformation and stress data, we compute a number of features describing the structure of the RVE. For a given open-porous material, the features are expected to pose as a parameterized representation of the material's microstructure. To achieve this, the set of values includes experimental data such as information about the material's pore size distribution and porous fraction but also features regarding the beam frame structure of the corresponding RVE. We denote the set of values which yield the parametrized representation of the material's microstructure as $I_{RVE} \in \mathbb{R}^{10}$. This vector is defined from the set of characteristics given by:

- Mean of pore size distribution
- Median of pore size distribution
- Standard deviation of pore size distribution
- Skewness of pore size distribution
- Kurtosis of pore size distribution
- Normalized orientation of the beam elements in x-direction
- Normalized orientation of the beam elements in y-direction
- Normalized orientation of the beam elements in z-direction
- Porous fraction
- Mean degree of the nodes within the RVE

The normalized orientations of the beam elements result from the composition of the RVE and the values result from the equation

$$\begin{pmatrix} O_x \\ O_y \\ O_z \end{pmatrix} = \sum_{e=(\boldsymbol{v}_i,\boldsymbol{v}_j)\in\mathcal{E}} (r_e^2 \cdot \pi) \cdot |\boldsymbol{v}_j - \boldsymbol{v}_i|.$$

Here, O_x, O_y, and O_z describe the cumulative orientations of the beam frame for the respective coordinate. To obtain the normalized values, we scale O_x, O_y, and O_z in way that the mean over the three values equals one. This is scaling is crucial to obtain comparable values for different RVEs. In mathematical description of the orientation values, $\mathcal{E}$ refers to the set of beam elements within the RVE, $e = (\boldsymbol{v}_i, \boldsymbol{v}_j)$ refers to a beam element with the respective start node $\boldsymbol{v}_i \in \mathbb{R}^3$ and end node $\boldsymbol{v}_j \in \mathbb{R}^3$, and $r_e > 0$ refers to the radius of the beam element e.

For each of the 250 beam frame structures, we compute and store the material parameters as defined above. This means that for each RVE, we obtain a vector $I_{RVE} \in \mathbb{R}^{10}$ that can be used as additional input to a considered surrogate model.

In addition to the 250 RVEs used for generating the training data, we consider 50 different RVEs created in the same manner as the first 250 RVEs. For creating the smaller set of 50 RVEs, we use the same creation method resembling the experimental data of cellulose aerogels. However, due to the random selection of the material parameters, for each of the 50 RVEs, the configuration of the pore size distributions and the porous fractions is different from every other RVE given in the set of 250. For each of the beam frame models resulting from the 50 RVEs, we create an individual data set of 100,000 deformation gradients and stress tensors. We use this data for validation of the surrogate model, as the data is not used for training the model. By computing the model's loss with respect to the data created by the 50 RVEs, we are able to evaluate the performance of the machine learning model with regard to the simulation of unknown materials. Based on this purpose of the data, we refer to the 50 RVEs as validation RVEs. In addition to the deformation and stress data sets, we also compute a vector containing the material

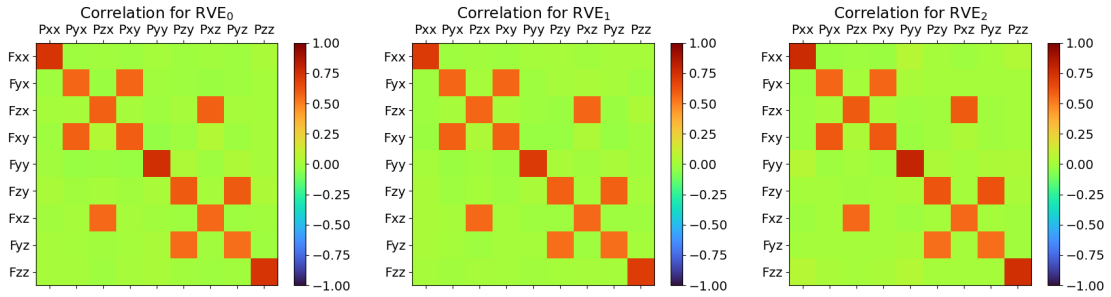


Figure 4.2: Correlation matrices for the data based on the first three training RVEs. Each tile within a given graph represents a single entry of the correlation matrix for the respective data. The matrices cover the correlation between the components of the deformation gradient and the homogenized first Piola-Kirchhoff stress tensor.

parameters for each validation RVE. As 100,000 deformation and stress data points are computed for each of 50 validation RVEs, this results in a total of 5,000,000 data points for the validation data set.

This section does not provide a detailed discussion with regard to the distribution of the data set resulting from the 250 training RVEs or of the data set resulting from the 50 validation RVEs. The properties of the deformation and stress data resulting from the mechanical behavior of a single RVE are similar to the data properties presented in Section 3.3. Especially, the correlation between the components of the deformation gradient and the components of the first Piola-Kirchhoff stress tensor are very similar. As an example, the values of the correlation matrices resulting from the deformation and stress data computed for the first three training RVEs are presented in Figure 4.2. For all three images, each presented tile refers to an entry of the respective correlation matrix. The color reveals the value of the entry while red indicates values close to one and green indicates values close to zero. As all three graphs are visually almost identical, the results indicate that the correlations are essentially equal. Although the correlations between deformation gradient and stress tensor are very similar, the range in which the resulting stress data is distributed can differ significantly. Depending on the porous fraction and the configuration of the beam elements within the RVE, we see significant differ-

ences with regard to the magnitude of the resulting stress data. To quantify the magnitude of the stress data for a given RVE, we compute the homogenized stress tensor for the compression of the RVE in each of the three dimensions and compute the absolute mean over the stress components. In practice, this means that we consider the deformation gradients

$$\overline{F}_1 = \begin{pmatrix} 0.5 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad \overline{F}_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \text{ and } \quad \overline{F}_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.5 \end{pmatrix}$$

and compute the respective first Piola-Kirchhoff stress tensors $\overline{P}_1$, $\overline{P}_2$, and $\overline{P}_3 \in \mathbb{R}^{3\times3}$ based on these given deformation gradients. The value 0.5 in one of the diagonal entries of the deformation gradient corresponds to a reduction of the RVE's volume by the factor 0.5 in the respective dimension. The value describing the magnitude of the stress values is then defined as the average of the components resulting from the sum of the considered stress tensors

$$\Delta_{RVE} = \frac{1}{9} \cdot \sum_{i=1}^{3} \sum_{j=1}^{3} \left(|P_1| + |P_2| + |P_3| \right)_{i,j} .$$

Here, we denote $\Delta_{RVE} \in \mathbb{R}$ as the magnitude of stress values with respect to a given RVE. Note that, the magnitude value is defined to quantify the differences between the scales of stress tensors for different RVEs. As each data set of stress tensors with regard to a single RVE is randomly generated, the range of the data generally does not align with the computed magnitude value.

To emphasize the differences in magnitude between the data sets for different RVEs, we use global scaling values for normalizing the training and validation data sets for training of the surrogate models in this chapter. This means that the individual data sets for each RVE are not normalized separately, but we use the same constants for normalizing the combined data sets uniformly. This approach is expected to help with the models differentiating between the stress components for different RVEs.

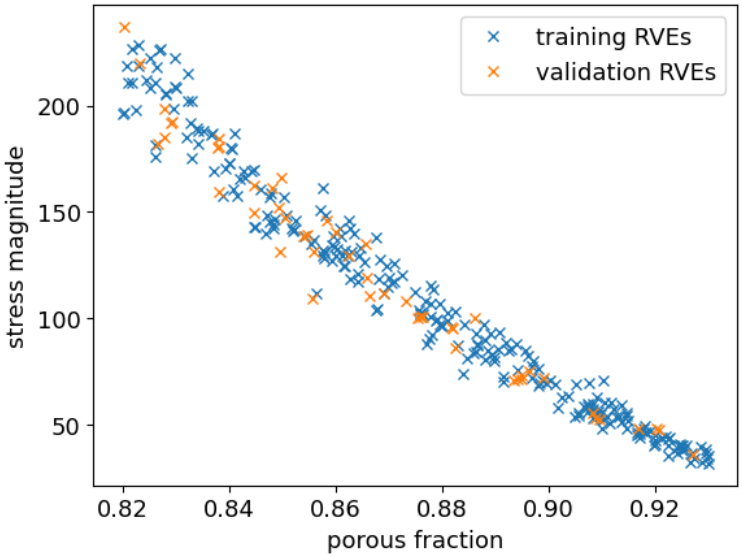


Figure 4.3: Relationship between the porous fraction and the magnitude of the stress values. Blue marks refer to the RVEs used to generate the training data set and orange marks refer to the validation RVEs.

As described in Section 2.2, the porous fraction is the main factor that influences the diameter of each beam element. Additionally, an RVE with thicker beam elements is also expected to produce higher absolute values of the homogenized stress components. As a result, we see a strong correlation between the porous fraction of the material and the magnitude of the stress values as described above. This relation of both variables is presented in Figure 4.3. Each blue mark shown in the image represents a single RVE used to generate the training data set and each orange mark refers to one of 50 validation RVEs. The marks shown in the figure indicate a negative correlation between the porous fraction of a given RVE and the resulting magnitude stress value. However, the relation is not linear but seems to follow an exponential curve. The relationship appears to be the same for the training RVEs as for the validation RVEs. Over the full set of 300 RVEs, we see a Pearson correlation coefficient between the porous fraction and the magnitude of the stress values of -0.9797. As this value is close to minus one, the values are almost perfectly negatively correlated.

4.2 Generalized Multi-Layer Perceptron

As one of the most direct approaches to include material parameters in the computation of the homogenized stress tensor, we consider the MLP architecture which is covered in Chapter 3. In the previous discussion with regard to the application of the MLP to predict the homogenized stress tensor, we considered the macroscopic deformation gradient as a single input of the model. In this section, we expand the input by including a number of material parameters. The general structure of the model architecture is the same as it is described in Section 3.1. As input to the model, we consider a combination of the macroscopic deformation gradient $\overline{F}$ and the parameterized representation of the RVE given by I_{RVE}. As we focus on the three-dimensional case for the development of the surrogate model, this results in the input parameters given by

$$\left(\overline{F},\, I_{RVE}\right) \in \mathbb{R}^{3\times 3} \times \mathbb{R}^{10}$$

The output layer of the MLP contains the components of the first Piola-Kirchhoff stress tensor. Therefore, the layer features nine neurons. Formally, the computation of the MLP can be described by the function

$$f_{MLP} : \mathbb{R}^{3\times 3} \times \mathbb{R}^{10} \to \mathbb{R}^{9}, \quad \left(\overline{F},\, I_{RVE}\right) \mapsto f_{MLP}\left(\overline{F},\, I_{RVE}\right) = \overline{P}.$$

4.2.1 Inclusion of Material Input Features

Proper optimization of the weight values of the MLP, requires a training data set that includes material features. These input features are processed in addition to the deformation gradient to compute the target stress values. To obtain the required features, we use the data set based on 250 mutually different RVEs that is described in Section 4.1 for creating the training data. Since training on the entire data set of 25,000,000 data points would require a large amount of computing time, we take a randomly selected portion of 1,800,000 data points from the full data set and use this reduced portion as training data for the MLP. Furthermore, manual

testing with different sizes of the used training data set has shown that a further increase in the number of data points used for training does not result in significant improvements with regard to the model's performance during training. In some cases, a truly large data set can also amplify the overfitting behavior of the model.

In addition to the training data set, we consider two different sets of validation data. One validation data set contains 200,000 data points randomly selected from the same full data set as the training data. It is ensured that the data points in the validation data set are different from the values contained in the training data set so that the union of both sets is empty. Evaluating the loss of the model with regard to this validation data set indicates how well the MLP performs with regard to unknown deformation data. The same material parameters are considered that are also included in the training. However, the deformations have not been included in the training. Based on the loss values with regard to this validation data, we can evaluate the model's overfitting behavior with respect to the deformation gradient input. Based on this purpose of the validation data, we refer to this data set as val-F data, as F is also the notation for the deformation gradient.

The second data set used for validation of the MLP model is created based on the 50 validation RVEs described in Section 4.1. As the full data set includes 5,000,000 data points, we reduce the number of points used for validation of the model to 400,000. This reduction of the data is supposed to reduce the computational effort of training and validation. The data points are randomly selected from the full data set of 5,000,000 points. As this validation data set contains material parameters and deformation gradients different from the data used for training the model, it can be used to evaluate the surrogate model's performance with regard to unknown materials. It also indicates whether the behavior during training is affected by overfitting. Since this validation data set is used to evaluate the model's performance with regard to unknown RVEs, we also refer to the data set as val-RVE data.

For defining the architecture of the considered MLP model to be trained on the data of multiple beam frame problems, we use the results of Section 3.2 and Section 3.3 as a basis. We use three hidden layers for the model. Each of the

layers features 512 neurons and $GELU$ activation. This structure is very similar to the architectures presented in Chapter 3. The results presented in the previous chapter have shown that $GELU$ activation function provides the best results. As the training data features significantly more data and the relationship is expected to be more complex due to an increased number of input features, we use an increased number of neurons per layer in comparison to the models described in Chapter 3.

In addition to the described MLP model that includes the material features, we also discuss the performance of a model which is based on the same architecture but using solely the deformation gradient as input. We use the data set described above resulting from the computations of multiple RVEs to train the MLP. It is expected that this approach does not achieve low loss values regarding the full data set, since the model does not use any input data to differentiate between different RVEs. Although the correlation between deformation gradient and resulting stress components is very similar for different RVEs, the magnitude of the stress values can differ significantly for different beam frame structures. However, we use the results of this model to provide a meaningful basis for a comparison with different approaches that include the material parameters. As this given MLP uses only the deformation gradient as input, a comparison to different models can highlight the benefits of including the material features in the input vector.

4.2.2 Training Results

For both MLP models with and without inclusion of material features, the weights of the models are optimized based on the mean squared error (MSE) loss term. Adam optimizer is used to minimize the respective loss values. To reduce volatile behavior of the loss values over the scope of the optimization, we use a learning rate scheduler which reduces the learning rate from $1e-3$ to $1e-5$ over the number of training epochs. The training data set is partitioned with a batch size of 512. The definition and training of the model is implemented using the TensorFlow framework [1].

The graphs presented in Figure 4.4 show the progression of the loss terms with

regard to the described training data and the validation data sets. The image on the left illustrates the training behavior of the MLP that uses only the deformation gradient as input. The blue graph refers to the loss based on the training data set, the orange curve refers to the loss based on the validation data set that is computed from the same RVEs as the training data but considers different deformations, and the green curve refers to the loss based on the validation data that is created from the validation RVEs. The results show that the training loss is reduced over the training process of 500 epochs. However, this reduction of the MSE loss is only minor, as the scale of the y-axis indicates. With regard to both validation data sets, the model performs significantly worse, since both loss curves increase over the training process. This is a typical behavior of the validation loss that indicates overfitting. This means that the model adjusts its weights to fit the training data, though it does not capture any general relationship between input and output data that also fits the validation data. This approach would not be applicable for the simulation of different materials, as it does not yield proper predictions for the resulting stress components.

In Figure 4.4, the image on the right shows the progression of the loss for the MLP that uses the material data as input. The blue curve referring to the training loss shows a more significant reduction of the MSE compared to the cases of the MLP that uses only the deformation gradient as input. This is expected, as the MLP is able to use the additional input features to differentiate between the stress data of different RVEs. For the first 400 training epochs, the loss regarding the val-F data set behaves very similar to the training loss. Visually, both curves show the same descent behavior. Although the curve regarding the training loss is relatively flat between epoch 200 and 400, it declines even further in the last 100 epochs. In these epochs of the training, the behavior of the training loss and the loss regarding the val-F data differ, as the validation loss is not declining as significant as the training loss. Based on this behavior, it is also expected that the loss regarding the val-F data would not reduce with any further training beyond 500 epochs. The green curve in the image represents the loss regarding the val-RVE data. This validation data set is created based on 50 validation RVEs that are different from the RVEs used for

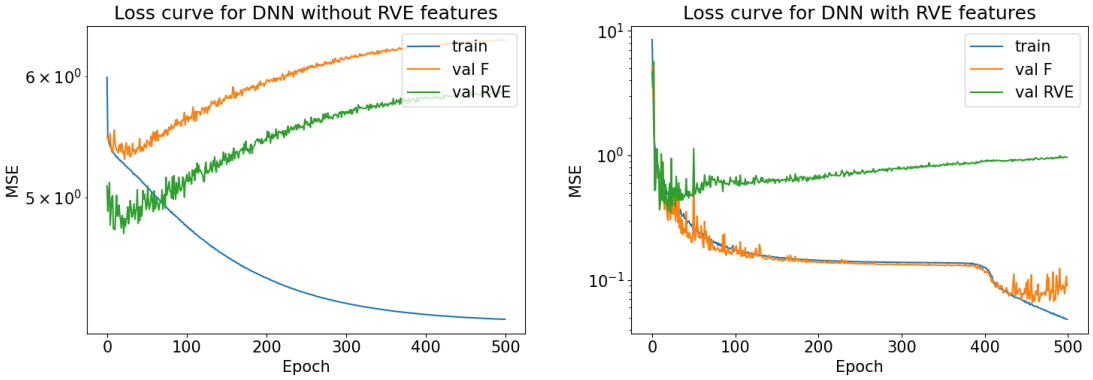


Figure 4.4: Loss curves for the MLP training on data resulting from multiple RVEs. Left image shows the training behavior, if the RVE features are excluded and image on the right shows the training behavior, if the RVE features are included as input.

creating the training data. After a steep reduction in the first few training epochs, the val-RVE loss curve is mainly rising over the course of the remaining training epochs. This progression of the loss curve indicates an overfitting behavior with regard to the RVE features. It shows that the MLP is able to differentiate between the RVEs covered within the training data set, though it does not capture the general relationship between the material features and the resulting stress values.

With regard to the MLP model architecture to be trained on the stress data resulting from multiple RVEs, the results in this section show that the considered MLP is capable of capturing the relationship between the deformation gradient and the resulting stress components for multiple RVEs. As the val-F loss is also significantly reduced, we see a similar behavior with regard to different deformations as seen for the model described in Section 3.2 and Section 3.3. For the validation data set created from the validation RVEs, the loss does not decline as significantly as for the training and val-F data sets. This implies that the MLP is not able to predict the mechanical behavior of RVEs that are not included in the training data set.

Since the MLP does not yield proper approximation of the stress data for the validation RVEs, it is expected that an application of the MLP in the FE^2 method would yield different mechanical behavior on the microscopic scale compared to

the beam frame model for any validation RVEs. As we see strong deviation with regard to the microscopic computations, the results of multiscale simulations with the MLP on the microscopic scale are also expected to include large approximation errors. Since the microscopic results show that the approach is not suited for the prediction of stress data for unknown materials, we do not consider any macroscopic testing for the model. It is not expected that the macroscopic results would yield any further insight with regard to the MLP's performance.

4.3 Mixture of Experts Model for Homogenization

As the name of the Mixture of Experts (MoE) machine learning approach [74, 79, 35, 40] suggests, the key feature of the MoE model is the combination of multiple expert NNs. Each of the expert NN produces its individual output and all expert outputs are aggregated in a weighted sum. With the weights of this sum determined by a separate gating NN, depending on the given input, the MoE model is able to vary the contribution of some of the expert NNs to the global output.

With a rising interest in large language models (LLMs) in recent years, the MoE approach attracted significant attention regarding the development of new machine learning models [21]. Some of the recent LLM developments such as the Mixtral of Experts model [77] or DeepSeek-V3 [28] use the MoE framework to reduce the computational effort of the evaluation. Additionally, the MoE model has shown to be beneficial in some applications regarding scientific machine learning. In the context of physical simulations, the MoE approach is used for predicting complex input-output correlations [25, 112, 10, 141]. With multiple expert NNs, the method allows experts to specialize on various aspects of the physical problem, for example, on different subdomains of the solution. Additionally, for solving PDEs, the MoE approach can be combined with physics loss terms [25, 10, 141]. This method is also referred to as GatedPINN and has been introduced in [141].

As the results in Chapter 3 show, the MLP model is well-suited for replicating the relationship between the macroscopic deformation gradient and the resulting

homogenized stress tensor for a given RVE. Based on these results, we consider the Mixture of Experts (MoE) model architecture for developing a surrogate model which can be trained to predict the mechanical behavior of various materials. In this context, we consider multiple expert NNs, each of which is trained on the stress data computed for a single RVE modeled as a beam frame. Thus, each expert model is associated with a given open-porous material and is trained to reproduce its mechanical behavior. The training of the expert NNs follows the same approach which is described in Chapter 3. We consider a representation of the open-porous nanostructure as input for the gating NN. Based on this information, the gating NN is trained to compute a fitting weight vector which is required for the weighted sum over the expert outputs. This means that the stress tensor resulting from the MoE model is given as the aggregation of the expert outputs based on the weights given by the gating network. We consider different MoE models for the application as surrogate model in the FE^2. To compare the given approaches, we evaluate each model's performance based on the microscopic and macroscopic deviation from the baseline beam frame model.

4.3.1 Mixture of Experts as Machine Learning Model

The model architecture described by the MoE approach differs primarily from MLPs as described in Section 3.1 in that it can activate or deactivate individual parts of the architecture depending on the input. The MoE machine learning model features expert NNs, each of which generates an output of the same shape. The resulting output of the MoE model is computed as the weighted sum over the outputs of the expert NNs with the weights determined by an additional gating NN. Based on a given input, the gating NN generates a probability distribution with the length equal to the number of experts given in the MoE model. Each value given by the output of the gating NN defines one weight in the weighted sum and determines the contribution of the corresponding expert NN to the final output. As it is common for the MoE model that some of the weights are zero or very close to zero, the gating NN can determine that the corresponding models do not contribute to the global

output at all. By evaluating only the expert models with meaningful contribution to the weighted sum, the computational effort for computing the output of the MoE can be significantly reduced.

We consider $n \in \mathbb{N}$ individual expert NNs that are defined by the functions $f_0, \ldots, f_{n-1}$ with

$$f_i : D_{expert} \to D_{output}, \quad x_{expert} \mapsto f_i(x_{expert}) = y_i$$

for $i = 0, \ldots, n-1$. Here, D_{expert} and D_{output} define arbitrary sets containing the input and output data for the expert NNs. While each of the expert models can use a different model architecture, the shape of the input and the produced output must match. As the outputs of all expert NNs are computed individually, the global output of the MoE model $y \in D_{output}$ is given by the weighted sum over expert outputs

$$y = \sum_{i=0}^{n-1} w_i \cdot y_i.$$

The weights $w \in \mathbb{R}^n$ are given as output of the gating model. The gating NN can be characterized by the function

$$f_{gating} : D_{gating} \to R^n, \quad x_{gating} \mapsto f_{gating}(x_{gating}) = w.$$

Here, D_{gating} refers to an arbitrary set containing the input data of the gating model. Though, in a number of applications, the input of the gating model is considered to be the same as the input of the expert models. The weights $w \in \mathbb{R}^n$ which the gating model produces as output can be regarded as a probability distribution which assigns a probability value to each expert model. Therefore, the gating model is basically trained to solve a classification problem. As the application of NNs for solving classification problems is well-studied, the *softmax* activation function has been established as the preferred choice for the activation function in the output layer. This activation function, which has already been used for classification applications in the early 1990s [16], produces an output that sums up

to one. For an arbitrary vector $x \in \mathbb{R}^n$, the application of the *softmax* function to the vector x results in the vector $\mathrm{softmax}(x) \in \mathbb{R}^n$. The computation of this vector is described for the i-th vector component by the expression

$$\mathrm{softmax}(x)_i = \frac{\exp(x_i)}{\sum_{j=0}^{n-1} \exp(x_j)}$$

with $i = 0, \ldots, n-1$. In the case of the gating network, with the *softmax* function used for the activation of the output layer, the weights satisfy the property $\sum_{i=0}^{n-1} w_i = 1$.

As it is common that some of the weights produced by the gating NN are very close to zero, it can be beneficial to limit the number of expert NNs and evaluate only models which make a significant contribution to the global output. This approach is referred to as hard selection and it includes an additional processing step to the computation of the weights. For a given value $k \in \{1, \ldots n\}$, the *Top K* algorithm selects the k largest values from a vector with n entries. In the context of the gating NN, the *Top K* algorithm can be used to select the k expert models with the largest associated weight values and only these NNs are considered in the evaluation of the MoE model. If the most of the weight values are close to zero, it is expected that this does not have a significant influence on the global output. However, as a large total number of expert models n is considered, this *Top K* approach can significantly reduce the computational effort for evaluating the MoE. Regarding the training of MoE models which include the *Top K* selection, optimizers based on stochastic gradient descent can show worse convergence behavior, as the *Top K* operation is generally not differentiable. With regard to these cases, an adjusted training process is required or the *Top K* operation is neglected during training. The MoE models which do not include a *Top K* algorithm in the gating network and compute the output of every expert NN for the global output, are referred to as soft selection.

The general structure of the MoE model including the *Top K* processing of the weights values is presented in Figure 4.5. In this figure the parts colored in blue

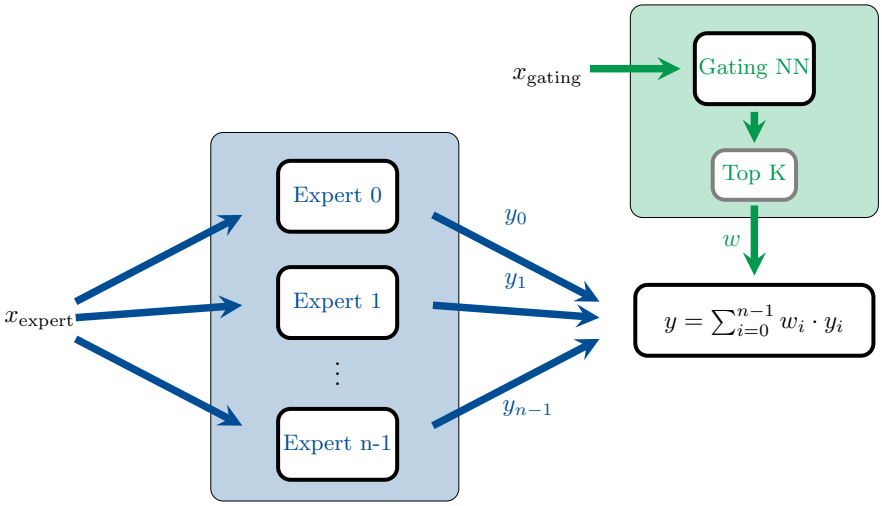


Figure 4.5: Illustration of the general structure of the Mixture of Experts model. Parts of the model colored in blue refer to the expert neural networks and parts colored in green refer to the gating network.

refer to the expert NNs and the parts colored in green refer to the gating model. The output of the gating model is passed on to the *Top K* selection and the weights are subsequently used for computing the weighted sum for the global output y. The *Top K* operation is presented in a gray frame to indicate that this operation is not essential for every MoE model. The *Top K* processing of the weights is not given in MoE architectures that use soft selection of the experts.

4.3.2 Mixture of Experts as Surrogate Model

In this section, we discuss the application of the MoE model architecture for developing a surrogate model for the homogenization of the beam frame RVE. As we consider the trained MoE model to predict the homogenized stress components for various open-porous material, we use the deformation gradient and a representation of the material properties as input. Large parts of this work are covered in [83] which has not yet been published at the time of writing.

As described in Section 4.3.1, each MoE model is characterized by the definition of the expert models and the gating network. Regarding the surrogate model for the

beam frame problem, we base the architecture for the expert NNs on the insights of Chapter 3. As the results have shown, the MLP model is well-suited to reproduce the stress-strain relationship for a single given RVE. To expand on this approach, we consider for each of the expert models an MLP model architecture and train the NN to predict the micromechanical behavior of a single material. Similar to the approaches presented in Chapter 3, each expert NN takes the macroscopic deformation gradient $\overline{F}$ as input and computes the homogenized stress tensor $\overline{P}$ based on this information. We consider the first Piola-Kirchhoff stress tensor as a stress measurement and as output of the expert NNs. Therefore, the global output of the MoE model is given by the weighted sum

$$\overline{P} = \sum_{i=0}^{n-1} w_i \cdot \overline{P}_i$$

with n the number of expert NNs, $w \in \mathbb{R}^n$ the weight vector, and $\overline{P}_i$ the output of the expert model with index i for $i \in \{0, \dots, n-1\}$. Each of the expert models shares the same MLP architecture. We consider a similar layer composition as in Chapter 3. Each expert NN consists of three hidden layers with $GELU$ activation and 256 neurons in each layer. As we consider the three-dimensional beam frame model, the dimension of both the input and output vectors equals nine.

The weight vector $w \in \mathbb{R}^n$ is computed as output of the gating network. In the case of the MoE surrogate model for the beam frame microstructure, we consider the gating model to take a parameterized representation of the material's microstructure as input. This representation is given by the material feature vector $I_{RVE} \in \mathbb{R}^{10}$ as described in Section 4.1. As the results of Section 4.2 show, different RVEs can be distinguished by a NN model based on the given material features. The results presented in Section 4.3.4 provide a deeper insight with regard to the influence of the input features on the output of the gating model.

For the architecture of the gating model, we also consider an MLP architecture. Based on manual testing, we define the gating network with two hidden layers and 128 neurons in each layer. The activation functions for the hidden layers are given

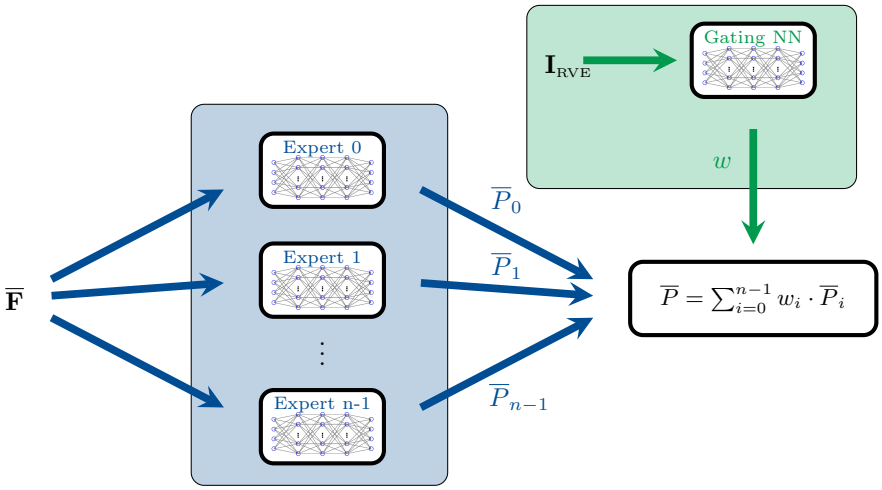


Figure 4.6: Illustration of the Mixture of Experts model for homogenization. Parts of the model colored in blue refer to the expert neural networks which process the deformation gradient. Parts of the model colored in green refer to the gating network which processes the material parameters.

by *tanh* and *ReLU*. As we expect the weights over the number of experts to sum up to one, we use the *softmax* activation function for the output layer of size n. For training the MoE model, we use soft selection of the expert models. This means that we do not include a *Top K* operation for determining the weight vector w. The full structure of the MoE surrogate model is illustrated in Figure 4.6. The sections colored in blue refer to the expert models and the sections colored in green refer to the gating NN.

4.3.3 Training Approach

The data sets, which we use for training the MoE, are based on the descriptions in Section 4.1. The RVEs that are used for computing the stress response for given deformation gradients are created based on the Laguerre-Voronoi tessellation. The pore size distributions and porous fractions that are used as input for the creation method are defined to resemble experimental data of various cellulose aerogels. Each data point in the data sets that are considered for the training of the MoE contains

a macroscopic deformation gradient $\overline{F}$ and a material feature vector $I_{RVE} \in \mathbb{R}^{10}$ as inputs. The entries of the material feature vector are based on the properties of the RVE. A description of the values is presented in Section 4.1. The homogenized first Piola-Kirchhoff stress tensor $\overline{P}$ corresponding to the given deformation gradient and RVE is given as output for each data point within the data sets.

We consider the 250 mutually different training RVEs as a basis for creating the training data set. Additionally, the training RVEs are used to generate a validation data set that includes the same material feature vectors as the training RVE but features different deformation and stress data. Furthermore, 50 validation RVEs are considered to create an additional validation data set that features different material feature vectors and different deformation states compared to the training data set. For each considered RVE, 100,000 macroscopic deformation gradients and corresponding homogenized stress tensors are computed.

We split the training of the MoE model in two separate training steps. In the first step, we train each expert NN to replicate the mechanical behavior of a single RVE. This is the same approach as the training of the MLPs presented in Chapter 3. As the results in Chapter 3 show, the MLP architecture is well suited for training a model on the data resulting from a single RVE. For each expert NN, we use 90,000 of the computed data points for the respective RVE as training data and the remaining 10,000 data points as validation data. Each expert NN is trained for 2,000 epochs with Adam optimizer [80] and a learning rate scheduler reducing the learning rate from $1e-3$ to $1e-5$. The mean squared error (mse) is considered as loss function. The loss curves for the training of four expert models is presented in Figure 4.7. The graphs show that for each expert model, the loss is significantly reduced over the range of 2,000 training epochs. The reduction of the validation loss is very similar to the reduction of the training loss and we end up with only slightly higher loss values with regard to the validation data. This behavior is very similar to the training behavior of the models discussed in Chapter 3. With this first training step, we achieve that every expert NN is trained to replicate the mechanical behavior of a single given RVE.

In the second training step, we train the MoE model to adjust the weights of

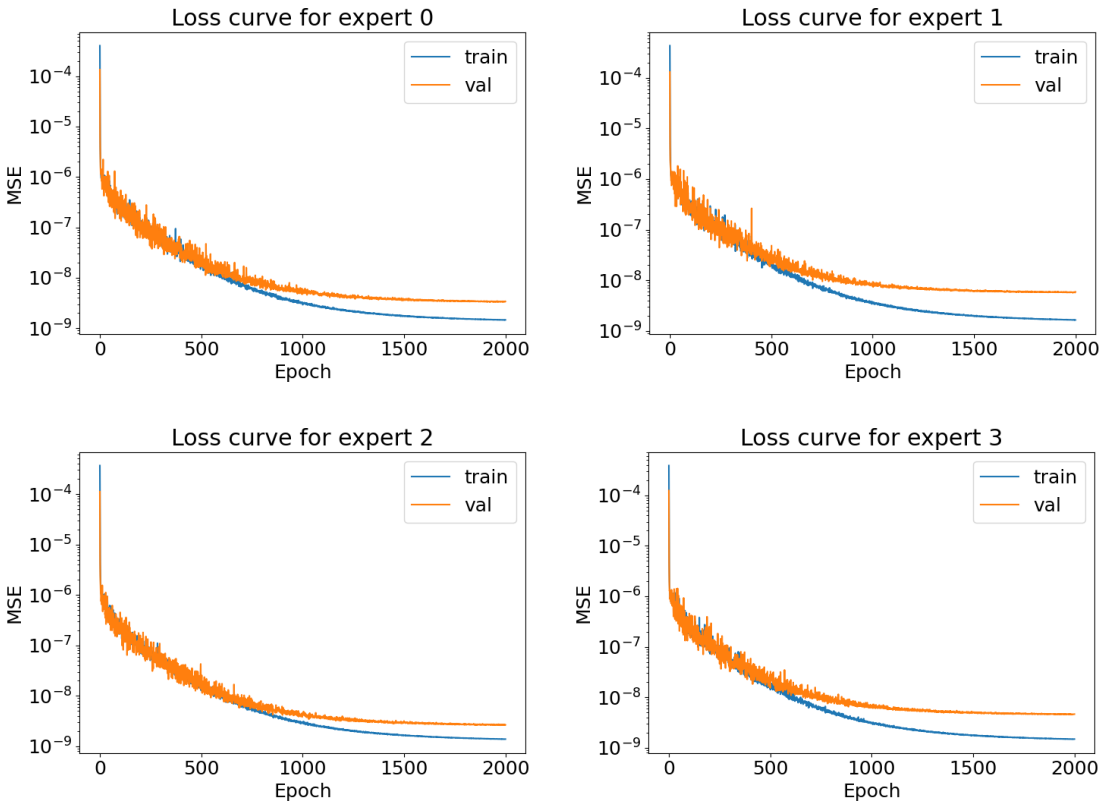


Figure 4.7: Loss curves resulting from training of four expert models. Progression of the training loss is illustrated with the blue curve and progression of the validation loss is given by the orange curve. Each model is trained for 2,000 training epochs.

the gating network. In this training phase, we can fix the weights of the expert model or define the expert weights as trainable. We refer to the approach of having all weights of the MoE model trainable including the weights of the expert NNs as the coherent training approach. The approach of adjusting only the weights of the gating model in this training phase with the weights of the expert NNs fixed is referred to as incoherent training approach.

4.3.4 Microscopic Performance for Different Architectures

Regarding the two training steps for optimizing the weights of the MoE model, we focus in this section on the second training step which is used to adjust the

weights of the gating model. Here, we consider the coherent training approach as well as the incoherent approach for different MoE architectures and compare the performance with regard to the reduction of the loss values. Additionally, we analyze the behavior of the gating model regarding the distribution of the resulting weight values for different material inputs.

For a given MoE architecture with $n \in \mathbb{N}$ experts, we use the pre-trained expert models resulting from the training process described in Section 4.3.3. The expert models are each trained on the mechanical behavior of one of the 250 training RVEs described in Section 4.1. From the group of 250 possible expert RVEs, we select n RVEs and use the corresponding expert models for the MoE. The selection of the RVEs is based on the magnitude of the stress values given for the corresponding data. As we target to predict stresses for a wide range of open-porous materials with the MoE model, we are also interested in selecting expert models that cover a large range of nanoporous properties. Regarding the mechanical behavior of the materials, we observed that the correlation between the components of the deformation gradient and entries of the resulting stress tensor is very similar for different RVEs. Though, the main difference between RVEs is explained by the magnitude of the stress values. Therefore, we select the set of expert models for the MoE approach to maximize the difference in the stress magnitudes between the materials. From 250 RVEs that are used for creating the training data, we target to find a selection of n RVEs which show the largest difference. This description translates to the max-min dispersion problem. This optimization task is a well-known NP-hard problem in computational science [119, 95]. For a given number of experts $n \in \mathbb{N}$, the problem can be described as finding a set of indices $\mathcal{Z}_{expert} \subset \{0, 1, \ldots, 249\}$ with $|\mathcal{Z}_{expert}| = n$ that maximizes the expression

$$\min_{\substack{i,j \in \mathcal{Z}_{expert}, \\ i \neq j}} |\Delta_{RVE_i} - \Delta_{RVE_j}|.$$

As it is sufficient for our application to find an acceptable approximation of the optimal set of RVEs, we use a greedy algorithm [24, chapter 15] to compute the

solution. This algorithm starts with a single RVE that defines the initial set of expert RVEs. In each step of an iterative process, the respective RVE is added to the set that features the largest distance to all current RVEs in the list of expert RVEs. This process is repeated until the number of RVEs in the list equals n. For the application with regard to the selection of expert RVEs, we use the same initialization of the greedy algorithm for any given number of experts $n \in \mathbb{N}$. Here, the initial set of expert RVEs is defined as the set containing only the single training RVE that features the lowest magnitude of stress value.

The first MoE model that we consider to be trained on the stress data of multiple RVEs uses ten expert models for computing its output. We train the surrogate model based on the incoherent approach which means that we keep the weights of the expert model fixed in the second training step. Only the weights of the gating network are adjusted in the training process. We use the Adam optimizer with a learning rate scheduler for training. The learning rate is reduced from $1e-3$ to $1e-5$ over all the considered training epochs. During training, the loss values with regard to the training data, val-F data, and val-RVE data are tracked. This allows us to evaluate the model's performance with regard to the validation data sets that differ from the training data. A batch size of 256 is used to divide the data sets. For the MoE model with ten experts, the progression of the loss terms is illustrated in Figure 4.8. In the image, the blue curve refers to the training loss, the orange curve refers to the val-F loss, and the green curve refers to the val-RVE loss. The graph shows that all losses are significantly reduced over 300 training epochs. The training loss and the val-F loss show very similar behavior. Especially, with regard to the last 150 training epochs, the curves are almost aligned and result in very similar loss values after the last training epoch. With regard to the training data the final loss values is $1.13e-1$ and the final loss value based on the val-F data equals $1.09e-2$. In Figure 4.8, the green curve referring to the val-RVE data shows different behavior than the curve referring to the training data. After a strong reduction of the error within the first few epochs, the val-RVE loss curve stays on a relatively constant level. After the last epoch the val-RVE loss equals $2.87e-1$ and is significantly higher than the loss terms based on the training RVEs. Based on

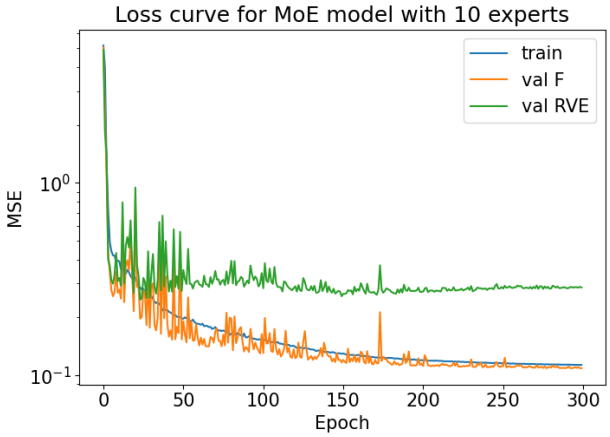


Figure 4.8: Loss curve for the incoherent training approach of the MoE model with ten experts. Blue curve refers to the loss regarding the training data, orange curve refers to the loss regarding the val-F data and green curve refers to the loss computed based on the val-RVE data.

these results, the MoE model performs much better regarding the stress predictions for the training RVEs in comparison to the predictions for the validation RVEs. Note that these results consider only the evaluation on the microscopic scale.

For further analysis of the gating model's behavior for different material inputs, we evaluate the gating output for the 250 training RVEs and 50 validation RVEs. The mean and variance values with respect to each output weight of the gating model are presented in Figure 4.9. The image on the left presents the mean output of the gating model over all 300 material feature vectors given for the training and validation RVEs. The height of each bar in the graph displays the mean weight computed by the gating model for the respective expert. The labels on the x-axis refer to the indices of the expert models that are used within the MoE framework. The indices are based on the order of the 250 training RVEs. The chart shows that the weight referring to the expert model with index 69 has the highest average. This means that the output of the respective expert NN has the highest average contribution to the weighted sum which defines the MoE model's output. The mean weights for some expert NNs are significantly lower. For example the experts with index 58 and 101 only account for a small average fraction of the weighted sum.

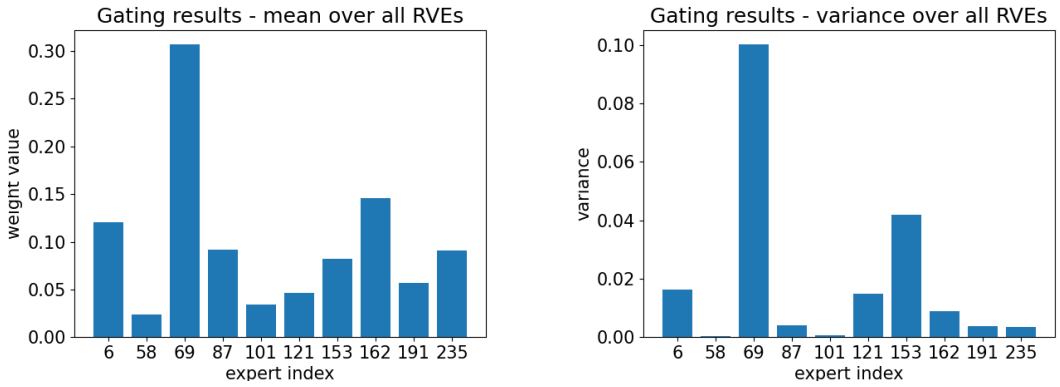


Figure 4.9: Mean and variance of the incoherently trained gating model with ten experts. The image on the left illustrates the mean output of the gating model over all 300 RVEs. The image on the right presents the variance of the output over all considered RVEs.

The image on the right of Figure 4.9 shows the variance with regard to the output of the gating model. This covers the output for all 300 material input vectors and computes the variance individually for each weight component of the gating output. In the chart, each bar refers to the weight output for a respective expert NN and the height of the bar represents the variance of the weight value over the set of 300 RVEs. This illustration helps to indicate how strongly the weights are influenced by the input material features. The greater the variance of a weight, the more the value depends on the value composition of the input vector I_{RVE}. The figure shows that the weight referring to the expert with index 69 has the highest variance. This expert shows also the highest average weight. The high variance indicates that there are relatively strong differences between the weight values for different RVEs. For experts with index 58 and 101, the variance is close to zero. This indicates that the weights regarding these expert NNs are almost constant for different material features.

As the selected set of experts that is used in the MoE model is based on the 250 training RVEs, the considered training data set includes data points that are computed based on these selected RVEs. This means that for these data points, the prediction of the corresponding expert NN included in the MoE model is expected

to show only minor deviations to the target stress data. Therefore, it is to be expected that the best output of the gating network assigns a weight of one to the respective expert and a weight of zero to all other experts. For given material features, this would lead to the gating model perfectly identifying the RVEs that the expert models are based on. However, as we investigate this behavior, we see that the gating model does not always identify the correct expert model for the respective material features. For the material features computed from the RVE with index 69, the gating network identifies the correct expert model with relatively high confidence, as the weight for the expert with index 69 equals 0.90. Similar behavior is also given for the RVE with index 153, as the gating model computes a weight value referring to the expert model with index 153 of 0.89. However, for the other experts, the results are mixed. If the material features of the RVE with index 6 are used as input, the respective weight referring to expert 6 equals 0.36. For these input features, the expert with index 6 has the greatest contribution to the weighted sum that defines the MoE output, though the weight value is significantly smaller as for the RVEs with indices 69 and 153. For the material features based on the RVE with index 58, the weight value for the corresponding expert model is even smaller. Here, the weight referring to the expert model with index 58 equals only 0.05. This means that in the weighted sum, the contributions of the other expert models are weighted significantly higher.

For a deeper insight regarding the relationship between the material input features and the corresponding output of the gating model, we compute the permutation feature importance [3] for the data set. For a given model, this measurement is a tool to evaluate the importance of each input feature for every component of the output vector. The idea of this measurement is to randomly shuffle one input feature of a given data set and compare the output of the modified data set to the original output. For defining the permutation feature importance, we consider the input data set D_{gating}. For a given input sample I_{RVE} we introduce the permuted sample $\hat{I}_{RVE,j\leftarrow\pi}$ which is identical to I_{RVE} in all components except of feature j which is randomly permuted with the j-th features of the other samples of the data set. We define $\Psi \in \mathbb{R}^{n\times n_{in}}$ as the matrix containing the importance measure-

ments regarding each feature and each output. The respective entry $\Psi_{i,j}$ for the importance of feature j for output i is then defined as

$$\Psi_{i,j} = \frac{1}{|D_{in}|} \sum_{I_{RVE} \in D_{in}} \left((f_{gating}(I_{RVE}))_i - (f_{gating}(\hat{I}_{RVE,j \leftarrow \pi}))_i \right)^2$$

for a given gating network f_{gating}. Note that the computations which define the values of the matrix Ψ are not deterministic, as the outcome of the analysis depends on the way the input components are shuffled. However, differences based on the randomness of the approach are expected to be minor. Regarding the application of the analysis in this work, we do not see significant variation in the results for different shuffling of the input features. Although individual values may differ slightly, the findings of the analysis remain the same.

For the incoherently trained MoE model with ten experts, the results of the permutation feature importance measurement are presented in Figure 4.10. The figure illustrates the values of the matrix $\Psi \in \mathbb{R}^{10 \times 10}$ with each tile of the graph referring to an individual entry of the matrix. Here, the color of each tile is defined based on the respective matrix value. Red colored tiles indicate a high sensitivity of an output component to changes in the respective input, while blue colored tiles indicate low sensitivity. Regarding the input parameters, the results in Figure 4.10 show that the porous fraction (abbreviation pf in the figure) generally has the highest importance for the output of the gating model. This aligns also with our finding that the main difference between RVEs can be explained by the magnitude of the stress values Δ_{RVE} which is also strongly correlated with the porous fraction. The highest importance value can be found for the porous fraction with respect to the weight value of expert with index 69. This result fits the observation with regard to the variance values presented in Figure 4.9, as it supports the statement that the weight value is highly dependent on the input material features. Additionally, regarding the importance with respect to the weights which correspond to the experts with indices 6, 121, and 153, we see a meaningful correlation to the variance values presented in Figure 4.9. For components with high variance, the input features also

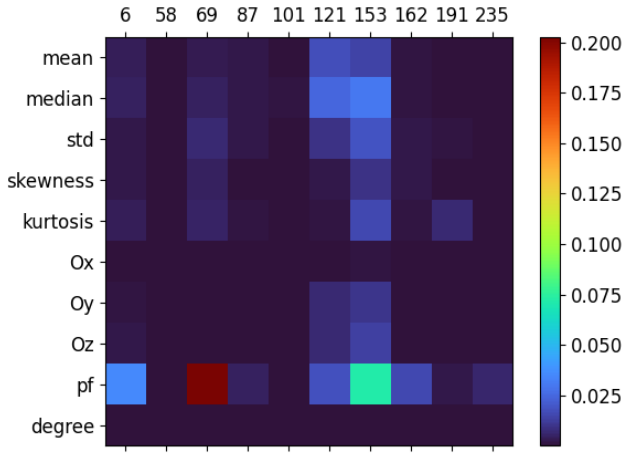


Figure 4.10: Results of the permutation feature importance measurement for the incoherently trained MoE model with ten experts. Each presented tile refers to a single component of the importance matrix Ψ with the colors representing the value of the respective entry.

show greater importance.

As we discuss the performance of the MoE model with ten expert NNs, we also regard the influence of an increased number of experts on the model's performance. This means that we use the same structure of the hidden layers for the gating network and increase the number of experts to twenty. The MoE model is again trained using the incoherent approach with Adam optimizer and the same learning rate scheduler. The selection of the RVEs that the considered expert NNs are based on is computed with the greedy algorithm for the max-min dispersion problem. The loss progression of the MoE model with twenty experts is presented in Figure 4.11. Similar to the training of the MoE with ten experts, we see a strong reduction of the training loss and the loss based on the val-F data. After 300 epochs of training, the MSE regarding the training data equals $9.22e-2$ and the loss regarding the val-F data equals $9.34e-2$. For the loss based on the data resulting from the validation RVEs, we see a different progression. The green loss curve does not decrease substantially after the first few epochs. The final values of the val-RVE

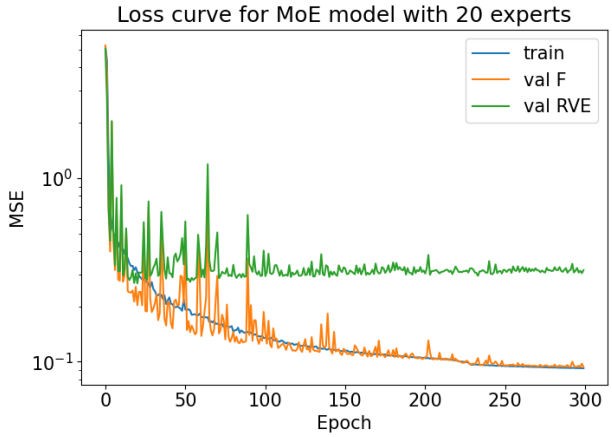


Figure 4.11: Loss curve for the incoherent training approach of the MoE model with twenty experts. Blue curve refers to the loss regarding the training data, orange curve refers to the loss regarding the val-F data and green curve refers to the loss computed based on the val-RVE data.

loss equals $3.16e-1$. In comparison to the training behavior of the RVE with ten expert NNs, we see slightly better reduction of the MSE with regard to the data sets based on the training RVEs. However, regarding the val-RVE data set, the loss of the MoE model with twenty experts is even higher than the resulting loss of the model with ten experts. Overall, the differences between both model with regard to the microscopic error are only minor. Though, based on the results, an increase in the number of considered expert models does not yield better microscopic performance of the MoE approach.

The mean and variance of the output values resulting from the gating model are presented in Figure 4.12. The chart on the left referring to the mean output shows that the expert models with indices 69 and 219 have the highest average weight values. It is notable that the expert with index 69 shows the highest average weight value for the gating model with twenty experts as well as for the gating model with ten experts (see Figure 4.9). For the expert models with indices 37 and 162, the average weight value is close to zero. This means that for none of the considered RVEs, the output of the respective expert models provides any meaningful contribution to the weighted sum that defines the output of the MoE model. Regarding

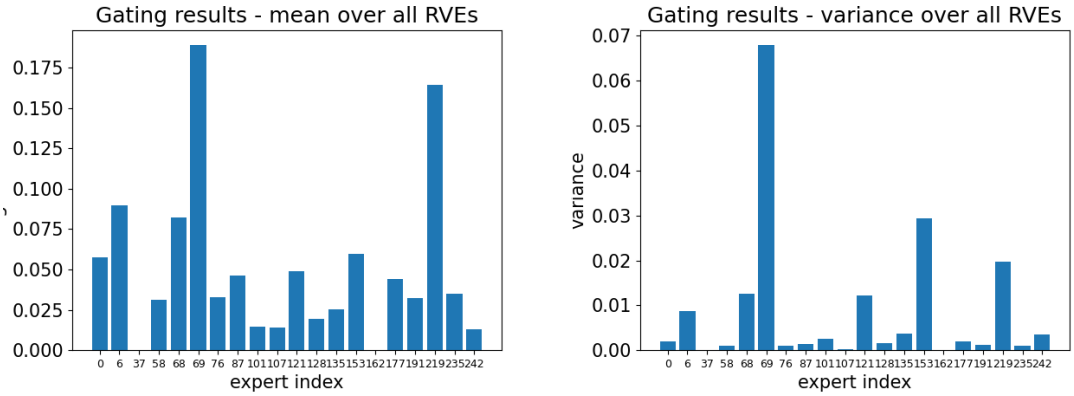


Figure 4.12: Mean and variance of the incoherently trained gating model with twenty experts. The image on the left illustrates the mean output of the gating model over all 300 RVEs. The image on the right presents the variance of the output over all considered RVEs.

the variance of the weight values, we see the highest variances for the weights referring to the experts with indices 69, 153, and 219. However, the variance values are generally smaller compared to the variances for the MoE model with ten experts presented in Figure 4.9.

For the MoE model with twenty experts, the results of the permutation feature importance measurement are presented in Figure 4.13. Similar to the MoE model with ten experts, the color of the tiles shown in the figure refers to the value of the importance measurement for the respective pair of input and output components. Similar to the measurements for the MoE model with ten experts, we see that the porous fraction (abbreviation pf) has the highest importance for the output of the gating model. Here, the highest importance values are computed with respect to the weights of the expert models with indices 69 and 153. This finding aligns with the high variances of the respective weights presented in Figure 4.12. The average degree of the beam frame structure does not have a significant impact on the output of the gating model, as the importance values with regard to this input feature are close to zero.

In addition to the incoherent training approach, we also consider the coherent approach for training the MoE model. This means that the weights of the gating

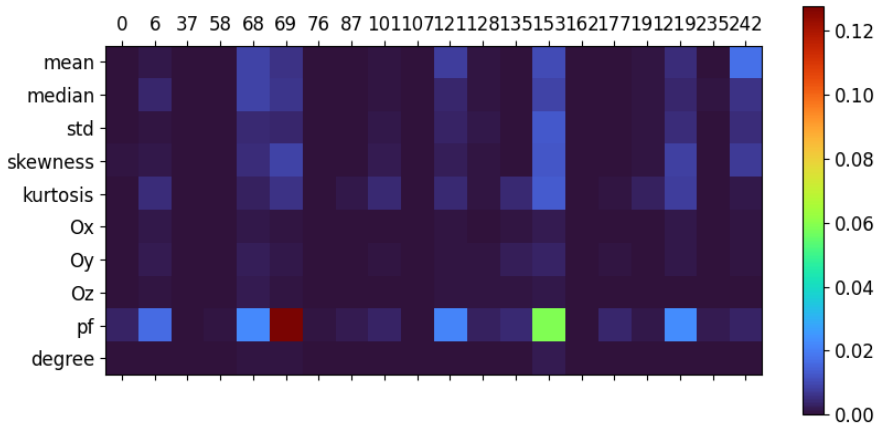


Figure 4.13: Results of the permutation feature importance measurement for the incoherently trained MoE model with twenty experts. Each presented tile refers to a single component of the importance matrix Ψ with the colors representing the value of the respective entry.

model and the weights of the expert models are set trainable in the second training step. Here, we still use the weights of the pre-trained expert NNs to initialize the MoE model. However, the weights of the experts are not fixed but can be adjusted during the training of the MoE model. We consider the MoE architecture with ten experts to test the coherent training approach. The set of expert models used for this approach is given as approximate solution of the max-min dispersion problem computed with the greedy algorithm. As we use the same initialization of the greedy algorithm, this results in the same set of expert indices used for the incoherently trained MoE model with ten experts.

The loss curve resulting from the coherent training approach for the MoE model with ten experts is presented in Figure 4.14. The MoE model is trained for 300 training epochs with Adam optimizer and learning rate scheduler which reduces the learning rate from $1e-3$ to $1e-5$ over the considered training epochs. For training, the data sets are partitioned into batches of size 256. Figure 4.14 shows that the blue curve referring to the training data set has the greatest reduction in the MSE

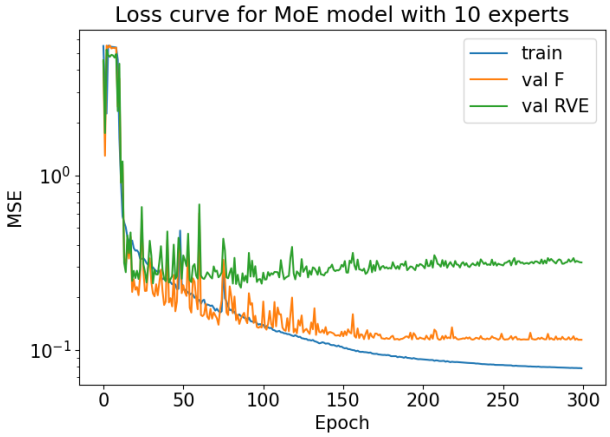


Figure 4.14: Loss curve for the coherent training approach of the MoE model with ten experts. Blue curve refers to the loss regarding the training data, orange curve refers to the loss regarding the val-F data and green curve refers to the loss computed based on the val-RVE data.

value. After 300 epochs of training the final training loss equals $7.86e{-}2$. Especially, for the last 200 training epochs, the difference between the curve of the training loss and the orange curve referring to the validation loss regarding the deformation gradient (val-F) grows with an increasing number of training steps. After training, the val-F loss equals $1.14e-1$. For the green curve referring to the validation loss based on the data computed with the validation RVEs, we see only a significant reduction regarding the MSE value within the first few training epochs. For large parts of the training process, the loss remains almost constant. After 300 training epochs, the final loss with regard to the val-RVE data equals $3.17e-1$. As we see a growing difference between the training loss and both considered validation loss values, the results indicate some overfitting behavior of the coherently trained MoE model. With regard to the generalization behavior of the MoE model, the coherent training approach does not yield significantly better performance.

The results in Figure 4.15 show the mean and variance of the output values computed with the coherently trained gating model with ten experts. The bar chart on the left refers to the average output components of the gating model with respect to the material features of all 300 considered RVEs. The chart shows

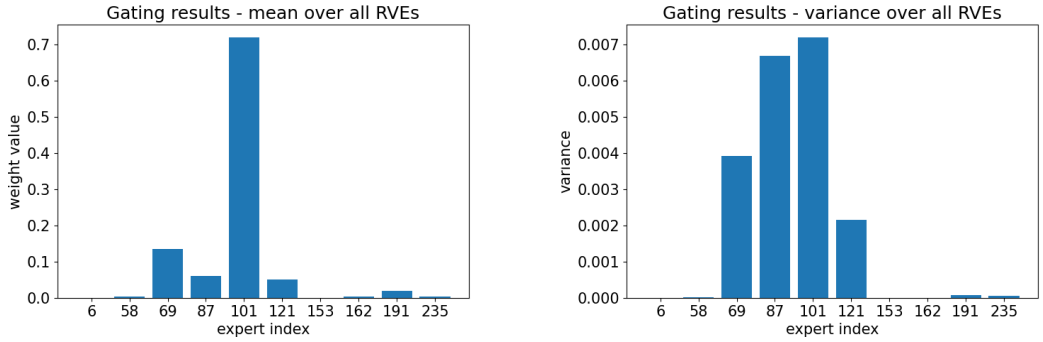


Figure 4.15: Mean and variance of the coherently trained gating model with ten experts. The image on the left illustrates the mean output of the gating model over all 300 RVEs. The image on the right presents the variance of the output over all considered RVEs.

that the weight corresponding to the expert with index 101 has the highest average weight. This means that the respective output of the expert has the greatest average contribution to the weighted sum that defines the output of the MoE model. Note that the weights and the resulting output of the expert with index 101 does not necessarily have to be the same as those of the expert model with index 101 used for the incoherently trained model with ten experts. As the MoE model is trained coherently, the weights and the resulting output of each expert model is expected to change. The image on the right of Figure 4.15 shows the variance for each of the weight components computed by the gating model. The chart shows that the weights corresponding to the experts with indices 69, 87, 101, and 121 have the highest variance. Therefore, the respective weight values are expected to be the most dependent on the composition of the material feature vector. Generally, we see a relatively low variance for the outputs of the gating model, as the scale of the bar chart shows significantly smaller values compared to the variance charts presented in Figure 4.9 and Figure 4.12. This indicates that the composition of the expert outputs for the weighted sum that defines the MoE output is very similar for different considered RVEs. The variance of the weights values referring to the expert models with indices 6, 58, 153, 162, 191, and 235 is close to zero.

In Figure 4.16, the results of the importance feature measurement are presented.

The measurement is based on the gating model that corresponds to the coherently trained MoE model with ten experts. Each tile displayed within the image refers to a component of the computed importance matrix Ψ. Each tile is colored based on the value of the respective matrix entry. The results presented in Figure 4.16 appear significantly different to the importance measurements based on the coherently trained MoE models presented in Figure 4.10 and Figure 4.13. With regard to the coherently trained model, we see generally significantly lower importance values compared to the values resulting from the coherent training approaches. The highest importance value of about 0.02 in Figure 4.16 is much lower than the maximum values of about 0.2 in Figure 4.10 and 0.12 in Figure 4.13. For the output components of the gating model, we see the highest importance values with regard to the weights referring to the expert models with indices 69, 87, 101, and 121. This finding aligns with the results of Figure 4.15, as the respective weight values also show the highest variance. In addition to the porous fraction (pf), the results in Figure 4.16 indicate also a significant importance of the input features that describe the material's pore size distribution (mean, median, std, skewness, and kurtosis) for the output of the gating network.

As we discussed the influence of increasing the number of experts for the incoherent training approach for the MoE model, we also consider an MoE model that includes twenty expert models and is coherently trained. The results presented for the coherent training approach based on the MoE model with ten experts does not yield a significant improvement with regard to the performance on the data based on the validation RVE. However, further analysis of the performance regarding the coherent training approach for the MoE model with twenty experts, is expected to yield further insight with regard to the differences between both approaches for training the MoE model. The MoE model considers the same set of pre-trained expert models that are also used for the incoherently trained MoE model with twenty experts. The set of indices results from the greedy algorithm that is used for approximating a solution of the max-min dispersion problem. Since we use the same initialization for the greedy algorithm, the identical expert indices are obtained. Although we use the same initialization with the weights of the pre-trained expert

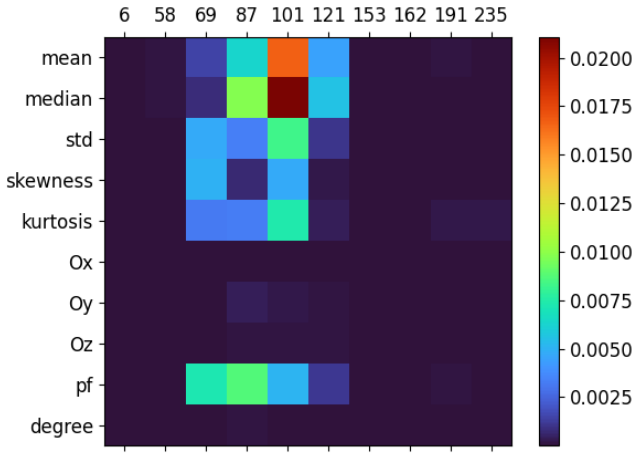


Figure 4.16: Results of the permutation feature importance measurement for the coherently trained MoE model with ten experts. Each presented tile refers to a single component of the importance matrix Ψ with the colors representing the value of the respective entry.

models, the weights of the experts are set trainable for the second training phase. Therefore, the expert weights of the coherently trained model are expected to differ from the expert weights of the incoherently trained model. For the coherent training approach, we consider 300 training epochs based on the Adam optimizer [80] with a learning rate scheduler reducing the learning rate from $1e-3$ to $1e-5$ over all considered training epochs. The batch size for partitioning the data is again set to 256.

For the coherent training approach for the MoE model with twenty experts, the progression of the loss values is presented in Figure 4.17. In the image, the loss curves are differentiated with regard to the considered training and validation data sets. The blue curve which shows the progression of the MSE loss computed for the training data has the greatest reduction. After 300 epochs of training, the final loss regarding the training data equals $6.99e-2$. The validation loss with regard to deformation gradient (val-F) is also significantly reduced within the training process. However, within the last 150 training epochs, the orange curve that displays the loss

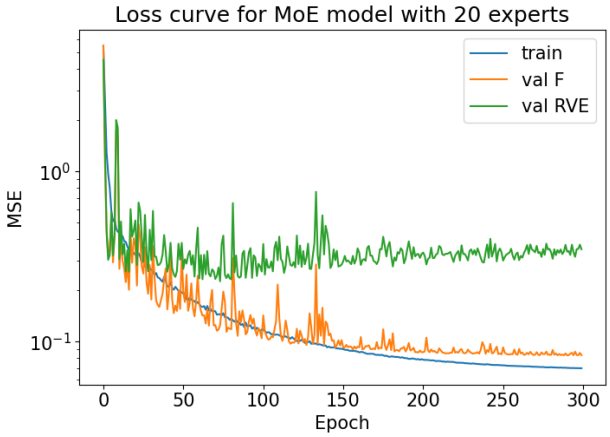


Figure 4.17: Loss curve for the coherent training approach of the MoE model with twenty experts. Blue curve refers to the loss regarding the training data, orange curve refers to the loss regarding the val-F data and green curve refers to the loss computed based on the val-RVE data.

computed for the val-F data set does not see the same reduction as the blue curve that refers to the training loss. After training, the val-F loss equals $8.33e-2$. The difference between training and validation loss is more pronounced with regard to the data computed with the validation RVEs. In Figure 4.17, the green curve shows the progression of the val-RVE loss. The image shows that the respective validation loss is only reduced within about the first 50 epochs of training. For the remaining training epochs, the MSE loss regarding the val-RVE data does not improve. The progression of the loss curve indicates even a slight increase in the MSE. After 300 training epochs, the loss value computed for the val-RVE data equals $3.49e-1$. This value is significantly higher than the loss computed for the training data. Based on the results for both validation data sets, we conclude that the coherently trained MoE model with twenty experts is affected by overfitting. As there is a distinct difference between the training loss and the validation loss values, we do not expect proper prediction quality with regard to open-porous materials that have not been included in the training data.

For analyzing the behavior of the gating network of the coherently trained MoE model with twenty experts, the mean and variance regarding the output of the

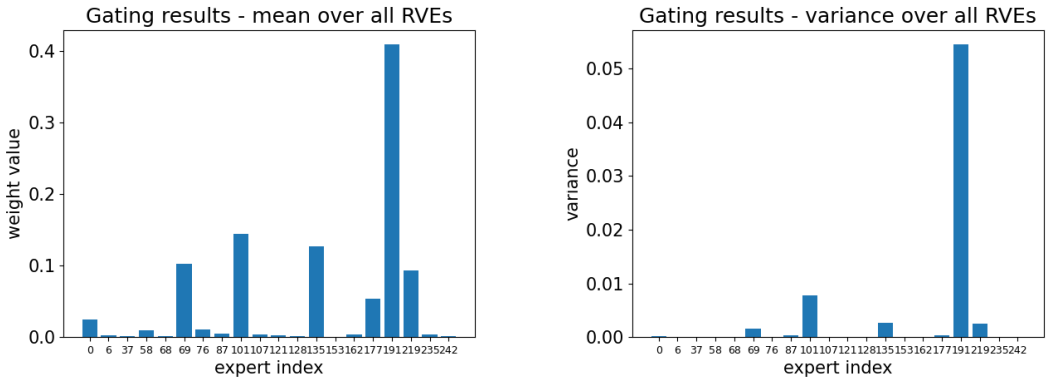


Figure 4.18: Mean and variance of the coherently trained gating model with twenty experts. The image on the left illustrates the mean output of the gating model over all 300 RVEs. The image on the right presents the variance of the output over all considered RVEs.

gating model is presented in Figure 4.18. For creating the plots, material features used as input for the gating model include both training and validation RVEs. In Figure 4.18, the mean over all 300 output vectors is illustrated in the bar chart on the left. The chart shows that the weight of the expert with index 191 has the highest average. Additionally, for the experts with indices 69, 103, 135, 177, and 219, the respective average weight value is also significantly greater than zero. As for the other weight values, the average is close to zero, it is expected that these experts provide that greatest contribution to the weighted sum that is computed as the output of the MoE model. The variances of the output components resulting from the gating model are presented in the right image of Figure 4.18. The bar chart shows that the highest variance is given for the weight corresponding to the expert with index 191. For the other experts, the variance of the weight values is close to zero. This indicates that the weight of the expert with index 191 is the most dependent on the material input features.

For additional analysis of the relationship between material input features and the weight values given as output of the gating model, we use the permutation feature importance measurement. The composition of the matrix Ψ that results from the computations is presented in Figure 4.19. In contrast to the previous illustra-

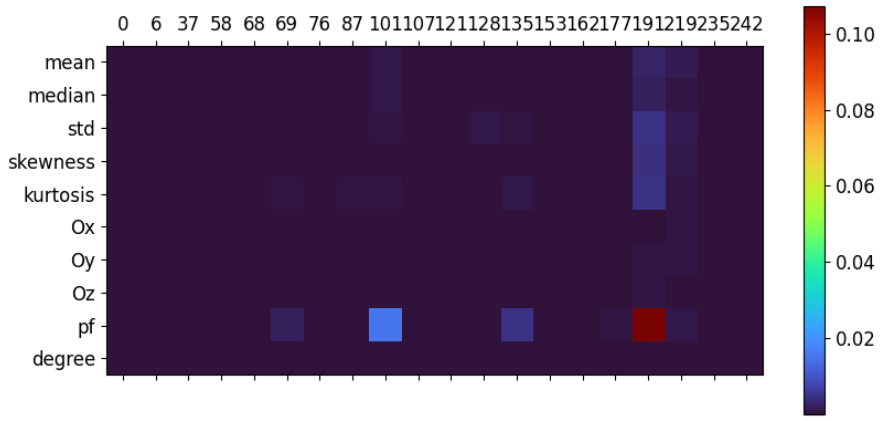


Figure 4.19: Results of the permutation feature importance measurement for the coherently trained MoE model with twenty experts. Each presented tile refers to a single component of the importance matrix Ψ with the colors representing the value of the respective entry.

tions that show the permutation feature importance for the other MoE approaches, the image in Figure 4.19 mainly highlights a single matrix component with a significantly higher value in relation to the other components. The porous fraction (pf) shows the highest importance value with respect to the weight value that refers to the expert with index 191. Additionally, the importance results indicate some dependence of the weight value referring to the expert with index 101 on the porous fraction input feature. However, based on the results, this dependence is less significant than the dependence with respect to the weight value for expert 191. Based on the coloring, the other components of the computed importance matrix Ψ appear relatively small and close to zero. These results align with the findings regarding the variance measurements shown in Figure 4.18, as the weight value referring to expert 191 also shows the greatest variance.

In general, the analyses of the presented MoE models reveal clear differences between the coherent and the incoherent training approaches. The gating models resulting from the coherent training method show significant lower variances with

regard to the computed weights. Additionally, for the coherently trained models, we see lower importance of the material input features for the output components. Analyses of the mean output values based on the coherently trained gating models have also shown that the number of expert models that yield a meaningful contribution to the weighted sum that defines the output of the MoE model is relatively low. On the other hand, for the gating models resulting from the incoherent training approach, the results show a higher number of expert models that contribute to the weighted sum with a weight distinct from zero. This behavior of the gating models leads to slightly better performance with regard to the data based on the validation RVEs.

To conclude the findings of this section, we present a comparison of the main microscopic results of the four described MoE approaches. For this comparison, we focus on the error with regard to the different data sets and additionally, introduce a confidence measurement for the gating network. The confidence value is computed based on the entropy. For a given gating model f_{gating} and a data set of input vectors for the gating model D_{gating}, the entropy $H \in \mathbb{R}$ is computed as introduced by Shannon in [136]. This means that for each weight value w resulting as output of the gating model, we compute the value $w \cdot \log(w)$ and sum it across all components. Formally, the entropy results from the relation

$$H = - \sum_{I_{RVE} \in D_{gating}} \sum_{i=0}^{n-1} \left(f_{gating}(I_{RVE})\right)_i \cdot \log\left(\left(f_{gating}(I_{RVE})\right)_i\right).$$

Note that for the computation of the entropy value, we add the machine accuracy to resulting weight values that equal zero. Since the logarithm of zero is not defined within the real numbers $\mathbb{R}$, having the logarithm for a weight value that equals zero would cause the computation to crash. Based on the entropy H, the confidence measurement is defined as

$$confidence = 1 - \frac{H}{\log(n)}$$

with the number of experts given as $n \in \mathbb{N}$. With this definition, for any gating model f_{gating} and any data set D_{gating} the confidence measurement lies within the interval of [0, 1]. The value indicates the extent to which the weighted sum used to compute the MoE model's output is dominated by the output of individual expert NNs. This means that the confidence value is close to one, if the gating network mostly determines the weights of a single expert model to be very high and for the other expert models to be close to zero. On the other hand, the confidence value is close to zero, if the gating model mostly predicts a relatively even distribution of the weights with all weights components close to $\frac{1}{n}$. Regarding a given MoE model, we compute the confidence value for the respective gating networks based on material features of the training and validation RVEs combined.

The results presented in Table 4.1 show the training losses computed after the last training epoch for each of the considered MoE models. The loss values refer to the MSE and are individually computed for the training data, for the validation data with regard to the deformation gradient (val-F), and for the validation data with regard to the material features (val-RVE). Additionally, the confidence values for each respective gating model are shown in the last column of Table 4.1. The comparison of the results shows that coherently trained MoE models yield a better reduction of the training loss. However, with regard to the val-F loss we do not see a clearly better performance of the coherently trained models and for the loss values based on the validation RVEs, the results of the coherently trained models are worse. The loss values indicate that the coherent training approach leads to a more pronounced overfitting behavior with regard to the material input features. The best performance with regard to the val-RVE data results from the incoherently trained MoE model with ten experts.

For the confidence measurements, the coherently trained MoE model with ten experts shows the highest value with a confidence of 61%. The values for the other three models is relatively similar between 36% and 43%. In general, the confidence values for the coherently trained MoE models are higher than the confidence values for the models based on the incoherent training approach. The low confidence measurements for the coherently trained gating models align with our findings that

training approach	#experts	training loss	val-F loss	val-RVE loss	confidence
incoherent	10	$1.13e-1$	$1.09e-1$	$2.87e-1$	36%
incoherent	20	$9.22e-2$	$9.34e-2$	$3.16e-1$	40%
coherent	10	$7.86e-2$	$1.14e-1$	$3.17e-1$	61%
coherent	20	$6.99e-2$	$8.33e-2$	$3.49e-1$	43%

Table 4.1: Collective results for different MoE approaches. The table shows the loss values after training with regard to the considered data sets. Additionally, the confidence measurements for the respective gating models are presented.

in the average output of the respective gating NNs, only few experts receive a weight that is significantly greater than zero.

4.3.5 Application in the Multiscale Framework

As Table 4.1 shows, the incoherently trained MoE model with ten experts yields the best results with regard to the data based on the validation RVEs. Therefore, we consider this model for the application within the multiscale framework and test it for different open-porous materials. To evaluate the performance of the MoE model with regard to multiscale simulations, we compare the macroscopic results to the results computed with the original beam frame solver on the microscopic scale. Additionally, we compare the error induced by the MoE approach to the error resulting from the MLP approach that we describe in Chapter 3.

With regard to the application of the MoE model within the multiscale framework, we see another advantage for the incoherent training approach. The results in Chapter 3 show that applying the MLP on the microscopic scale of the FE^2 method does not significantly influence the convergence behavior of the non-linear macroscopic solver. The results show the same number of iterations until convergence for the surrogate model as well as for the beam frame model on the microscopic scale. As we use the same approach based on the MLP architecture to derive the expert NNs for the MoE model, it is expected that each expert NN also provides the same convergence behavior. Since the MoE model computes its output as the

weighted sum of the outputs provided by the expert models, we expect the macroscopic solver, which uses the MoE model on the microscopic scale, to also exhibit proper convergence behavior.

For analyzing the macroscopic behavior of the multiscale approach with the microscopic computations based on the MoE model, we consider the torsion of a cube geometry on the macroscopic scale. This test case is similar to the simulation setup described in Section 3.3.3. The cube shaped geometry on the macroscopic scale is discretized with 20,480 P1 elements and contains 4,913 nodes. This results in a macroscopic problem with 14,739 DOF. For the non-linear solver on the macroscopic scale, we consider Newton's method with the tolerance set to $1e-6$. To define the macroscopic deformation, we apply Dirichlet boundary conditions on the two opposing surfaces that represent the boundaries of the cube in x-direction. For the nodes located within these surfaces, we set the deformation in y and z direction dependent on the respective distance from the center of the boundary. In total the Dirichlet boundary conditions apply a torsion of 36°to the cube. For the nodes affected by the Dirichlet boundary conditions, the deformation in x-direction is set to zero.

For defining the computations on the microscopic scale, we consider a random RVE selected from the set of 50 validation RVEs that are described in Section 4.1. This means that no stress data based on this RVE has been included in the training for the MoE model. We use the selected RVE to define the beam frame problem, as well as to compute the material feature vector I_{RVE} based on this RVE. With this configuration of the beam frame model and the MoE approach, the results are expected to be comparable.

Newton's method shows fast convergence behavior for both microscopic models. As we use a single load step to apply the deformation, both multiscale methods achieve a sufficiently small macroscopic residuum within four Newton iterations. This means that for this application, using the MoE model for the microscopic computations does not influence the macroscopic convergence behavior in comparison to the beam frame model on the microscopic scale.

The computed solutions of the described torsion test are presented in Figure 4.20.

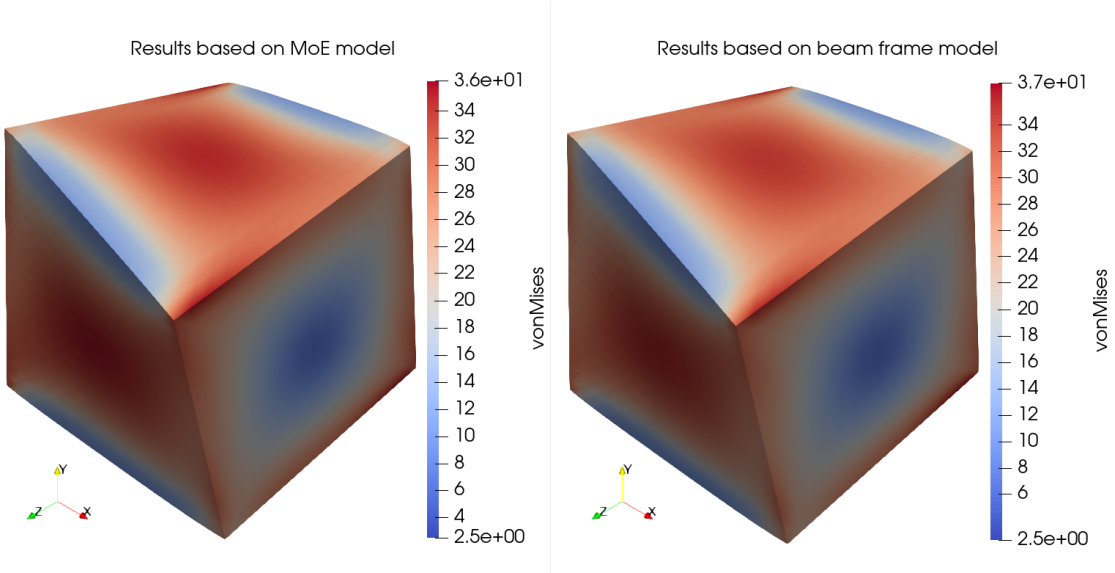


Figure 4.20: Results for the torsion of the cube geometry computed with the beam frame model and the MoE model. The image on the left shows the macroscopic results of the simulation based on the MoE model and the image on the right shows the results computed with the beam frame model on the microscopic scale. The colors of both solutions represent the von Mises stress resulting from the deformations.

Here, the image on the left shows the solution computed with the MoE model on the microscopic scale and the image on the right shows the solution with the microscopic computations based on the beam frame model. The coloring of the geometries illustrates the distribution of the von Mises stress measurement within the cubes. Based on the distribution of the stresses, there is no visual difference between both results. However, the scales of the von Mises stress values show that the stress measurements are not exactly identical, as both scales feature different maximum values. Therefore, to highlight the differences of both results, Figure 4.21 illustrates the distribution of the absolute deviation between both solutions. The image on the left shows the deviation with respect to the macroscopic deformation. In each point the local deviation is computed as the norm over the differences of

the deformation vectors for both solutions

$$\|\overline{\boldsymbol{u}}_{BF} - \overline{\boldsymbol{u}}_{MoE}\|_2$$

with $\overline{\boldsymbol{u}}_{MoE} \in \mathbb{R}^3$ referring to the local deformation vector resulting from the MoE computations and $\overline{\boldsymbol{u}}_{BF} \in \mathbb{R}^3$ referring to the deformation vector given from the solution computed with the beam frame model. The image shows that some regions of the geometry exhibit a higher deviation in terms of the deformation vectors. However, in total the deviation remains relatively low and does not exceed $4e-3$. With regard to both opposing surfaces of the geometry aligned with the $y-z$-plane, the error is zero at the corresponding nodes, as the same Dirichlet boundary conditions are used for both simulations. In Figure 4.21, the image on the right illustrates the distribution of the deviation with regard to the von Mises stress measurements. Here, the local deviation is computed as the absolute difference between both von Mises measurements

$$|\overline{\sigma}_{vM,BF} - \overline{\sigma}_{vM,MoE}|$$

with $\overline{\sigma}_{vM,BF}$ referring to the local von Mises stress based on the beam frame model and $\overline{\sigma}_{vM,MoE}$ referring to the von Mises stress based on the MoE model. The illustration of the deviation with regard to the von Mises stress shows that the error is relatively evenly distributed over the visible surface of the geometry. The image shows only small regions in which the deviation is higher.

The results in Figure 4.9 and Figure 4.12 show that for both incoherently trained MoE models, the expert with index 69 provides the greatest average contribution to the weighted sum that defines the output the MoE models. Based on this behavior of the trained gating models, the results indicate that the stress-strain relation of the expert with index 69 is expected to be similar to the respective relation of a wide range of RVEs included in the set training RVEs. As this expert appears to explain a significant part of the stress-strain relationship of the MoE model, we use the MLP that defines the expert model as basis for a comparison between the

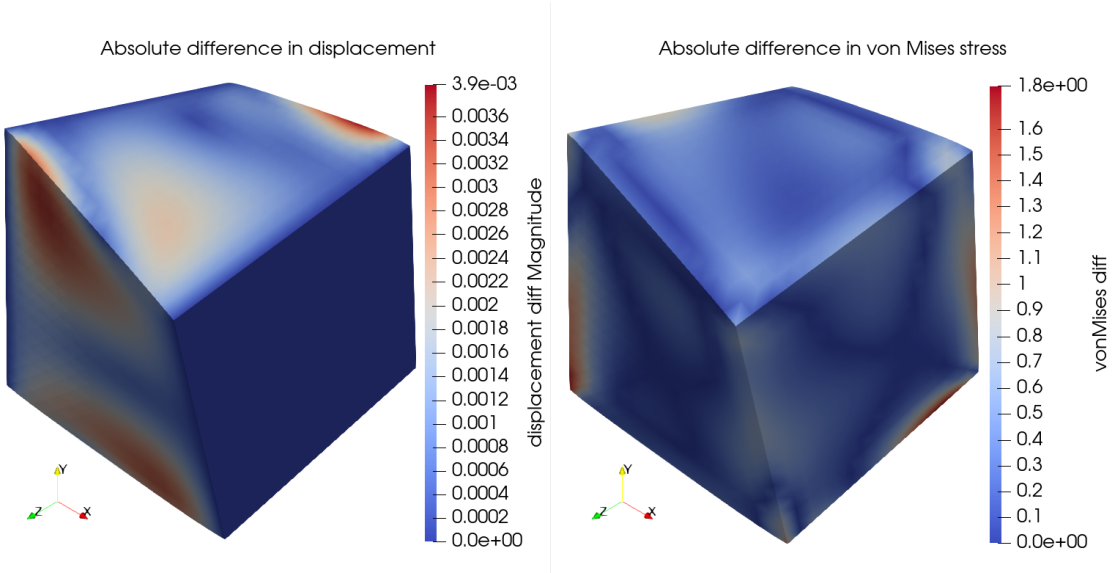


Figure 4.21: Difference between the macroscopic results computed with the MoE model and the beam frame model. The left image shows the difference between the solutions with respect to the deformation vector and the image on the right shows the difference with regard to the von Mises stress measurement.

MoE approach and the approach of using a single MLP on the microscopic scale as described in Chapter 3. As shown in Figure 4.2, the correlation between the components of the deformation gradient and the homogenized first Piola-Kirchhoff stress tensor are very similar for every considered RVE. Since the deformation response on the macroscopic scale is determined by the relationship between the deformation gradient and the stress tensor, the macroscopic deformation behavior for different RVEs is expected to be very similar. To quantify the benefit that the MoE approach provides to the multiscale simulations, we compare the performance of the MoE surrogate model to the performance of the approach that considers only the expert with index 69 on the microscopic scale.

To analyze the differences between both surrogate approaches, we compute the mechanical behavior regarding the torsion of the cube for all 250 training RVEs and 50 validation RVEs. For each of the RVEs, we run an individual simulation based on the MoE model that takes the material features as input. Additionally, for each of the RVEs, we run a multiscale simulation that uses the beam frame

model on the microscopic scale. The respective solutions are used as reference for the comparison. As the corresponding MLP of the expert with index 69 uses no material features as input, we run a single simulation that uses the MLP surrogate model on the microscopic scale. For each RVE, we compute the difference between the solution based on the MoE model and the solution based on the beam frame model. Additionally, we compute the difference of every solution computed with the beam frame model to the solution that uses the MLP of the expert with index 69 on the microscopic scale. To quantify the macroscopic error induced by the use of the surrogate models on the microscopic scale, we compute the relative difference in terms of the macroscopic deformation vector and in terms of the von Mises stress measurements. This means that we consider the error measurements

$$\frac{\|\overline{\boldsymbol{U}}_{BF} - \overline{\boldsymbol{U}}_{MoE}\|_2}{\|\overline{\boldsymbol{U}}_{BF}\|_2} \quad \text{or} \quad \frac{\|\overline{\boldsymbol{U}}_{BF} - \overline{\boldsymbol{U}}_{MLP}\|_2}{\|\overline{\boldsymbol{U}}_{BF}\|_2}$$

for the deformation vectors with $\overline{\boldsymbol{U}}_{BF}$ referring to the vector based on the beam frame model, $\overline{\boldsymbol{U}}_{MoE}$ being computed with the MoE model, and $\overline{\boldsymbol{U}}_{MLP}$ being based on the computations of the expert with index 69. Additionally, for the von Mises stress, we consider the error measurements

$$\frac{\|\overline{\boldsymbol{\sigma}}_{BF} - \overline{\boldsymbol{\sigma}}_{MoE}\|_2}{\|\overline{\boldsymbol{\sigma}}_{BF}\|_2} \quad \text{or} \quad \frac{\|\overline{\boldsymbol{\sigma}}_{BF} - \overline{\boldsymbol{\sigma}}_{MLP}\|_2}{\|\overline{\boldsymbol{\sigma}}_{BF}\|_2}$$

with $\overline{\boldsymbol{\sigma}}_{vM,BF}$, $\overline{\boldsymbol{\sigma}}_{vM,MoE}$, and $\overline{\boldsymbol{\sigma}}_{vM,MLP}$ referring to the vectors that contain the von Mises measurements for every node within the macroscopic finite element grid. As with the previous notation, the indices indicate which microscopic model the results are based on.

With the given error measurements, for the MoE surrogate model and the surrogate model based on the expert with index 69, we obtain 300 error values that include measurements for the training and validation RVEs. The distribution of these error values is illustrated with box plots in Figure 4.22. For each box plot, the box shows the range between first and third quartile and the length of the whiskers is computed based on the 1.5 interquartile range (IQR) value. Addition-

ally, for the data set of error values, the median is highlighted within the box. Data points that exceed the 1.5 IQR are marked as outliers. In Figure 4.22, the image on the left illustrates the distribution of the error values based on the deformation vectors. The blue colored box plots refer to the relative deviations based on the MoE model and the green box plots refer to the error values induced by the MLP surrogate model. Additionally, we differentiate between the error data based on the training RVEs and the error data based on the validation RVEs. The results show that the range between the first and third quartile is generally lower for the MoE model than for the expert with index 69. The error data for the MoE model shows a number of outliers with large error values. However, with regard to the total number of 300 data points, the number of large error values is relatively small. As the error data for the validation RVEs covers only 50 data points, the number of outliers is smaller and the 1.5 IQR marked by the whiskers is also smaller. The median and the range of the box is very similar for the training RVEs and for the validation RVEs. For the error data based on the MLP, we see a similar distribution of the error based on the training RVEs and the error based on the validation RVEs. The single outlier with very low relative error that is based on a training RVE results from the computations regarding the RVE with index 69. As the MLP is trained to predict the mechanical behavior of this beam frame model, the deviations are very small in this case. Although the MoE model generally shows smaller error values, with regard to the deformation vector, the difference between both surrogate approaches is relatively small.

In Figure 4.22, the image on the right illustrates the distribution of the error values based on the von Mises stress measurements. Similar to the measurements regarding the deformation vector, we show the distribution of the data referring to the MoE model in blue and the distribution of the data based on the expert with index 69 in green. Additionally, different box plots are presented covering the error values for the training RVEs and for the validation RVEs. The image shows that the error values resulting from the MoE model are significantly lower than the error values resulting from the MLP surrogate model. Except of few outliers, the vast majority error values for the MoE model are below the value of 0.2. The

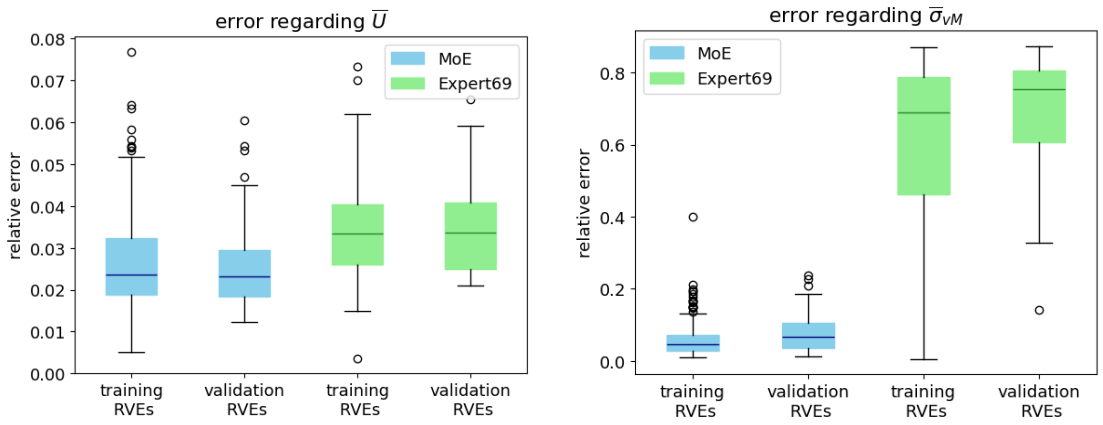


Figure 4.22: Distribution of the macroscopic deviations between the beam frame model and the surrogate models. The box plots in the left image refer to the distribution of the error regarding the displacement variables and the image on the right presents the distribution of the error with regard to the von Mises stress.

box plots indicate that the MoE model performs slightly worse on the validation RVEs compared to the training RVEs. As the training RVEs include the RVE with index 69, for the MLP based on the expert with index 69, we see that the whisker regarding the training RVEs shows a very low minimum value. For the RVE with index 69, the MLP is expected to achieve a low error, as the network is trained to predict the mechanical behavior of the beam frame model.

For a quantitative comparison between the error values resulting from the MoE approach and the error values induced by the expert model with index 69, we also compute the mean over the error data sets. The resulting mean values with regard to the training and validation RVEs are presented in Table 4.2. Generally, the mean error values align with the conclusions drawn from the illustrations in Figure 4.22. With regard to the deformation vector, the MoE model induces slightly lower average error values than the expert with index 69. For the von Mises stress measurements, the difference between the MLP and the MoE approach is much more significant, as the error values resulting from the computations based on the MLP are about ten times larger. Note that the differences between the surrogate models and the beam frame model are generally much greater for the von Mises

	error regarding $\overline{\boldsymbol{U}}$		error regarding $\overline{\boldsymbol{\sigma}}_{vM}$	
	training RVEs	validation RVEs	training RVEs	validation RVEs
MoE model	$2.63e-2$	$2.58e-2$	$5.78e-2$	$8.05e-2$
Expert 69	$3.42e-2$	$3.40e-2$	$6.20e-1$	$6.89e-1$

Table 4.2: Mean deviations from the beam frame solutions for MoE model and single expert. The table shows the mean error values with regard to the displacement variables and the von Mises stress. The error is evaluated based on 250 training and 50 validation RVEs.

stress measurement than for the measurement based on the deformation vector. While the error values with respect to the deformation vectors are very similar, the MoE model shows higher deviations for the validation RVEs with regard to the von Mises stress measurements. It is expected that this difference is mainly caused by the difference in stress magnitude between the beam frame solution and the computations of the MoE model. The stress magnitude resulting from the MoE model for the training RVEs appears to be closer to the beam frame model than the predictions of the MoE model for the validation RVEs.

To conclude the findings with regard to the MoE approach, the results show that the MoE model is able to yield proper predictions for the mechanical behavior of different cellulose aerogels. Especially, with regard to the computation of the von Mises stress measurements, the surrogate model induces higher error values compared to the approach of training an individual MLP for each considered RVE as described in Chapter 3. However, the MoE approach yields a significant benefit with regard to the consideration of new open-porous materials for multiscale methods. As the MLP approach requires an individual training process for each considered material, the MoE model requires only a single training process for being able to predict the mechanical properties for a wide range of open-porous materials.

4.4 Graph Neural Networks for Homogenization

The beam frame RVE as described in Section 2.2 is given by a set of beam elements $\mathcal{E}$ connected at vertices given by the set $\mathcal{V}$. This definition is analog to the description of an undirected graph structure. Each of the vertices translates to a node in the graph and each of the beam elements defines an undirected edge of the graph. Here, the two ends of the beam refer to the respective nodes for the edge. For setting up a machine learning model which is capable of processing RVE data, we can use the graph information given by the beam frame RVE as an input.

With regard to the processing graph data as input, graph neural networks (GNN) are a family of machine learning models well-known for this kind of applications. The GNN architecture for machine learning has first been introduced by Gori in 2005 [60] and has been further elaborated in the ongoing work [133]. Since the development in the beginning of the 21st century, a wide range of different GNN architectures have established as approaches for including graph data in machine learning models. The models are often distinguished by the type of the computed output, as the target output can be computed on the node level as a node feature, on the edge level as an edge feature, or on the graph level as a feature of the whole graph. For example, in the case of graph classification problems, graph level output would be considered.

4.4.1 Graph Neural Network as Machine Learning Model

In this section, we describe the basic components and the functionality of GNN machine learning models. Further information about the models and more detailed descriptions can be found in the literature [133, 150, 155].

Let $\mathcal{G} = (\mathcal{E}, \mathcal{V})$ be a graph with a given set of edges $\mathcal{E}$ and a set of nodes $\mathcal{V}$. For each node $\boldsymbol{v} \in \mathcal{V}$, we consider a number of node features given by h_v and additionally, for each edge $e \in \mathcal{E}$ we consider a number of edge features h_e. Additionally, we define the set of neighbors of $\boldsymbol{v}$ by

$$\mathcal{N}(\boldsymbol{v}) := \{\tilde{\boldsymbol{v}} \in \mathcal{V} \mid \exists\, e \in \mathcal{E} : e = \{\boldsymbol{v}, \tilde{\boldsymbol{v}}\}\}\,.$$

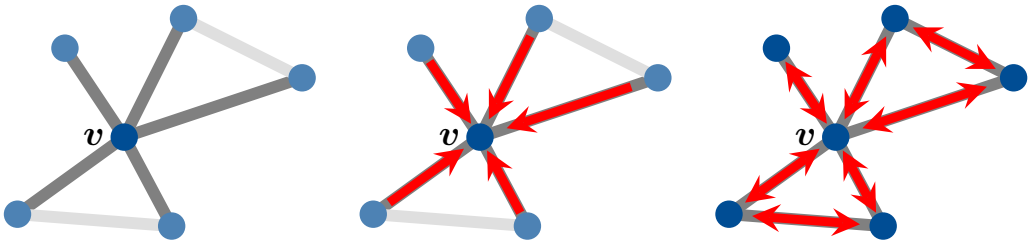


Figure 4.23: Illustration of message passing within a GNN. The image on the left shows the general graph structure with the node v the edges connecting it with its neighboring nodes being highlighted. Middle image illustrates the message passing regarding the node v and right images illustrates the global message passing which considers all graph nodes.

Some of the most fundamental elements of a graph neural network are its message passing layers. Within a GNN, information is transferred over the edges of the graph from each node to its neighboring nodes. To determine how the information is processed from one node to its neighbors, message passing layers are defined. Usually, these functions are defined similarly to MLP models as described in Section 3.1 and include trainable weights. The three images presented in Figure 4.23 illustrate the general principle of message passing. The image on the left shows a simple graph structure with the node $\boldsymbol{v}$ and the edges connecting it with its neighboring nodes highlighted. The middle image shows how the information is transferred in one message passing step from its neighboring nodes to the node $\boldsymbol{v}$. As each message passing step processes the data from all nodes to the respective neighboring nodes concurrently, the right image shows all data transfers executed in a single message passing step.

Let $h_v^0 := h_v$ be the initial node feature of node $\boldsymbol{v} \in \mathcal{V}$. For an arbitrary processing step $t = 0, 1, 2, \ldots$ within the GNN, we define the state of the feature vector of node $\boldsymbol{v} \in \mathcal{V}$ as $h_v^{(t)}$. In line with the terminology introduced in Section 3.1, the state of the feature vector is also referred to as hidden state. A message passing layer defines the computation rule used for computing the update of the feature vector. Even though there are different types of message passing layers, they share the common characteristic of using the current node state $h_v^{(t)}$ and the states of

neighboring nodes $h_{\tilde{v}}^{(t)}$ for $\tilde{\boldsymbol{v}} \in \mathcal{N}(\boldsymbol{v})$ to compute the node update $h_v^{(t+1)}$.

Message passing layers can be distinguished based on three core components, which define the computation rule for the node update. The first component is the message computation function which processes the information, also referred to as message, from a neighboring node. As the message computation function can additionally use the respective edge features $h_{\{\boldsymbol{v},\tilde{\boldsymbol{v}}\}}$ and the hidden state of node $\boldsymbol{v}$, for a given neighboring node $\tilde{\boldsymbol{v}} \in \mathcal{N}(\boldsymbol{v})$, we denote the output of the message computation function as $M(h_v^{(t)}, h_{\tilde{v}}^{(t)}, h_{\{\boldsymbol{v},\tilde{\boldsymbol{v}}\}})$. The second characteristic component of a message passing layer is its aggregation function which gathers the processed messages of all neighboring nodes. We denote the output of the aggregation function as $AGG\left(\{M(h_v^{(t)}, h_{\tilde{v}}^{(t)}, h_{\{\boldsymbol{v},\tilde{\boldsymbol{v}}\}}) \mid \tilde{\boldsymbol{v}} \in \mathcal{N}(\boldsymbol{v})\}\right)$. Typical operations used for defining the aggregation function include mean, sum, max, and the attention-weighted sum. The attention-weighted sum refers to a weighted sum operation with the weights being computed by an attention layer [145]. As this aggregation approach is widely used in graph applications, GNNs which are based on this message passing are also referred to as graph attention networks (GATs) [146]. Some more insight regarding the GAT including the functionality of the attention-weighted sum and applications can be found in [155].

The third component of the message passing layer is the update function, which defines how the update of the hidden state $h_v^{(t+1)}$ is computed based on the current hidden state $h_v^{(t)}$ and the aggregated messages from the neighboring nodes. We denote the output of the update function for the message passing layer as

$$\begin{aligned} h_v^{(t+1)} &= UPDATE\left(h_v^{(t)}, AGG\left(\{M(h_v^{(t)}, h_{\tilde{v}}^{(t)}, h_{\{\boldsymbol{v},\tilde{\boldsymbol{v}}\}}) \mid \tilde{\boldsymbol{v}} \in \mathcal{N}(\boldsymbol{v})\}\right)\right) \\ &=: MP^{(t)}\left(h_v^{(t)}, \{h_{\tilde{v}}^{(t)} \mid \tilde{\boldsymbol{v}} \in \mathcal{N}(\boldsymbol{v})\}, \{h_{\{\boldsymbol{v},\tilde{\boldsymbol{v}}\}} \mid \tilde{\boldsymbol{v}} \in \mathcal{N}(\boldsymbol{v})\}\right) \end{aligned}$$

with $MP^{(t)}$ representing the full message passing function in step t.

One of the simplest examples for a message passing function is given by the

iteration rule

$$h_v^{(t+1)} = \phi\left(W_{self} \cdot h_v^{(t)} + W_{\mathcal{N}} \sum_{\tilde{\boldsymbol{v}} \in \mathcal{N}(\boldsymbol{v})} h_{\tilde{v}}^{(t)}\right) \tag{4.1}$$

regarding the node update $h_v^{(t+1)}$. Here, ϕ describes an activation function as introduced in Section 3.1 and W_{self} and $W_{\mathcal{N}}$ are trainable weight matrices. Hence, this message passing layer shares a very similar structure with the hidden layers described in Section 3.1. For defining this message passing layer, the message computation function is defined as the identity $M(h_v^{(t)}, h_{\tilde{v}}^{(t)}, h_{\{\boldsymbol{v},\tilde{\boldsymbol{v}}\}}) = h_{\tilde{v}}^{(t)}$, the mean is used as aggregation operation, and the update function is defined from the activation function ϕ and the weight matrices W_{self} and $W_{\mathcal{N}}$.

Note that the introduction of message passing layers is focused only on the update regarding the hidden state of a single node $\boldsymbol{v} \in \mathcal{V}$. The reduction of the update process to a single node is supposed to help with the understanding of the functionality of message passing. However, within each message passing step, the hidden states of all graph nodes are updated simultaniously and information is transferred between all neighboring nodes. This process is also illustrated with the image on the right in Figure 4.23. In contrast to the local node update $h_v^{(t+1)}$ which we described with regard to the single node, the global node update is denoted by $H^{(t+1)}$ and combines all node updates in a single vector. For the simple example defined in Equation (4.1), the global node update is given by

$$H^{(t+1)} = \phi\left(W_{global} \cdot H^{(t)}\right)$$

with the global weight matrix W_{global} that is created based on the adjacency of the graph $\mathcal{G}$ and the local weight matrices W_{self} and $W_{\mathcal{N}}$.

In addition to the message passing layers, GNN models often feature pooling layers. These layers aggregate information from multiple nodes and downsample the graph in a way that the number of nodes and edges is reduced. This functionality is very similar to the definition of pooling layers in the context of convolutional neural networks (CNNs) [98]. Here, pooling layers also aggregate and downsample the

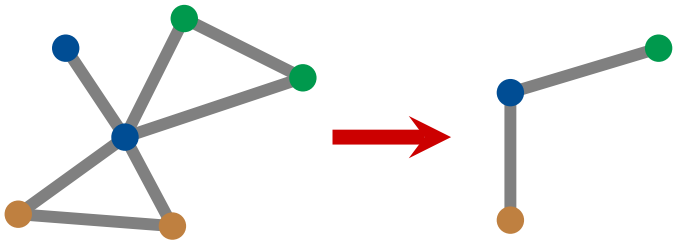


Figure 4.24: Illustration of a pooling layer within a GNN. The image on the left shows the graph structure before the application of the pooling layer. Nodes which are aggregated by the pooling layer appear in the same color. The image on the right shows the graph structure after the pooling layer has been applied.

information of many vectors to fewer vectors. More insight regarding the application of pooling layers in the case of CNNs can be found in [59, chapter 9].

The basic principle of the pooling layer is presented in Figure 4.24. The image on the left shows the graph structure before the pooling layer is applied. The nodes featuring the same color define each a set of nodes which is aggregated by the pooling function. The image on the right presents the graph structure resulting after the pooling layer is applied. With the application of the pooling layer, the total number of nodes within this example graph $\mathcal{G}$ gets reduced from 6 nodes to 3 nodes. For computing the hidden states of the nodes resulting from the pooling layer, an aggregation function needs to be defined. Similar to pooling in the context of CNNs, typical aggregation operations used for the pooling layer include max, min, mean, and sum.

The representation in Figure 4.24 also illustrates how the edges are treated with regard to the pooling operation. Edges connecting nodes that are aggregated by the pooling layer are removed from the graph. Let $\mathcal{V}_1 \subset \mathcal{V}$ and $\mathcal{V}_2 \subset \mathcal{V}$ with $\mathcal{V}_1 \cap \mathcal{V}_2 = \varnothing$ be two distinct sets of nodes. By applying a pooling layer to the graph, the nodes of $\mathcal{V}_1$ are aggregated to the node $\tilde{\boldsymbol{v}}_1$ and the nodes of $\mathcal{V}_2$ are aggregated to the node $\tilde{\boldsymbol{v}}_2$. If it exists an edge between a node of $\mathcal{V}_1$ and a node of $\mathcal{V}_2$, the pooling layer also creates an edge between the aggregated nodes $\tilde{\boldsymbol{v}}_1$ and $\tilde{\boldsymbol{v}}_2$ in the resulting graph structure. If edge features are considered and more than one edge is given between arbitrary nodes of $\mathcal{V}_1$ and $\mathcal{V}_2$, the edge features can also be aggregated to define the

hidden state of the edge connecting $\tilde{\boldsymbol{v}}_1$ and $\tilde{\boldsymbol{v}}_2$. Regarding the aggregation operations for the edges, the same selection of functions is valid as for the aggregation of the node features.

In the context of GNNs, the application of pooling operations is not only beneficial for creating a new graph with a reduced set of nodes and edges. By aggregating the hidden states of all nodes of the graph to a single feature vector, the GNN can produce a global context vector which gathers information of the full graph. This operation is also referred to as global pooling in contrast to the local pooling operation which produces a graph structure of multiple nodes as described above. Global pooling can be regarded as a special case of local pooling, as it aggregates the information of all nodes to a single node. Hence, the resulting graph consists only of the single node with the global feature vector and no edges. Global pooling is usually required in graph classification applications, as the information at all graph nodes needs to be aggregated to a single output.

For training a GNN, the same optimization techniques including backpropagation can be applied as described in Section 3.1. The trainable weights of a GNN are usually given with the definition of the message passing layers. As the depth of the message passing layer is mainly dependent on the size of the hidden node feature vector and not the size of the graph, the number of weights given in a GNN is often relatively small compared to the size of the problem. However, if large graph structures are considered, training a GNN can still be computationally expensive, as the backpropagation algorithm needs to consider the application of the message passing to all graph nodes.

4.4.2 Graph Neural Network as Surrogate Model

The results in Section 4.3.5 emphasize the benefits of the MoE approach for multi-scale simulation regarding open-porous materials. The MoE model allows to compute simulation results of various open-porous material with a single surrogate model. Here, the gating model uses the material parameters I_{RVE} as input to determine the composition of the weighted sum that defines the output of the MoE

model. Analyses of the output of the gating model presented in Section 4.3.4 show that the output of the gating model is mainly depended on the porous fraction. Other material parameters show only small importance for the output components computed by the gating model. Note that the set of material features I_{RVE} contains only statistically properties of the pore size distribution and the composition of the beam elements within the RVE. In order to incorporate the pore structure constellation more strongly into the computations of the gating network, we consider the full beam frame structure of the RVE as input of the gating model. Since the porous fraction is used as an input parameter for defining the cross-sectional diameter of the beam elements, this material feature is incorporated in the properties of the pore structure of an RVE. To process this graph data, the architecture of the gating model must by defined as a GNN. We refer to the gating model that is based on a GNN as gating GNN. In addition to the description of message passing layers, the gating GNN model requires a pooling operation to reduce the graph structure to a single output array that resembles the weights related to the expert models.

For the definition of the graph structure of a given RVE, we consider node features and edge features that are processed by the message passing layers of the gating GNN. The node features are defined by the positional coordinates of the respective nodes. In the three-dimensional case, this means that the feature vector corresponding to the node $\boldsymbol{v} = (x, y, z)^{\mathsf{T}} \in \mathcal{V}$ is defined by

$$h_{\boldsymbol{v}} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathbb{R}^3.$$

For the edges, we consider a more extensive set of features. The feature vector of each edge that corresponds to the beam element $e = (\boldsymbol{v}_i, \boldsymbol{v}_j)$ contains the radius of the circular cross section r_e, the length of the beam $L_e = \|\boldsymbol{v}_j - \boldsymbol{v}_i\|_2$ and the normalized orientation of the beam element within the coordinate system given by

$\frac{|\boldsymbol{v}_j - \boldsymbol{v}_i|}{L_e}$. For the three-dimensional RVE, this results in five edge features

$$h_e = \begin{pmatrix} r_e \\ L_e \\ \frac{|x_j - x_i|}{L_e} \\ \frac{|y_j - y_i|}{L_e} \\ \frac{|z_j - z_i|}{L_e} \end{pmatrix} \in \mathbb{R}^5$$

for each edge $e \in \mathcal{E}$.

We use the TensorFlow-GNN (TF-GNN) [43] framework for the definition of the GNN. This project is based on the TensorFlow [1] software library and relies on many of its functionalities. For the GNN architecture, we use the GATv2 model introduced by Brody in 2022 [17]. The GATv2 model architecture is derivative of the graph attention network (GAT) and addresses its weakness, which is the static attention behavior. In the context of GNNs, static attention refers to the behavior of the model, which exhibits only a slight dependence of the attention values on different compositions neighboring nodes. On the other hand, dynamic attention behavior allows a stronger adjustment of the model's weight with regard to the neighborhood relationships. Deeper analyses of this behavior with a comparison between a standard GAT and the GATv2 approach can be found in [17].

For a single message passing layer of the GATv2 model, the feature update of a node $\boldsymbol{v} \in \mathcal{V}$ is defined by the iteration rule

$$h_v^{(t+1)} = \phi \left(\sum_{\tilde{\boldsymbol{v}} \in \mathcal{N}(\boldsymbol{v})} \alpha_{\{v,\tilde{v}\}} \cdot W_h \cdot h_{\tilde{v}}^{(t)} \right) \tag{4.2}$$

with $\alpha_{\{v,\tilde{v}\}} \in [0, 1]$ referring to the attention value induced by the edge between node $\boldsymbol{v}$ and the neighboring node $\tilde{\boldsymbol{v}}$. The matrix W_h contains trainable weights and ϕ describes an activation function. The attention values $\alpha_{\{v,\tilde{v}\}}$ corresponding to the node $\boldsymbol{v}$ add up to one. This is due to the *softmax* activation function which is used to obtain the vector $\alpha_{\{v,\cdot\}} \in [0, 1]^{|\mathcal{N}(\boldsymbol{v})|}$ that contains the attention values.

Therefore, the length of this vector equals the number of neighboring nodes of $\boldsymbol{v}$. The vector containing the attention values can be expressed as

$$\alpha_{\{v,\cdot\}} = \text{softmax}\left(\{M_v(h_v^{(t)}, h_{\tilde{v}}^{(t)}, h_{\{\boldsymbol{v},\tilde{\boldsymbol{v}}\}}) \mid \tilde{\boldsymbol{v}} \in \mathcal{N}(\boldsymbol{v})\}\right)$$

with the values of the message computation functions $M_v(h_v^{(t)}, h_{\tilde{v}}^{(t)}, h_{\{\boldsymbol{v},\tilde{\boldsymbol{v}}\}}) \in [0,\, 1]$ corresponding to each neighboring node $\tilde{\boldsymbol{v}} \in \mathcal{N}(\boldsymbol{v})$. For the node $\boldsymbol{v} \in \mathcal{V}$ and a given neighboring node $\tilde{\boldsymbol{v}} \in \mathcal{N}(\boldsymbol{v})$, the output of the message computation function is defined by

$$M_v(h_v^{(t)}, h_{\tilde{v}}^{(t)}, h_{\{\boldsymbol{v},\tilde{\boldsymbol{v}}\}}) = \frac{\exp\left(a^{\mathsf{T}} \cdot \text{LeakyReLU}\left(W_M \cdot \text{vec}\left(h_v^{(t)}, h_{\tilde{v}}^{(t)}, h_{\{\boldsymbol{v},\tilde{\boldsymbol{v}}\}}\right)\right)\right)}{\sum\limits_{\tilde{\boldsymbol{v}}' \in \mathcal{N}(\boldsymbol{v})} \exp\left(a^{\mathsf{T}} \cdot \text{LeakyReLU}\left(W_M \cdot \text{vec}\left(h_v^{(t)}, h_{\tilde{v}'}^{(t)}, h_{\{\boldsymbol{v},\tilde{\boldsymbol{v}}'\}}\right)\right)\right)}$$

with a trainable weight vector a and a trainable weight matrix W_M that processes the vectorized representation of the features of both nodes and the edge features. Given an arbitrary scalar value $x \in \mathbb{R}$, the *LeakyReLU* activation function is defined by

$$\text{LeakyReLU}(x) = \begin{cases} x & \text{, if } x \geq 0 \\ m \cdot x & \text{, if } x < 0 \end{cases}$$

for a slope $0 < m < 1$. The activation function equals the *ReLU* function for $x \geq 0$, but allows negative values for a negative input of x. In the TensorFlow software package, the default slope for *LeakyReLU* is set to $m = 0.3$. We apply this default slope value to the activation function in our implementation.

Additionally, we define a pooling operation based on the GATv2 model that reduces the feature vectors of all nodes within the graph $\mathcal{G}$ to a global context vector h_{global}. This operation is similar to the node feature update described in Equation (4.2) and is given by the relation

$$h_{global} = \phi\left(\sum_{\boldsymbol{v} \in \mathcal{V}} \alpha_v \cdot W_h \cdot h_v^{(T)}\right). \tag{4.3}$$

Here, W_h describes a matrix of trainable weights and $h_v^{(T)}$ refers to the node feature vector that results from the final message passing layer of the GNN. The attention values α_v are based on the same computation rules introduced for $\alpha_{\{v,\cdot\}}$. In the case of the node-based attention value α_v, the definition of the message computation function changes to

$$M(h_v^{(T)}) = \frac{\exp\left(a^\mathsf{T} \cdot \text{LeakyReLU}\left(W_M \cdot h_v^{(T)}\right)\right)}{\sum\limits_{\tilde{v} \in \mathcal{V}} \exp\left(a^\mathsf{T} \cdot \text{LeakyReLU}\left(W_M \cdot h_{\tilde{v}}^{(T)}\right)\right)},$$

as all nodes in the graph are considered for the computation and the edge features are neglected. As in the previous definition, the entries of the vector a and the matrix W_M are trainable.

For the definition of the gating GNN, we use a single message passing layer that updates the node feature vectors of the graph as described by Equation (4.2). The resulting node feature vectors are pooled to a global feature vector h_{global} that results from the relation Equation (4.3). This means that the message passing layer and the pooling layer are both based on the GATv2 architecture. For both layers, we use the *gelu* activation function and define the dimension of the resulting feature vectors to equal 32. To obtain a weight vector $w \in \mathbb{R}^n$ that is applicable for an MoE model with $n \in \mathbb{N}$ experts, we apply further processing of the global feature vector h_{global}. We use the feature vector as input to a small MLP consisting of two hidden layers with 64 and 32 neurons, and an output layer of size n with *softmax* activation. The softmax activation function ensures that the sum over the weight values equals one. The definition of the architecture for the gating GNN and the corresponding hyperparameters are the result of manual testing regarding the classification of the expert RVEs. This classification problem is based on a one-hot-encoding of the RVE indices and is described in detail in the following part of this section.

As described in Section 4.3.1, the training of the gating network within an MoE model translates to solving a classification problem. Due to this relationship, we define a classification problem for the beam frame RVEs and train the gating

GNN to identify the RVE structures corresponding to the given expert models. For this classification problem, we define the training data as a list of one-hot-encoded vectors based on the indices of the RVEs. This means that the training data contains n target output vectors which correspond each to a single expert RVE. For the RVE with index i, the one-hot-encoded output vector equals one at index i and the values at all other indices equal zero. The input is defined by the beam structure of the respective RVE. Note that this classification problem is generally much easier to solve with the GNN model than training the gating GNN as a part within the MoE model. However, the training based on the one-hot-encoded representation of the expert RVE gives insight in the GNN's behavior with regard to the given RVEs. Although the model is not expected to behave in the same way for the training within the MoE framework, the classification problem can be used to analyze how different architectures affect the processing of graph data.

With the gating GNN for ten experts, the progression of the training loss values is presented in Figure 4.25. The number of experts is set to ten, with the selection of expert RVEs based on the approximate solution of the max-min dispersion problem as described in Section 4.3.4. This set of expert RVEs has shown proper performance in the case of the MoE model based on an MLP gating model. Since we consider the model to be applied to a classification problem, we define the loss function used for training as the categorical cross-entropy. This loss function is a common choice with regard to classification problems with multiple classes. For a given prediction $w \in \mathbb{R}^n$ and the one-hot representation $w^* \in \mathbb{R}^n$ that defines the corresponding target output, the categorical cross-entropy loss is defined by the expression

$$\ell\left(w, w^*\right) = -\sum_{i=0}^{n} w_i \cdot \log(w_i^*).$$

Note that in the exact implementation of the loss term, a tolerance in the range of machine accuracy is added to the entries of w^* that are zero to prevent errors in the computation of the logarithm. Additionally, the same tolerance is deducted from the entry of w^* that equals one. In addition to the loss term, the accuracy of the model

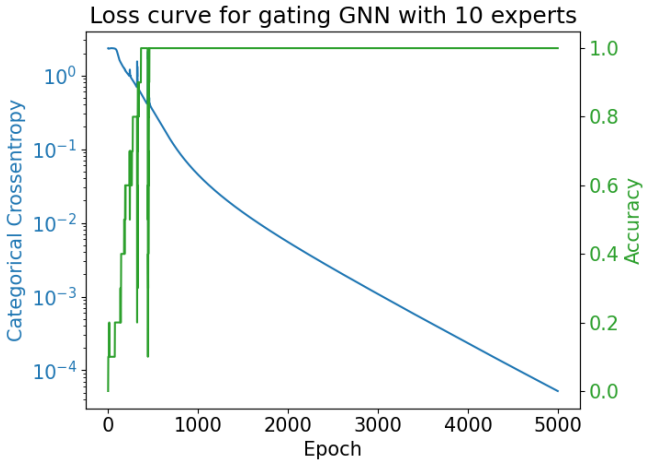


Figure 4.25: Loss curve for training the gating GNN with ten experts on the one-hot encoded data. The blue curve presents the progression of the categorical cross entropy and the green curve refers to the accuracy measurement.

is evaluated in each training epoch. This metric evaluates the ratio of entries of w and w^* that are identical. In our considered classification problem, this means that the accuracy measurement is zero, if none of the expert RVEs is identified correctly, and it equals 1, if all expert RVEs are identified correctly by the gating GNN. The model is trained for 5,000 epochs with Adam optimizer and a fixed learning rate of $1e-3$. We use this high number of training epochs, due to the small training data set. The results of the training process presented in Figure 4.25 show that the categorical cross-entropy loss is significantly reduced over the scope of 5,000 epochs. Based on the visualization with a logarithmic scale, the reduction of the loss appears approximately exponential. For the accuracy measurement that is presented by the green curve in Figure 4.25, the value rises fast within the first approximately 500 training epochs. Apart from a few drops in the curve, the accuracy then remains constant at a value of one. Further analyses of the output of the gating GNN show that the model identifies each of the expert RVEs correctly based on the graph structure with a confidence value that rounded to the fourth decimal place is equal to one. The results show that the gating GNN model is generally able to process the graph structure of the given RVEs and to identify differences between the beam frame structures.

To integrate the gating GNN into the MoE framework, we consider the same general structure as presented in Figure 4.6. In the case of the gating GNN, the gating NN is replaced by the GNN that uses the graph structure $\mathcal{G}_{RVE}$ as input to compute the weight vector $w \in \mathbb{R}^n$. We use the coherent training approach for training the gating GNN. Here, the weights of the pretrained expert models are fixed and only the weights of the gating model are optimized in the training process. The data that is used for optimization of the weights of the gating GNN is based on the RVEs described in Section 4.1. Similar to the previous MoE approach based on the MLP gating model, we consider a training data set and two validation data sets. Since the full graph structure including node feature vectors and edge feature vectors needs to be stored for each data point, the memory requirements increase significantly with an increasing number of input graphs in the data sets. With a number of beam elements and node coordinates in the thousands, the memory requirement is many times greater than that of the MoE model with an MLP gating network. To reduce the memory demand and to allow training of the gating GNN that can be completed with reasonable computation times, we restrict the number of data points considered for the data sets. This means that the training data set contains 90,000 data points that are computed from deformation data based on 250 training RVEs. The validation data set that differentiates in the deformation gradient is computed from the same training RVEs and contains 10,000 data points. The val-RVE data set contains 20,000 data points that are computed based on 50 validation RVEs.

For training the gating GNN within the MoE framework, we use Adam optimizer with a relatively high learning rate of $5e-2$ that is constant over the scope of training. We use a batch size 256 to partition the data sets. The results of the training procedure are presented in Figure 4.26. The loss curves in the image for all considered data sets appear constant over the scope of 300 training epochs. A deeper analysis of the loss progression shows that the loss values change only with a magnitude of about $5e-5$ over all training epochs. This behavior indicates that the backpropagation algorithm is not able to properly adjust the weights of the gating GNN based on the mean squared error (MSE) loss induced by the homogenized

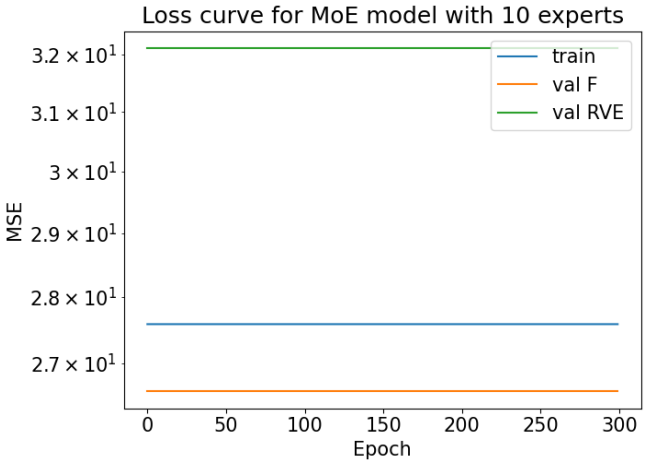


Figure 4.26: Loss curve for the training of the gating GNN integrated in the MoE framework. The MoE model includes ten expert models.

Piola-Kirchhoff stress tensor. Despite the relative high learning rate, the adjustment in the weight values is not great enough to cause a significant change in the output of the MoE model.

The results of training the gating GNN on the classification problem based on the one-hot representation show that the model architecture generally allows a processing of the RVEs in a way that the graph structures are differentiated from each other. Despite the decent training behavior for the classification problem, integrating the gating GNN into the MoE framework does not yield satisfying training results. The training results indicate a too weak gradient signal for the gating GNN within the MoE model that is induced by the backpropagation. Different explanations are plausible for this weak training performance, such as too similar expert outputs. However, based on the behavior, the MoE approach does not benefit from the processing of the graph structure defined by the RVE. To identify the factors that cause this behavior and to describe possible approaches to resolve the issues, would require further analyses of gating GNNs performance. Our research regarding this surrogate model includes testing with different GNN architectures and with different training approaches. However, so far, the considered trials did not yield significant improvements with regard to the performance of the MoE model. Fur-

ther analyses and the development of an improved gating GNN model could be an interesting topic for future work.

Chapter 5

High-Performance Computing and Scalability

In each iteration of a macroscopic solver, the FE^2 method requires the solution of a microscopic problem for each macroscopic Gaussian quadrature point. Especially for geometries with a fine macroscopic discretization, the computational effort of computing the microscopic problems becomes significant. In Chapter 3, we already discussed the use of machine learning-based surrogate models to reduce the computational cost of each microscopic problem. However, even with low cost surrogate models, the sequential computation time for solving all microscopic problems in each iteration increases linearly with the macroscopic degrees of freedom.

Besides the use of surrogate models, the computational effort can be reduced by parallelizing the computation on the microscopic scale. Here, the term parallelization refers to the division of a computation process into small independent parts that can be carried out simultaneously on multiple CPU (short for central processing unit) cores. For the FE^2 approach, each homogenized stress tensor resulting from a microscopic problem depends only on the respective deformation gradient at the Gaussian quadrature point. This means that in one iteration of the macroscopic solver, all microscopic problems can be solved independently of each other and therefore perfectly parallelized. Parallelization can be applied to any compu-

tation method at the microscopic level. This means that both surrogate models and beam frame models can be considered for the parallel implementation of the multiscale method.

The potential of the FE^2 method for parallelization is well-known and has already been successfully implemented in projects like EXXASTEEL [84, 87]. The FE2TI software [85, 86] by Klawonn, Lanser, and Rheinbach used for this project demonstrated strong scaling behavior with regard to the parallel execution on large scale compute clusters [81].

With regard to the parallelization of the multiscale simulation method, Section 5.1 describes the general software tools and numerical methods that we utilize for the parallel implementation. For the evaluation of the performance of this implementation, Section 5.2 describes the results of a strong scaling test which compares the runtimes of parallel computations conducted on different numbers of compute cores.

5.1 Implementation

For the implementation of the FE^2 method as a parallel algorithm, we use C code based on the PETSc (short for Portable, Extensible Toolkit for Scientific Computation) framework [6, 5]. PETSc is an open-source software which was built to offer efficient computational solutions for scientific applications. It features many implementations of parallel matrix-matrix and matrix-vector operations which are crucial for solving partial differential equations. It also features a wide range of parallel preconditioners that can be used for improving the robustness of iterative linear solvers [130, chapter 9]. For parallelization, PETSc employs the Message Passing Interface (MPI) [107] standard for communication between parallel processes.

The MPI standard allows communication between multiple processes with distributed memory. This means that each process has only access to its own memory and if data is required from a different rank, communication is needed to transfer the data. In the MPI framework, the compute processes are referred to as MPI

ranks. For our parallel computations based on the FE^2 method, the number of MPI ranks is equal to the physical CPU cores within the respective system. Here, every physical core is assigned to a single MPI rank. Due to this relationship, we use the terms MPI rank and CPU core interchangeably in the following explanations.

The parallel implementation of the FE^2 method includes different microscopic models, which are described in the previous sections. In addition to the linear beam frame model, the evaluation of various surrogate models can be used on the microscopic scale. As described in Section 3.1, the evaluation of a multi-layer perceptron (MLP) mainly requires the computation of matrix-vector products. For carrying out the computations of the FE^2 method on multiple CPU cores, each process solves a subset of the microscopic problems. Here, the operations required to compute the homogenized stress tensor that corresponds to a single Gaussian quadrature point are sequentially executed on a single compute core. In the case of an MLP as surrogate model, this requires sequential matrix-vector product computations. For matrix operations that are conducted on a single node, which include the operations for evaluation of the surrogate models, we use the Basic Linear Algebra Subprograms (BLAS) [11] routines that are provided by the Intel Math Kernel Library (Intel MKL) [76, chapter 11]. As the kernel of an MLP is usually dense, BLAS routines are expected to achieve fast computation of the output of a given surrogate model. In the case of the Mixture of Experts (MoE) model, we use the same routines for computing the output of the expert models and for evaluating the gating network. To reduce the computation time, for each evaluation of the MoE model, we compute only the output of expert models with a corresponding weight value that is substantially greater than zero. For an MoE model with $n \in \mathbb{N}$ expert NNs, we define the threshold for the weight values to be $\frac{0.1}{n}$.

5.2 Strong Parallel Scalability

In computer science, the term strong scaling refers to an algorithm's behavior regarding an increasing number of processing cores. For the strong scaling test, we

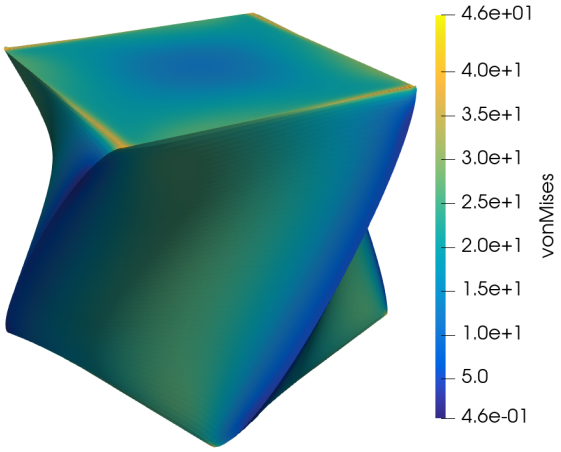


Figure 5.1: Results for the torsion of the cube geometry computed with the parallel implementation of the FE^2 approach. The coloring of the geometry refers to the von Mises stress resulting from the deformation.

consider a fixed problem size and compute a solution with different numbers of MPI ranks. In our case, the problem is defined as the torsion of a cube geometry in z-direction. Dirichlet boundary conditions are applied to the faces of the body pointing towards the axis of rotation. We consider a discretization of the cube with 1,310,720 P1 elements. This results in 823,875 degrees of freedom for the macroscopic problem. We assign this rather large problem for the strong scaling test because larger problems yield more potential for parallelization. The macroscopic result of the described problem is shown in Figure 5.1.

As described in Section 2.3.2, for the FE^2 method, the macroscopic boundary value problem is solved using a non-linear solver. In Section 2.3.2, we describe the sequential Newton's method and quasi-Newton methods. For the parallel implementation, we consider a Newton-Krylov method [88] for minimizing the macroscopic residual. This method is based on Newton's method and generally uses the same iteration rule. In each iteration, macroscopic update $\overline{\boldsymbol{S}}_k$ is computed as the solution of the system of linear equations

$$\overline{DR}(\overline{\boldsymbol{U}}_k) \cdot \overline{\boldsymbol{S}}_k = -\overline{R}(\overline{\boldsymbol{U}}_k).$$

In the Newton-Krylov method, the linear system is not solved directly. Instead, a Krylov subspace method [94] is used to approximate the solution to the system of linear equations. In general, for a given $m \in \mathbb{N}$, a Krylov subspace method computes the approximation of $\overline{\boldsymbol{S}}_k$ within the span of the images of $-\overline{R}(\overline{\boldsymbol{U}}_k)$ under the first m powers of $\overline{DR}(\overline{\boldsymbol{U}}_k)$. This span defines the subspace

$$\begin{aligned}&\mathcal{K}_m(\overline{DR}(\overline{\boldsymbol{U}}_k), -\overline{R}(\overline{\boldsymbol{U}}_k)) \\ &= \operatorname{span}\left(-\overline{R}(\overline{\boldsymbol{U}}_k), -\overline{DR}(\overline{\boldsymbol{U}}_k) \cdot \overline{R}(\overline{\boldsymbol{U}}_k), \ldots, -\overline{DR}(\overline{\boldsymbol{U}}_k)^{m-1} \cdot \overline{R}(\overline{\boldsymbol{U}}_k)\right).\end{aligned}$$

The category of Krylov subspace methods features a number of algorithms for approximating the solution of a system of linear equations. This category includes the conjugate gradient (CG) method [66] and the generalized minimum residual (GMRES) method [129]. Deeper insight with regard to Krylov Subspace methods can be found in [130, chapter 6].

For the strong scaling test, we use the parallel implementation of the GMRES solver [129] which is included in the PETSc framework. The main advantage of this solver is that it is highly parallel and it is expected to show good scaling behavior for an increased number of compute cores. Therefore, it is well suited for the application in our parallel implementation. The performance of a Krylov subspace method like GMRES can often be improved with the use of preconditioners. Therefore, we use the SOR method [153, chapter 3] as a preconditioner for reducing the number of GMRES steps in each Newton iteration. The tolerance for the Newton-Krylov method in the strong scaling test is set to $1e-6$ for each number of used CPU cores.

On the microscopic scale of our FE^2 method, we consider the MLP approach for computing the homogenized stresses. The considered MLP has 3 hidden layers with 256 neurons in each layer. *GELU* is used for the activation functions of the hidden layers and the output layer uses linear activation. Each MPI rank holds its own copy of the MLP in its memory and the number of evaluations is evenly split between the ranks to equally distribute the compute load in each iteration.

The simulations for the strong scaling test are executed on a single node of the

compute cluster *Fritz* at Friedrich-Alexander-Universität Erlangen-Nürnberg [49]. Each node features two Intel Xeon Platinum 8360Y central processing units (CPUs) with 36 cores each. For each compute node, this results in a total of 72 cores and for the scaling test, we solve the described problem with up to 64 of these cores in parallel.

The graphs in Figure 5.2 show the results of the strong scaling test. Generally, for measuring computation times, we distinguish between the computations on the macroscopic and on the microscopic scale. The blue line refers to the computation time required for computing the NN evaluations on the microscopic scale. For solving the problem described above, in each iteration of Newton's method, the macroscopic residual vector $\overline{R}$ and the tangential matrix $\overline{DR}$ need to be assembled. Both require the evaluation of the NN or the computation of its derivative for given macroscopic deformation gradients. We sum up the computation times for the assembling of $\overline{R}$ and $\overline{DR}$ over all Newton iterations until the convergence criterion is satisfied. The data points on the blue graph of Figure 5.2 show the resulting overall computation times induced by the microscopic computations as described. The red line presents the computation times required for solving the system of linear equations in parallel for the Newton-Krylov method in each iteration. Similarly, the data points result from the summed computation times over all Newton iterations. This includes the computation times required for the iterative GMRES solver and the corresponding SOR preconditioner. As the implementation of the GMRES solver in PETSc takes advantage of the parallelization potential of the algorithm, the computation times for the GMRES method decrease with an increasing number of MPI ranks.

Furthermore, the theoretical slope for an optimal parallelization is indicated by the black dotted line. With a slope equal to $-\frac{1}{\#\text{MPI ranks}}$, the inverse proportional function appears linear in the log-log-scale of Figure 5.2. It represents the theoretical behavior of a perfectly parallelized algorithm with the workload fully balanced over all MPI ranks.

The lines regarding the NN evaluations and the GMRES linear solver both show a strong descent for an increasing number of MPI ranks. For the computation times

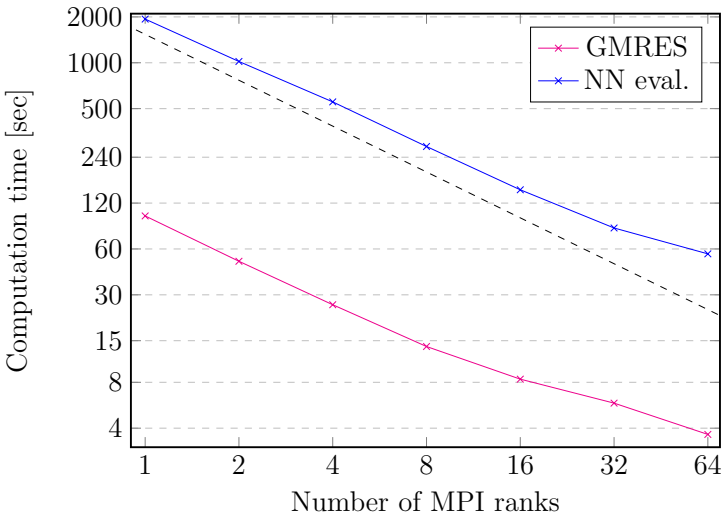


Figure 5.2: Strong scaling results for the torsion of a cube. The graph shows the computation times for the NN evaluations on the microscopic scale in blue and for the preconditioned GMRES solver for the Newton's method on the macroscopic scale in red.

up to 16 MPI ranks, both computation times scale almost perfectly with the slope of the lines close to the theoretical dotted line. For 32 and 64 MPI ranks, a slight flattening of both curves can be observed. This smaller decrease in the computation times for the high numbers of MPI ranks could be due to the sharing of resources on one compute node. Resources such as the L3 cache are shared by multiple cores within a single computing node. As the number of processes increases, the performance of each core can decrease if the shared resource becomes a bottleneck. Overall, however, the results of the strong scaling test show that the solver behaves about as expected, with computation times cut in half for double the number of MPI ranks.

In addition to the considered parallel implementation of the multiscale approach for multiple CPU cores, it would be possible to consider a parallel implementation of the described FE^2 framework for graphics processing units (GPUs). Due to their high number of parallel compute cores, GPUs are well known for fast computation of matrix multiplications [104]. Due to this property, this hardware can yield sig-

nificant benefits for the evaluation of NN models, which is mainly based on matrix multiplications. In the case of surrogate based microscopic models, the computation time on the microscopic scale could be reduced with the use of GPUs. As a non-linear solver, the L-BFGS method shows decent performance with regard to parallel GPU implementations [41]. As the computation of its iterative update is mainly based on the computation of vector operations, the method is well-suited for GPU implementations. With the L-BFGS method as macroscopic solver, an implementation of the multiscale method that is primarily conducted on a GPU, would be conceivable. The exact impact of such an implementation on the required computation time would have to be investigated. Developing a proper implementation and analyzing the resulting behavior of the method could be the topic of a follow-up project.

Chapter 6

Comparison of Simulation and Experimental Results

One of the central goals of this work is the development of numerical models that can be applied for the development of new nanoporous materials. Necessarily, it is in the nature of numerical methods to idealize real-world conditions using modeling techniques. With a given model, it is possible to find an approximation for the realistic behavior of the material. Reducing the discrepancy between simulation and experimental behavior is therefore a key objective in model development. Despite the importance of such validation based on experimental results, this subject is often not sufficiently covered. Since collecting experimental data is often very difficult and expensive, the database that could be used for a comparison is usually sparse. In addition, accurately replicating the conditions of a mechanical experiment with a simulation setup can also become complicated.

In this chapter, we describe experimental tests that investigate the mechanical behavior of aerogel materials. To replicate the experiments, we apply the multi-scale simulation method described in Chapter 2. Setting up the simulations requires modeling of the material on both the macroscopic and microscopic scales. On the macroscopic scale, the specimen geometry must be represented as a three-dimensional model, which can be discretized with finite elements. Additionally,

accurate definition of the macroscopic boundary conditions is required to reproduce the imposed deformation in the experiments. On the microscopic scale, the considered RVE is expected to represent the nanostructure of the respective material in a meaningful way. To obtain the RVE for a given aerogel, we use the experimentally measured pore size distribution and porous fraction to generate the beam frame model as described in Section 2.2.

We consider two fundamentally different experimental setups in the scope of this chapter. Section 6.1 describes the tension of hole plates that are made from cellulose aerogel. The experimental data resulting from two different aerogel materials is compared to the respective simulation results. The second experimental setup in Section 6.2 deals with the torsion of a aerogel plate. The three-dimensional deformation data that results from this experiment is compared to the corresponding simulation results. For both experimental setups, we evaluate the deviation between the experimental data and the simulations.

The experimental data used in this chapter is provided courtesy of the colleagues from the Institute for Frontier Materials on Earth and in Space at the German Aerospace Center (DLR) in Cologne. At DLR, the required aerogel samples were synthesized, characterized with respect to the properties of the respective nanostructures, and tested with mechanical experiments. Special mention is due to Bruno Gonzales, who synthesized the cellulose aerogel samples and carried out the measurements of pore size distributions and porous fractions. In addition, thanks to the work of Max Zinke, Lennart Barth, Eric Dietrich, and Dr. Eric Breitbarth, the mechanical tests could be performed and evaluated using digital image correlation.

6.1 Tension of Hole Plates from Different Cellulose Aerogels

For evaluating the behavior of the multiscale approach for different nanostructured materials, experimental tests with different cellulose aerogels are considered. In general, cellulose aerogels typically feature a fibrillar nanostructure that is well

suited for modeling with the beam frame RVE. Despite the general structure being similar, different pore size distributions and different porous fractions are considered as a basis for the creation of the RVE. As it is given that the nanostructure of an aerogel has an important influence on its mechanical behavior, we investigate the effect of the pore structure on the multiscale simulations.

Video files presenting the experimental and simulation results are provided in a Zenodo database which can be accessed via [101]. The videos cover side-by-side comparisons of the experimental and simulation results. Videos showing the results of additional analyses of the simulation data with regard to stress measurement and the computed deviation from the experimental results are included in the database.

6.1.1 Experimental Setup

The most important steps for the synthesis of cellulose aerogels are the sol-gel process and subsequent supercritical drying. The exact execution of these procedures is crucial for the pore size composition within the resulting aerogel. Regarding the sol-gel chemistry, the ratio of the starting materials for the solution is also an important input factor. Especially, the porous fraction is strongly affected by the amount of cellulose used for mixing the starting solution. As known from the insight of experimental testing, the amount of cellulose within the starting solution and the porous fraction of the resulting aerogel have a strong negative correlation.

Regarding the samples of cellulose aerogel considered in this section, the materials differ from each other in the conditions used for the synthesis. While all other synthesis conditions are kept constant, the amount of cellulose used in the starting solution of the sol-gel process is varied. Three different synthesis approaches are considered with three percent, six percent, and nine percent of cellulose used in the solution. As mentioned above, the different ratios of cellulose have a significant influence on the porous fraction of the resulting aerogel and on the pore size distribution.

For determining the pore size distribution of the aerogel materials, gas adsorption methods are used in combination with the Barrett-Joyner-Halenda (BJH)

model [7]. The resulting pore size distributions for the aerogels synthesized with different weight percentages of cellulose are presented in Figure 6.1. For the illustration, the raw curves of the data are slightly smoothed so that the differences between the data distributions can be interpreted more easily by the reader. However, regarding the utilization of the data for creating the RVEs, no processing has been applied to the pore size data sets. At first glance, the three distributions behave very similarly with the majority of the pore sizes in the range between 20 and 80 nanometers. However, the distributions are still distinct from one another and the frequencies of the graphs differ in some regions. The blue curve in Figure 6.1 features a lower frequency in the range between 20 and 40 nanometers in comparison to the graphs referring to the aerogels with six and nine percent cellulose. Regarding the porous fraction of the cellulose aerogel samples, a more significant difference can be observed. The values of the porous fractions are provided courtesy of the Institute for Frontier Materials on Earth and in Space at the German Aerospace Center (DLR). For the three samples, the porous fractions are given by:

- 3% cellulose aerogel: 0.9464
- 6% cellulose aerogel: 0.9011
- 9% cellulose aerogel: 0.8624.

The values confirm the inverse correlation between the weight percentage of cellulose used for the sol-gel process and the porous fraction.

For the mechanical testing of the aerogel samples, the tension of a plate with a single hole in the center is considered. For each of the cellulose aerogels, a hole plate sample is created purely from the open-porous material. The samples are then individually tested. To apply tensile loading to the aerogel hole plates, the sample is clamped at both ends and the clamps are slowly pulled apart. During the test, the deformation of the samples is recorded with cameras and with the use of digital image correlation (DIC) [117], the deformations are quantified. The DIC software tracks multiple randomly distributed points that are applied through spray paint on the surface of the sample. The software translates the position of the points into

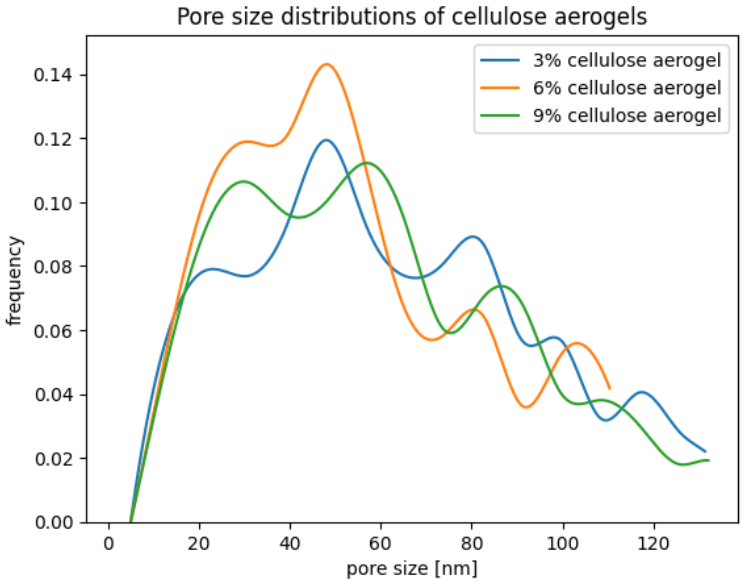


Figure 6.1: Pore size distributions of the cellulose aerogel samples produced with three, six, and nine percent cellulose used in the starting solution. The data used for creating the graphs is provided courtesy of the Institute for Frontier Materials on Earth and in Space at the German Aerospace Center (DLR).

a given coordinate system and computes the deformation for each considered time step of the experiment.

Since experimental testing of aerogels is difficult and time-consuming, only the experiments for the cellulose aerogel samples produced from three percent and six percent cellulose in the starting solution were carried out. Except for the pore size distribution and porous fraction, no experimental results are obtained from the sample produced with nine percent cellulose used in the sol-gel process.

The full plate of the three percent cellulose sample measures 60 mm in length, 20 mm in width, and 2 mm in thickness. The hole in the center of the sample has a diameter of 13 mm. As the clamps used for locking the sample into position and applying the considered deformation cover 15 mm at each end of the sample, only a total length of 30 mm of the sample is tracked by the DIC software. Regarding the deformation of the sample, this is also the only part of the sample that is interesting for the comparison with the simulation. The measurement points of the sample used

by the DIC software for tracking the deformation are presented in Figure 6.2 on the left. For this sample, a total of 4,761 evaluation points are considered. As the measurement points are only given on one surface of the sample, the experimental results computed by the DIC software are only two-dimensional. However, this is not a crucial restriction of the experiment, as the samples are relatively thin and it is expected that deformation is constant over the thickness of the material.

For the aerogel sample produced from a starting solution with six weight percent cellulose, the dimensions of the sample are similar to the three percent case but the size of the hole is changed. Analog to the three percent cellulose aerogel, the sample measures 60 mm in length, 20 mm in width, and 2 mm in thickness. However, the hole in the center of the sample is slightly smaller and measures 11 mm in diameter. With 15 mm at each end of the aerogel covered by the clamps, the length of the sample which is tracked by the DIC software reduces to 30 mm. The measurement points considered for the DIC are presented in the right image of Figure 6.2. As in the case of the three percent cellulose aerogel, the measurement points are only considered on one surface of the sample, which results in a two-dimensional DIC data set. A total of 4,687 evaluation points are tracked by the DIC software.

As cellulose aerogels are very sensitive to tensile forces and break easily, the deformation in the experiments is only considered up to a tensile strain of about 1.8% relative to the sample length. The deformation was applied to the three percent and six percent cellulose aerogels in 646 and 659 load steps, respectively. During this deformation process, 36 images have been taken for the three percent cellulose aerogel and 44 images for the six percent cellulose aerogel. Each of the images has been evaluated with the DIC software.

In addition to the deformation data, the overall stress resulting from the deformation has been measured through a force sensor connected with the clamps. As the samples are only stretched in one dimension, the scalar stress measurement refers to the direction of the tensile force. For every deformation step, the strain and the resulting stress measurement are shown in Figure 6.3. The green graph refers to the sample produced with three percent cellulose used in the starting solution and the blue graph refers to the sample produced with six percent cellulose in the

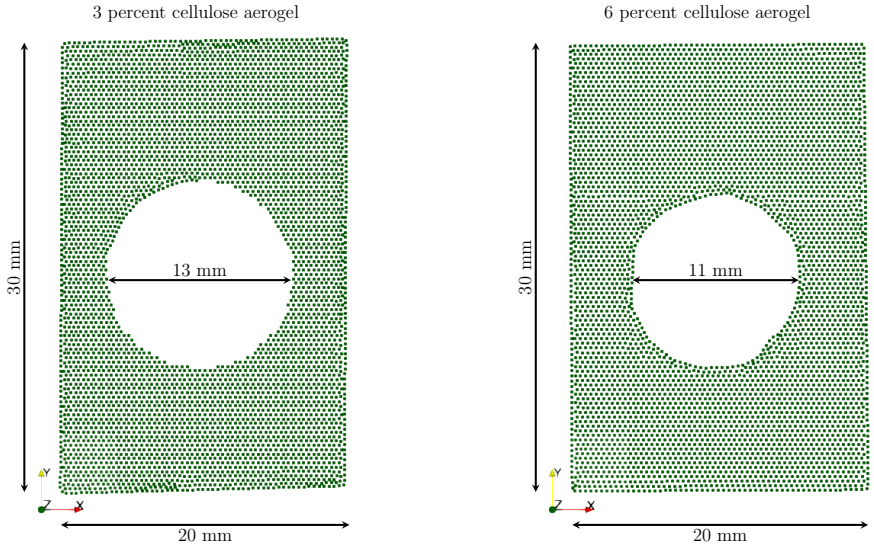


Figure 6.2: Evaluation points of the DIC software for tracking the deformation of the cellulose aerogel samples. The evaluation points of the sample produced with three weight percent cellulose are shown on the left, and those of the sample produced with six weight percent cellulose are shown on the right. The data used for creating the figures is provided courtesy of the Institute for Frontier Materials on Earth and in Space at the German Aerospace Center (DLR).

starting solution. Regarding the stress-strain graphs, both samples show a similar stress-strain behavior with the significant difference that the three percent cellulose aerogel sample shows much lower stress values. Both graphs exhibit two significant dips regarding the stress value. For both materials, the dips appear at about 1.3% and 1.6% strain. The sudden drops in the stress values are caused by failure of the materials. As observed during the experiments, both samples first fail at one of the sides next to the center hole. The weakest parts of the samples are given close to the holes, as the material is thin in these sections. The second drop in stress is then caused by the failure on the other side of the hole. At this point of the experiment, the sample is torn into two separated pieces and the stress value drops to zero.

Regarding the experiment for the sample produced with three percent cellulose, the deformation computed by the DIC software is presented in Figure 6.4. The figure shows two deformation states captured by the software during the experiment. In

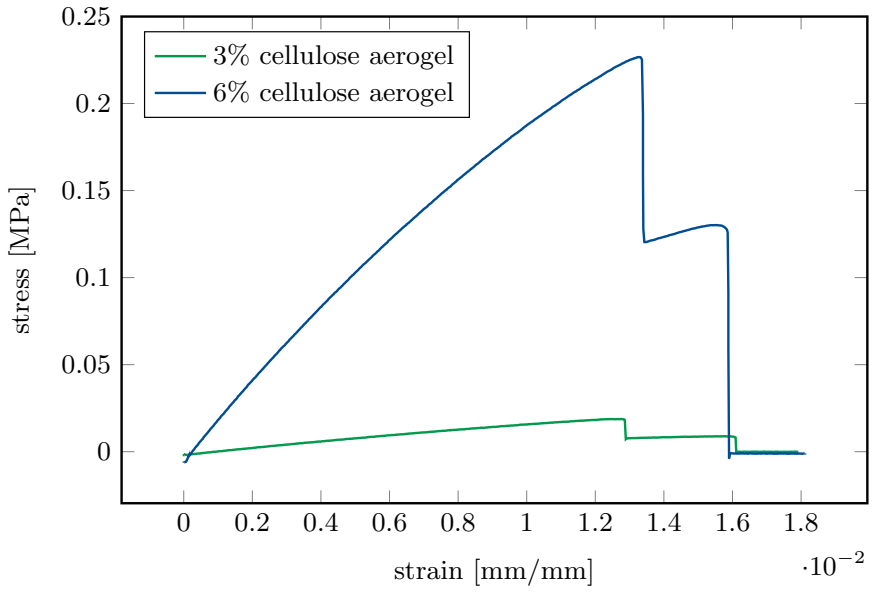


Figure 6.3: Stress-strain relationship resulting from tension of the cellulose aerogel samples. Green graph refers to the sample produced from three percent cellulose and blue graph refers to the sample produced from six percent cellulose. The stress data is given in megapascal (MPa). The data used for creating the graphs is provided courtesy of the Institute for Frontier Materials on Earth and in Space at the German Aerospace Center (DLR).

Figure 6.4, the image on the left shows the deformation state computed from the DIC image number 34. This was the deformation state of the sample right before the material failed on the right side next to the hole. The dots again represent the measurement points considered for the DIC and the color of the dots illustrates the L^2-norm of the deformation vector at the respective measurement point. The illustration indicates that before the failure of the material, the sample shows some slight necking behavior close to the hole. As the tension becomes stronger, the middle section of the material becomes thinner. On the right side of Figure 6.4, the deformation state of the sample right after the failure of the material is presented. The state is captured by the DIC image number 36. The image shows a visible crack on the right side of the sample next to the hole. As a result of the failure, the right side of the sample shows significantly greater deformation than the left side.

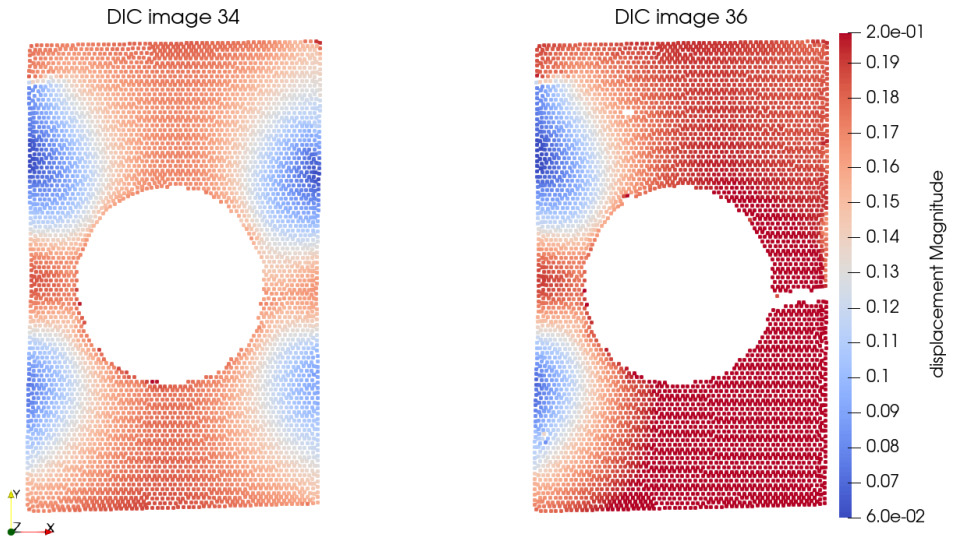


Figure 6.4: DIC results for the three percent cellulose aerogel sample. The image on the left shows the deformation state of the sample right before the failure of the material at DIC image number 34. The image on the right shows the deformation state of the material after the failure at DIC image number 36. The colors of the dots represent the norm of the deformation vector at the respective measurement point in millimeters (mm). The data used for creating the figures is provided courtesy of the Institute for Frontier Materials on Earth and in Space at the German Aerospace Center (DLR).

Similar to the illustration of deformation regarding the three percent cellulose aerogel given in Figure 6.4, the images in Figure 6.5 show two deformation states of the sample produced from six percent cellulose in the sol-gel process. In Figure 6.5, the image on the left shows the deformation state of the sample right before first failure of the material. The deformation state is captured by the DIC software with image 36. As described for the case regarding the three percent cellulose aerogel, the dots shown in the image represent the measurement points with the color of the dots illustrating the L^2-norm of the deformation vector at the respective measurement point. The colors in the image indicate that the deformation of the sample is largest at the two ends of the sample where the clamps are positioned that apply the tension. Additionally, the colors show that the deformation is also

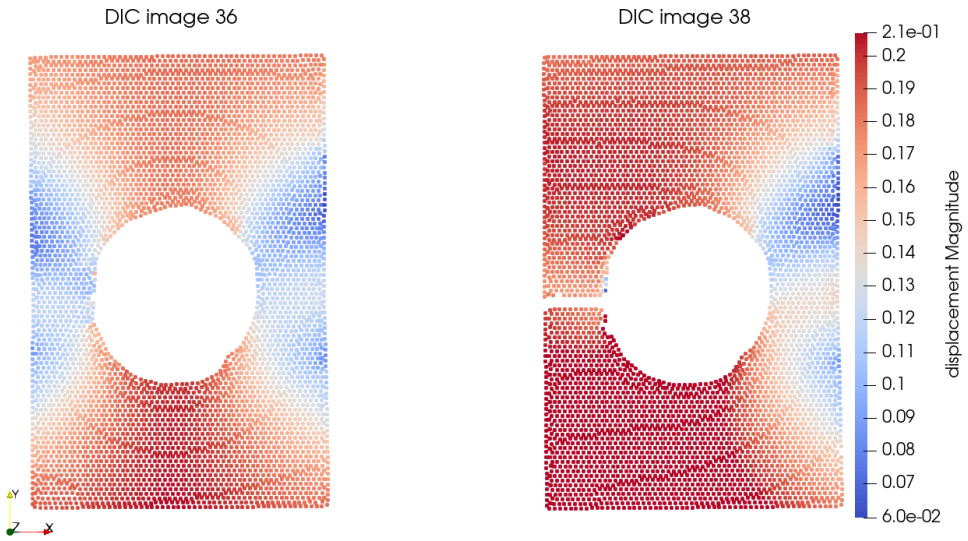


Figure 6.5: DIC results for the six percent cellulose aerogel sample. The image on the left shows the deformation state of the sample right before the failure of the material at DIC image number 36. The image on the right shows the deformation state of the material after the failure at DIC image number 38. The colors of the dots represent the norm of the deformation vector at the respective measurement point in millimeters (mm). The data used for creating the figures is provided courtesy of the Institute for Frontier Materials on Earth and in Space at the German Aerospace Center (DLR).

greatest along the central y-axis, which divides the sample vertically. In the image on the right of Figure 6.5, the deformation state of the sample after failure of the material is presented. The deformation state is captured with DIC image number 38. It shows a significant crack in the sample on the left side next to the hole. As the crack occurred directly at the hole in both considered samples, it can be concluded that this is the weak point of the materials.

6.1.2 Simulation Setup

For simulating the tension test of the cellulose aerogels, we consider the FE^2 approach with the beam frame model on the microscopic scale as described in Chapter 2. This approach is expected to suit the nanoporous structure of the cellulose

aerogel samples. Generally, cellulose aerogels are a suitable test material for the application of the described multiscale method, as the fibrillar nanostructure formed in the synthesis of cellulose aerogels suits the modeling with beam elements. As described in Chapter 2, for setting up the FE^2 approach, the macroscopic and microscopic problems need to be defined.

On the macroscopic scale, to replicate the experimental results with a simulation, we need to create digital copies of the experimental samples that can be discretized with finite elements. For modeling the geometries, we consider the same dimensions as the original samples and only the part of the aerogels is considered which is between the two clamps, as this is also the relevant section for the simulation. In contrast to the experimental results obtained from the DIC software, the digital samples considered for the simulation are three-dimensional. The three-dimensional simulation, which also considers the thickness of the samples is expected to yield more accurate results. The software Gmsh [55] is used for modeling the geometries and meshing. Tetrahedral P1 elements are used for discretization of the finite element mesh. With the modeling of the digital geometries as described, the finite element mesh based on the three percent cellulose aerogel samples features 5,763 P1 elements with 1,768 nodes and the mesh based on the six percent cellulose aerogel features 6,236 P1 elements with 1,888 nodes. We consider the momentum balance equation with respect to the first Piola-Kirchhoff stress tensor as a basis for describing the mechanical behavior on the macroscopic scale. As macroscopic solver, we apply the parallel implementation of the Newton-Krylov method with a tolerance of $1e-6$ as described in Chapter 5.

As shown in Figure 6.3, the strain applied to the experimental samples is known for each step of the experiment. By using this data, we are able to apply the same tensile deformations to the digital copies within our simulations. For applying the deformation to the geometry, Dirichlet boundary conditions are applied to the nodes on two opposing surfaces of the mesh. Both surfaces of the model matching the ends of the experimental samples that were fixed by the clamps are exposed to Dirichlet boundary conditions. This means that the deformation of the nodes on these surfaces is given by the tensile deformation in y-direction resulting from the exper-

imental strain value and zero deformation in x and z-direction. The definition of these boundary conditions is the best fit to the experimental setup of the tensile deformation being applied by two clamps. No other macroscopic boundary conditions are applied to the geometries. For the simulation, we consider only the deformation states which are also captured by the DIC, as these are the most interesting results to compare to the experimental data. For the three percent cellulose aerogel, this results in the simulation of 36 deformation states and for the six percent cellulose aerogels, 44 deformation states are considered for simulations.

Regarding the microscopic problem for the FE^2 method, we create the beam frame RVEs from the pore size distributions given in Figure 6.1 and the given porous fractions of the materials. Based on the experimental data, the algorithm for creating the RVEs is applied as described in Section 2.2. As a basis of the creation method for the RVEs of both materials, 500 pore diameters following the respective pore size distributions are given as input to the sphere packing algorithm. We ensure that the RVEs are periodic and in the FE^2 framework the microscopic boundary conditions are applied as described in Section 2.3.1.

The resulting RVE created from the measurements of the three percent cellulose aerogel samples consists of 6,028 beam elements connected at 3,661 nodes and the RVE resulting from the creation method based on the measurements of the six percent cellulose aerogel consists of 5,990 beam elements connected at 3,620 nodes. Both RVEs are presented in Figure 6.6 with the RVE based on the three percent cellulose aerogel on the left and the RVE based on the six percent cellulose aerogel on the right. With the random nature of the RVEs and the similar experimental pore size distributions, it is difficult to recognize significant differences between the configurations of the pores within both RVEs. However, a visual difference in the thickness of the beam elements is observable. The beam elements of the RVE based on the three percent cellulose aerogel appear thinner than the beam elements based on the six percent cellulose aerogel. This impression is also supported by the numbers, as the six percent cellulose aerogel RVE features an average beam radius of 4.33 nm and the three percent cellulose aerogel RVE features an average beam radius of 3.79 nm. The greater radius of the beam elements results from the lower

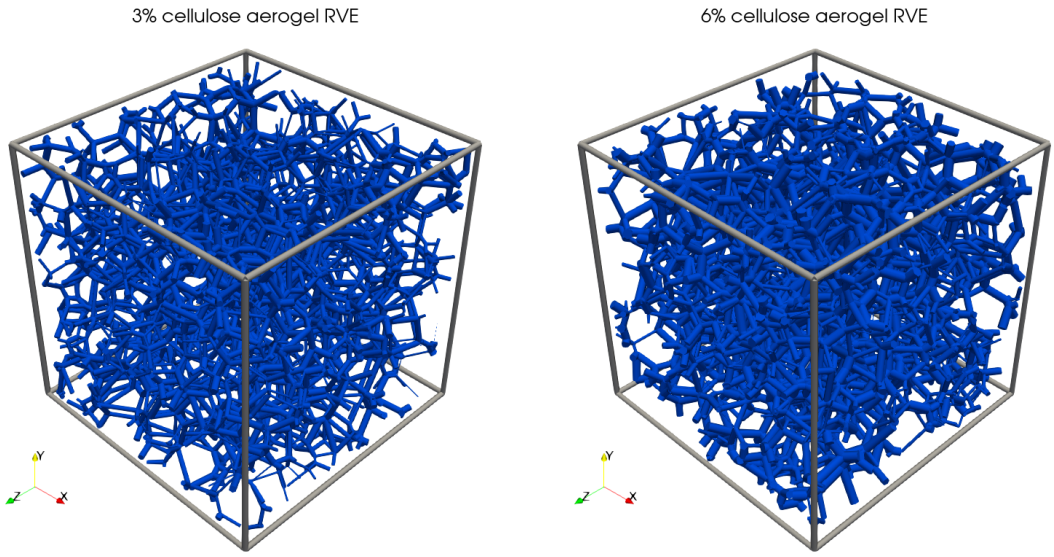


Figure 6.6: Beam frame-structured RVEs modeled to suit the nanostructure of the cellulose aerogel samples. Left image shows the RVE which is created based on the measurements of the three percent cellulose aerogel and right image shows the RVE based on the aerogel sample produced from six percent cellulose.

experimental porous fraction which is considered as input of the algorithm used for creating the RVEs. For the microscopic computations regarding both RVEs, the linear beam frame model is considered as described by Equation (2.9).

The results of the simulation based on the three percent cellulose aerogel are presented in Figure 6.7. The image on the left shows the results of simulation step 34 and the image on the right shows the results of simulation step 36. For the illustration, we considered the same time steps that are also presented for the illustration of the experimental data in Figure 6.4. This allows a better comparison of the results. To be consistent with the experimental results, the coloring of the results in Figure 6.7 refers to the norm of the deformation vector in each respective point. The color pattern of the geometries appears very similar to the coloring shown in the experimental result before failure of the material. We see the largest deformation values close to the ends of the geometry which the Dirichlet boundary conditions are applied to. In addition to the presented deformation values, the FE^2 approach also allows to compute the von Mises stress values within the finite

element mesh. The computation of the stress values is also based on the microscopic model given by the experimental data. An illustration of a number of simulation results for given time steps is presented in the Appendix in Figure A.1. As the coloring of the results refers to the distribution of the von Mises stress, the set of images shows the progression of the stress within the geometry over the timescale of the simulation. With regard to the later time steps 40 and 45, we see larger stress values occurring on the left and right side next to the hole. As these are also the regions in which the experimental samples failed under tension, it can be expected that the stress in the specimens has been the highest in these parts of material.

A video that presents a side by side comparison of the experimental and simulation results for the three percent cellulose aerogel is provided by [101, *holeplate3pct_displacement.mp4*]. The coloring of the geometries refers to the displacement magnitude of the respective evaluation points. In the video, each frame refers to a single deformation step. With a frame rate of one frame per second, each second of the video displays a different deformation state. Additionally, the von Mises stress distribution resulting from the simulation are presented in [101, *holeplate3pct_simulation_vonMises.mp4*]. Here, the coloring of the geometry refers to the von Mises stress measurements.

The images given in Figure 6.8 show the results of the simulation based on the six percent cellulose aerogel sample for two given deformation steps. The same time steps have been considered in the illustration that have also been presented in Figure 6.5. The image on the left shows the deformation step 36 and the image on the right shows the deformation step 38. Like in the results given for the simulation of the three percent cellulose aerogel, the colors in the images refer to the norm of the deformation vector. The colors indicate that the largest deformation occurs close to the edges which the Dirichlet boundary conditions are applied to and along the center y-axis. The pattern of the norm values regarding the deformation appears very similar to the deformation pattern resulting from the experimental data before failure of the material. The distribution of the von Mises stress within the geometry is presented in the Appendix in Figure A.2. As multiple time steps are illustrated in this figure, the images show the progression of the stress values over the timescale

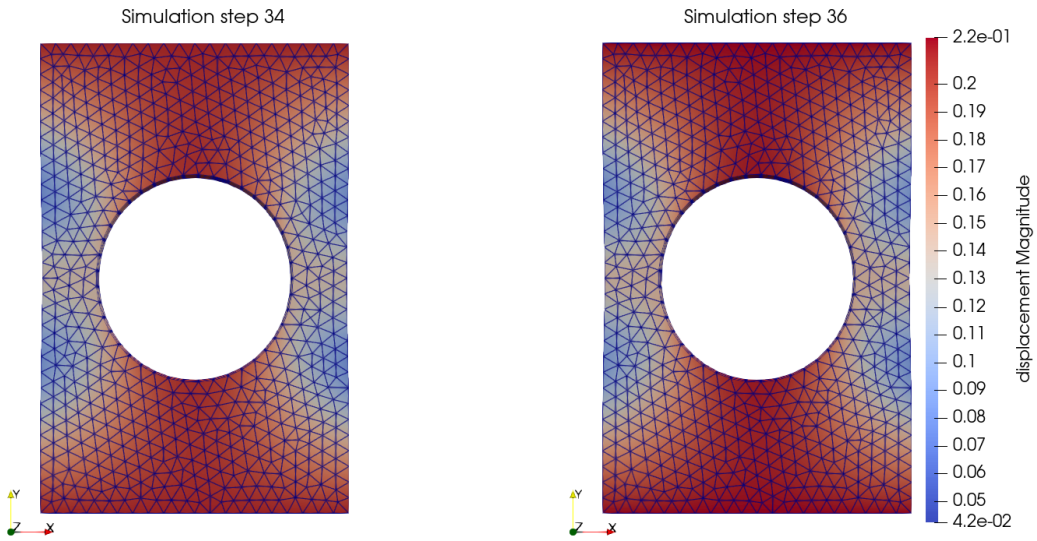


Figure 6.7: Simulation results based on the three percent cellulose aerogel sample. The image on the left shows the simulation result of deformation step 34 and image on the right shows the simulation result of deformation step 36. The colors of the elements represent the norm of the deformation vector in millimeters (mm).

of the simulation. The last images for simulation steps 40 and 45 show that larger stress values occur on the left and right side next to the hole. This behavior aligns with the experimental results, as this is the region where the experimental samples failed. Regarding the scale of the colorbar in Figure A.2, we see significantly larger stress values compared to the scale of the stresses in Figure A.1. This difference between the simulations aligns with the insight of the experimental results presented in Figure 6.3. Here, the cellulose aerogel produced from six percent cellulose in the sol-gel process also shows larger stress values.

Analogous to the three percent cellulose aerogel, a side by side comparison of the experimental and simulation results for the six percent cellulose aerogel is given in the video provided by [101, *holeplate6pct_ displacement.mp4*]. The coloring of the geometries refers to the displacement magnitude and each second of the video presents a different deformation state. The von Mises stress distribution resulting

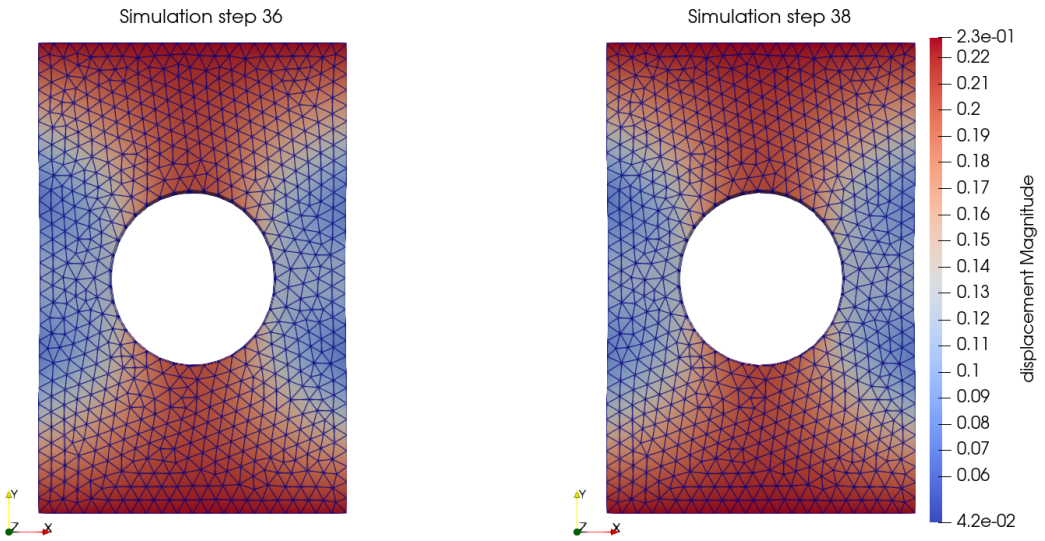


Figure 6.8: Simulation results based on the six percent cellulose aerogel sample. The image on the left shows the simulation result of deformation step 36 and image on the right shows the simulation result of deformation step 38. The colors of the elements represent the norm of the deformation vector in millimeters (mm).

from the simulation of the six percent cellulose aerogel are presented in [101, *hole-plate6pct_simulation_vonMises.mp4*]. The coloring of the geometry refers to the von Mises stress measurements.

6.1.3 Comparison of Experimental and Simulation Data

As the structures of the measurement points, which the experimental data is based on, do not align with the structures of the finite element meshes used for the simulations, a comparison of the nodal deformation values is not directly possible. To allow a comparison of the experimental deformation and the deformation resulting from the simulations, we are able to interpolate the simulation results. Based on the considered Ansatz functions, the solution of the macroscopic boundary value problem can be computed on all points of the macroscopic domain including the measurement points of the DIC software. With this interpolation method, we ob-

tain a set of points and corresponding deformation vectors given with the same structure as presented for the experimental data in Figure 6.4 or Figure 6.5. As both point structures align, it is possible to compute the deviation of the simulation results from the experimental data for each given measurement point. Figure 6.9 shows the deviation with regard to the three percent cellulose aerogel sample for two given time steps. Here, the color of each measurement point refers to the norm of the relative difference between the deformation resulting from the simulation and the deformation given from the experimental data. This means that for a given measurement point, the illustrated color is determined by the value

$$\frac{\|\overline{\boldsymbol{u}}_{sim} - \overline{\boldsymbol{u}}_{exp}\|_2}{\|\overline{\boldsymbol{u}}_{exp}\|_2}$$

with $\overline{\boldsymbol{u}}_{sim}, \overline{\boldsymbol{u}}_{exp} \in \mathbb{R}^3$ being the respective deformation vectors with regard to the simulation or the experimental data. In Figure 6.9, the image on the left shows the relative difference in the deformation state 20 and the image on the right shows the relative difference in deformation state 34 right before the material failed. We see that the distribution of the relative deviation of the simulation results is very similar in both time steps. Especially, the region in the upper left of the sample shows higher relative error values. As shown in Figure 6.4, this part of the aerogel also shows relatively low deformation values which leads to higher relative error values, as we divide by the norm of the experimental deformation. It is notable that especially in the regions of the sample with higher deformation values, the error is relatively small.

The progression of the relative deviation between the experimental and the simulation data is presented in the video provided by [101, *holeplate3pct_rel_diff.mp4*]. The video shows that for the greater part of the tension test, the distribution of the relative deviation is relatively similar.

To allow an appropriate comparison of the experimental stress values presented in Figure 6.3, we compute a similar stress value with regard to the full geometry for each simulation step. To obtain an appropriate stress data for the simulation results, we compute the macroscopic Cauchy stress tensor $\overline{\sigma}$ in each element of

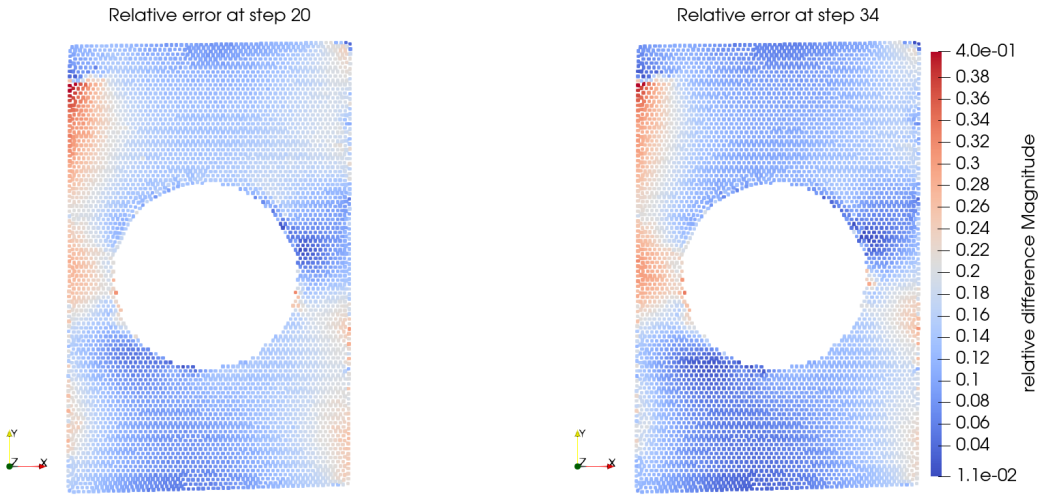


Figure 6.9: Relative difference between the simulation results and the experimental data for the three percent cellulose aerogel sample. Colors of the measurement points represent the norm of the relative difference between the deformation resulting from the simulation and the deformation given by the experimental data.

the macroscopic mesh based on the homogenization of the respective microscopic problem. As a best fit to the experimental stress data, we integrate the main component of the homogenized Cauchy stress tensor referring to the y-direction over the macroscopic geometry

$$\int_{\overline{\mathcal{B}}_0} \overline{\sigma}_{yy} \, \mathrm{d}\overline{V}.$$

As the tension is also applied in y-direction, this is expected to produce the relevant values for the comparison with the experimental data.

The integrated stress values computed from the simulation results are presented together with the experimental stress data in Figure 6.10. For each of the simulation steps of the geometries based on both the three and six percent cellulose aerogels, the stress values are illustrated with dashed lines. Here, the green dashed line refers to the simulation data based on the three percent cellulose aerogel and the blue dashed line refers to the simulation data based on the six percent cellulose aerogel. The progression of the dashed lines indicates a linear stress-strain relationship induced by the beam frame model on the microscopic scale. As we

do not consider any failing of the material for the simulation, the stresses keep increasing with an increase in strain. The stress-strain behavior of both experimental samples is illustrated with continuous lines. We use the same illustration for the experimental data as shown in Figure 6.3. For both experimental curves, we see a slightly non-linear behavior with regard to the stress-strain relationship, as a curvature unequal to zero can be observed visually. Due to the linear behavior of the stress-strain relationship of the microscopic problem, we see a deviation between the curves based on the experimental data and the curves based on the simulation data. However, until the point at which the first crack appears in the material, the deviations between the simulation results and the respective results from the experiments are relatively small. Until that point, the simulation results can provide appropriate approximations of the stress-strain behavior of the cellulose aerogel samples.

For the simulations regarding both materials, we considered the same Young's modulus for each beam element on the microscopic scale. As Young's modulus and the composition of the beam elements within the RVE drive the scaling of the resulting stresses, the only property that differs regarding the stress computation on the microscopic scale is the structure of the RVE which is based on the experimental data. The results in Figure 6.10 show that the difference in the stress scales between the three percent cellulose aerogel and the six percent cellulose aerogel is slightly underestimated by the simulation results, as the stress-strain curve of the simulation based on the three percent cellulose aerogel is slightly higher than the experimental data and the stress-strain curve of the simulation based on the three percent cellulose aerogel is slightly lower than the respective experimental data. However, as this difference is only driven by the difference in the pore size distributions and porous fractions given from the experimental analyses, we can conclude that the modeling of the nanostructure as presented in Section 2.2 is quite suiting for cellulose aerogels and can largely be used for describing the materials mechanical behavior.

Based on the distribution of the von Mises stress of the simulation results presented in Figure A.1 and Figure A.2, we see that the highest stress values occur in the regions of the geometries in which the respective experimental aerogel samples

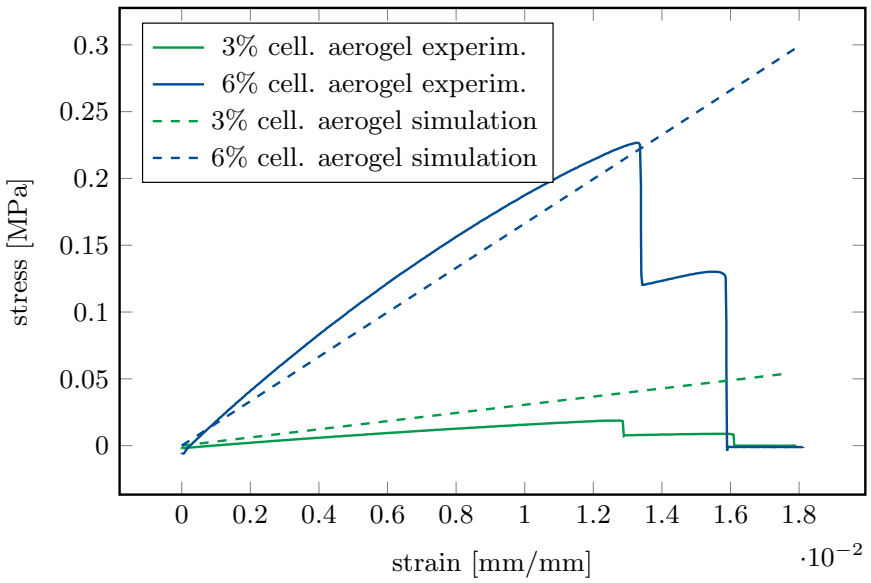


Figure 6.10: Stress-strain relationships resulting from the experimental data and the simulations. Green lines refer to the aerogel material produced from three percent cellulose and blue lines refer to the aerogel produced from six percent cellulose. Continuous lines show the experimental results and the dashed lines show the data resulting from the simulations. The stress data is given in megapascal (MPa).

failed. As we see a strong correlation between the stresses in the simulation and the real-world failure of the specimen, it would be possible to derive a method for predicting the failing of different cellulose aerogel samples. The findings of this comparison with regard to the tension experiments indicate that the relationship between simulation and mechanical behavior of the experimental samples could be used to develop a method of this kind. However, to achieve this, much more research with different experimental setups would be needed with regard to this relationship. This could be an interesting topic for some future work.

6.2 Torsion Experiment for Cellulose Aerogel

In the tension experiments for the cellulose aerogel samples which are presented in Section 6.1.1, the deformation is mainly applied to the material in one direction. As the DIC software only tracks the deformation on one surface of each specimen, the resulting deformation data is two-dimensional. In addition to the comparison for the tension experiments, we also consider a comparison of simulation and experimental results for a truly three-dimensional problem.

The torsion of a solid causes deformations orthogonal to the axis of rotation and applies complex shear forces to the material. Therefore, this is an interesting test case with regard to experimental analysis and the simulation. To create a truly three-dimensional deformation problem, we consider the torsion of a cellulose aerogel sample and run a simulation based on the given experiment. The comparison between the experimental data and the data resulting from the simulation is used to evaluate the performance of the multiscale simulation method.

The progression of the experimental and simulation results of the torsion test are presented in videos that are provided via the Zenodo database [102]. The entry refers to a Zenodo page that provides access to multiple videos presenting side-by-side comparisons of experimental and simulation results for the torsion test. The videos cover different viewing angles for the three-dimensional results. Additionally, two video representations of the progression of the differences between the experimental and simulation results are included in the database.

6.2.1 Experimental Setup

The torsion experiment was performed on a plate that has been synthesized from cellulose aerogel. As described with regard to the tension experiment presented in Section 6.1, cellulose aerogels are well suited for comparing experimental and simulation results because they feature a fibrillar nanostructure that fits the modeling with beam elements. The specimen is given as a long and thin plate with the dimensions approximately given by a length of 200 mm, width of 20 mm, and

thickness of 2 mm.

To apply the deformation to the sample, the specimen is fixed with two clamps placed at both ends of the plate. One of the clamps is connected to a motor that is able to rotate the clamp and apply the torsional deformation to the sample. The full setup is presented in Figure 6.11. The photograph shows a snapshot taken during the experiment. In the center of the image, the white cellulose aerogel sample is shown with a strong torsional deformation applied to the specimen. The two metal structures on the top and on the bottom of the image involve the clamping mechanisms used to fix the sample. The motor that drives the rotation can be seen under the lower bracket. The motor rotates the entire lower clamping mechanism, which transfers the deformation to the aerogel plate.

Similar to the experimental results presented in Section 6.1.1, the digital deformation data regarding the torsion of the aerogel plate is obtained from DIC. Two cameras are used which take multiple photographs of the deformation state of the specimen. With the use of DIC software, the captured images can be processed to compute three-dimensional deformation data for a number of measurement points densely distributed over the surface of the sample. Note that, similar to the case of the hole plate geometries described in Section 6.1.1, the DIC software only produces deformation data for the measurement points on one surface of the specimen. This means that we do not know the deformation of the full sample. However, as the sample is relatively thin in comparison to its length, it is expected that the deformation behavior is similar over the thickness of the material. As parts of the specimen are rotated over the timescale of the experiment and the position of the cameras is fixed, it happens that the DIC software loses track of some measurement points if the tracked surface is rotated so far that it gets hidden from the cameras' perspective. In this case, no deformation is tracked by the DIC software for the respective measurement points. This behavior is also shown in Figure 6.11, as only the thin surface on the side is exposed to the view of the cameras for some parts of the sample. With the number of available measurement points decreasing for greater torsion of the aerogel sample, this can worsen the quality of our deformation data and become a problem for the analysis. The torsion experiment has been

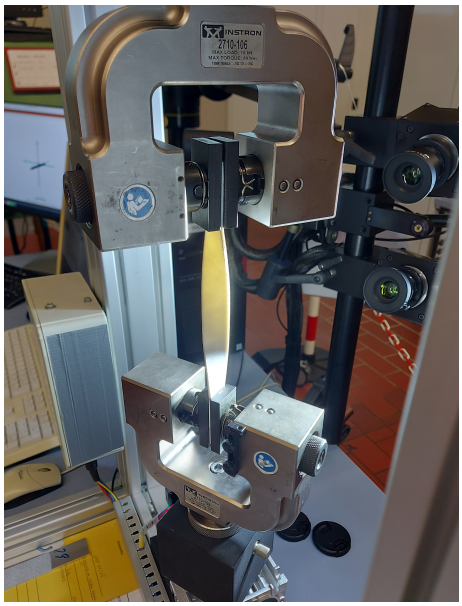


Figure 6.11: Setup of the torsion experiment. The photograph is provided courtesy of the Institute for Frontier Materials on Earth and in Space at German Aerospace Center (DLR). The photograph has been taken by Max Zinke.

executed seven times with different cellulose aerogel samples used in each run. We focus only on the sixth execution for both the analysis of the experimental data and the comparison with simulation results. Regarding this single experiment, the quality of the experimental data is considered the best in comparison to the other runs, as only a relatively small ratio of deformation data is missing.

For the experiment, the cellulose aerogel sample has been twisted up to an angle of about 90° and subsequently twisted back again to its initial state. The angle of the torsion is not exactly given, as the motor which transferred the deformation is operated manually. The DIC software has been used to track the experiment including the torsion up to 90° and the subsequent relaxation of the material. A total of 7,350 measurement points were considered on the surface of the material. Over the timescale of the experiment, 630 snapshots of the aerogel's deformation state have been taken in parallel by both cameras. Each of the images has been processed by the DIC software and has produced a deformation state for each measurement point, if the point had been visible to the cameras in the respective time step.

Two videos illustrate the deformation states resulting from the DIC. For the videos [102, *torsion_displacement_iso.mp4* and *torsion_displacement_xy.mp4*], access is provided via the Zenodo data base. In the video data, the experimental results are presented on the left. Each frame that is displayed refers to a single deformation step for the torsion experiment that has been captured with the DIC software. As the videos feature a frame rate of ten frames per second, ten deformation steps are displayed in each second of the videos. It is apparent that some parts of the experimental results disappear over the timescale of the experiment. This behavior indicates the missing deformation data of measurement points that could not be tracked by the cameras. In the resulting video, the coloring of the sample illustrates the magnitude of the deformation. This value is computed as the norm over the deformation vector given at a respective point. The results show that the deformation is largest in the regions close to the upper and lower boundaries where the sample is fixed by the clamps.

6.2.2 Simulation Setup

To replicate the experimental setup with a simulation, the microscopic and macroscopic boundary conditions need to be defined for the multiscale framework. On the macroscopic scale, the simulation requires a geometry which suits the real-world dimensions of the aerogel plate. Additionally, for controlling the torsional deformation of the geometry, proper boundary conditions need to be defined that resemble the conditions during the experiment. As the measurement points of the DIC are densely given on the surface of the aerogel sample, we can use these coordinates as nodes of the finite element mesh. For this definition of the macroscopic grid, we use the first state of the sample given by the DIC software. This state acts as the reference configuration in which the deformation of all measurement points is zero. The deformation vectors of the subsequent states are each based on this reference configuration. Additionally, this means that all nodes considered in any given deformation state of the experiment are included in the undeformed first state. Using this state as reference for creating the domain for the simulation ensures that all

relevant measurement points are included. The analysis of the DIC measurement points shows that the image processing causes the appearance of some false nodes as duplicates or nodes that are not deformed correctly. It is crucial to identify all of these nodes, as they can produce hanging nodes or other defects within the macroscopic finite element mesh. Already minor defects in the mesh can worsen the convergence behavior of the macroscopic solver. With a consistent set of measurement points, we use the coordinates as nodes of a finite element surface mesh. These nodes create the basis for one surface of our macroscopic geometry. As we consider a three-dimensional simulation setup, we duplicate the nodes and transfer them according to the thickness of the aerogel sample. Based on the assumption that the aerogel sample is symmetric, this method produces a front and back surface for the three-dimensional geometry. Connecting the boundary nodes of both surfaces yields an enclosed volume that acts as domain for the simulation. We use the meshing software Gmsh [55] for creating the finite element discretization of the volume. Tetrahedral P1 elements are used as basis for the discretization. The meshing tool uses the measurement points provided by the DIC data as nodes for the mesh and includes additional points to ensure an even distribution of the length scales of the tetrahedral elements. The full finite element mesh contains 17,982 nodes and 70,963 tetrahedral elements. This results in 53,967 degrees of freedom (DOF) for the macroscopic problem. As described in Section 2.3, the macroscopic problem is based on the momentum balance equation. For the simulation regarding the torsion test, we use the weak formulation that is based on the Cauchy stress tensor.

Using the DIC measurement points as basis for creating the finite element mesh yields two additional advantages. On the one hand, it simplifies the comparison between simulation and experimental results, as the differences between the deformation states can be directly compared without the need of additional interpolation. On the other hand, the method allows a definition of the boundary conditions based on the given experimental deformation states. To align the deformation conditions of the simulation and the experiment, we apply Dirichlet boundary conditions to the finite element nodes located at the opposing upper and lower edges of the geometry. These are also the parts of the aerogel sample that are fixed with the clamps.

The deformation at these Dirichlet nodes is defined to align with the deformation states of the experiment. For a given time step, this means that we define the deformation of each Dirichlet node as being equal to the deformation vector computed by the DIC software for the corresponding measurement point. In Section 6.2.1, we described the issue of missing data within the DIC results. Since data is most frequently missing from the nodes at the edge of the aerogel sample, this behavior can also cause problems with regard to the definition of the Dirichlet boundary conditions. In order to deal with the missing deformation data, we apply Dirichlet boundary conditions only to nodes for which the deformation data is given by the DIC results. If the data for one node at the edge of the geometry is missing in a considered time step, the deformation of the respective node is not specified and it is not affected by boundary conditions. Especially, for the time steps with large deformation, this can lead to a significantly reduced number of Dirichlet nodes. To reduce the impact of the missing data on the simulation, we neglect a few nodes at the very bottom of the sample plate for the simulation and the comparison with experimental results. This approach yields a slightly reduced domain for the simulation but significantly improves the consistent definition of boundary conditions. With regard to the comparison of the simulation and experimental results, we discuss the impact of the reduced quality of the deformation data on the accuracy of the simulation.

On the microscopic scale, we consider a beam frame model with the RVE created from the method described in Section 2.2. As the specimen used for the torsion experiment is made from cellulose aerogel, we consider the experimental pore size distribution and porous fraction of cellulose aerogel as inputs of the creation method. Note that the material that was used for the analyses of the nanostructure is obtained from the same synthesis batch as the aerogel specimen utilized for the torsion test. The materials are synthesized with seven percent cellulose used in the sol-gel process. As the same conditions are applied, it is expected that the materials from the same batch share very similar properties on the nanoscale. The measured porous fraction equals 0.882 for the cellulose aerogel. For creating the RVE, we use 500 pore diameters as basis for the sphere packing algorithm. The diameters

are randomly generated based on the distribution of the pore sizes provided by the experimental analyses. The beam frame structure resulting from the Voronoi tessellation contains 6,064 beam elements connected at 3,711 nodes. For the microscopic computations in the FE^2 method, we compute the homogenized Cauchy stress tensor from a small matrix-vector product as described in Section 3.4.

For the simulation, we consider the same 630 deformation steps that have also been covered by the experimental data. In each time step, the deformation of the geometry is applied by the definition of the Dirichlet boundary conditions. We use the parallel Newton-Krylov method as described in Chapter 5 for computing the solution in each time step. Here, the computed solution of the previous step is used as the initial guess for the next deformation. For the non-linear solver, a tolerance of $1e-6$ is considered. As the differences between subsequent deformations are relatively small, the macroscopic solver converges within a few iterations for each time step. For reducing the computation time of the simulation, the microscopic and macroscopic computations are executed in parallel with a single compute node on *Fritz* HPC cluster [49] with 72 cores. The setup of this cluster is described in Chapter 5. The parallelization of the multiscale method reduces the runtime for all time steps to about 14 minutes.

The deformation states of simulation and experimental results are presented with a side-by-side comparison in two videos that can be accessed over the Zenodo repository [102, *torsion_displacement_iso.mp4* and *torsion_displacement_xy.mp4*]. The coloring of the macroscopic geometries illustrates the distribution of the displacement magnitude. The video *torsion_displacement_iso.mp4* displays both results with an isometric view and uses a meshing of the experimental DIC measurement points to achieve a three-dimensional appearance of the experimental sample. In *torsion_displacement_xy.mp4*, the viewing angle is set orthogonal to the xy-plane. As the surface of the experimental sample is displayed, the DIC measurement points are used for the illustration of the experimental results. Both videos show that for large parts of the torsion test, the simulation yields very similar deformation behavior as the experimental sample. For large torsional deformations of the specimen, we see volatile behavior of the simulation with more apparent differences between

subsequent deformation states. This unnatural behavior is induced by missing data close to the edges of the aerogel sample. As no deformation data is given for these nodes, the Dirichlet boundary conditions are not correctly defined. This leads to inaccurate deformation behavior of the simulation. Note that the issue with missing data is only noticeable for a very short time frame of the experiment. For large parts of the torsion test, the Dirichlet boundary conditions are correctly applied and the simulation and experimental results behave similar.

6.2.3 Comparison of Experimental and Simulation Data

As explained in Section 6.2.2, the creation of the macroscopic geometry for the simulation setup is based on the coordinates of the measurement points that are tracked by the DIC software. This means that large parts of the coordinates of the nodes within the experimental and simulation geometries are identical. Therefore, the difference in deformation between both results can be directly computed. No additional interpolation of the solutions is required for the comparison between experimental and simulation data.

To evaluate the performance of the multiscale approach for the torsion of the cellulose aerogel plate, we compute the difference between the displacement vectors resulting from the simulation and the experimental data. This means that for each position vector that is identically given in the set of DIC measurement points and in the finite element mesh for the simulation, the deviation between both results is computed as the norm over the differences

$$\|\overline{\boldsymbol{u}}_{sim} - \overline{\boldsymbol{u}}_{exp}\|_2$$

with $\overline{\boldsymbol{u}}_{sim}, \overline{\boldsymbol{u}}_{exp} \in \mathbb{R}^3$ being the macroscopic displacement vectors given by the simulation and experimental results. The distribution of this deviation measurement and its progression over the scope of the experiment is presented in a video that can be accessed via [102, *torsion_ difference_ xy.mp4*]. The coloring of the results presented in the video refers to the norm of the deviation that is evaluated at each

measurement point that is included in both node sets for the simulation and experimental results. The video shows that with an increase of the torsion that is applied to the aerogel plate, the error values induced by the simulation increase. The largest deviations occur during the maximum torsion of the sample. Generally the error values close to the edges of the specimen are higher than in the vertical center. Especially, high values for the deviation measurements are apparent in the upper left and lower right sections of the geometry. However, the nodes within these sections of the material are also exposed to large deformations.

For further analyses of the deviation between the simulation and experimental results, we compute the relative deviation between the displacement vectors given by

$$\frac{\|\overline{\boldsymbol{u}}_{sim} - \overline{\boldsymbol{u}}_{exp}\|_2}{\|\overline{\boldsymbol{u}}_{exp}\|_2}.$$

In the analysis of the error computed as the norm of the difference between both displacement vectors, we observed a significant correlation between the deviation magnitude and the resulting error. The relative error helps to filter out this effect. The video provided by [102, *torsion_ rel_ difference_ xy.mp4*] presents the distribution of this relative error over the timescale of the torsion experiment. Here, the coloring of the results refers to the measurement of the relative deviation computed in each measurement point. The results show that for the largest part of the experiment, the distribution of the relative error over the measurement points remains similar. The largest relative error values are observed close to the center line that is vertically aligned with the x-axis. Based on the results of the error measurements shown in [102, *torsion_ difference_ xy.mp4*], the absolute deviation between experimental and simulation results is small in this region. This means that the large relative error is mainly driven by the small deformation of the experimental sample in this section of the aerogel plate. Additionally, the video shows an increase in the relative error toward the end of the experiment. For these time steps, the torsion of the sample is reversed and the aerogel plate is returning to its initial state. Therefore, in this stage of the experiment, the experimental deformation magnitude approaches zero and this increases the relative error. However, with regard to the

absolute deviation, no significant increase in the error induced by the simulation can be observed toward the end of the experiment.

A global evaluation of the progression of the displacement regarding the simulation and experimental data is presented in Figure 6.12. For each time step, the values presented in the graph are computed as the average over all displacement magnitudes of the nodes within a macroscopic geometry. This average displacement measurement is computed for experimental and simulation data. It allows a global quantification of the displacement of the macroscopic geometry for each time step. In Figure 6.12, the blue curve refers to the progression of the experimental average displacement and the red curve refers to the measurement that is based on the simulation results. The general behavior of both curves is very similar. The increase in average displacement appears linear until the peak that is approximately at deformation step number 315. In the experiment, this is the time step in which the direction of torsion is reversed and the aerogel plate is twisted back to its initial state. Therefore, after the peak, we see an approximately linear decrease in the average displacement measurement for the experimental and the simulation results. For the reverse torsion of the cellulose aerogel plate, a slight difference between the experimental and the simulation curves is apparent. For the first half of the experiment, both curves are almost aligned. This deviation in average displacement could indicate some plastic behavior of the aerogel material that is not considered in the multiscale simulation. However, the major material behavior of the aerogel in this experiment appears to be elastic, since the curve approaches zero in the last time steps.

In addition to the local evaluations of the deviation between the simulation and the experimental results, we compute a global deviation of the solutions. This means that the norm over the difference between the global deformation vectors is evaluated. Let $\overline{\boldsymbol{U}}_{sim}$ and $\overline{\boldsymbol{U}}_{exp}$ be the vectors that contain the displacement vectors for all macroscopic nodes that are included in the DIC measurement points and in the finite element mesh used for the simulation. The absolute deviation between

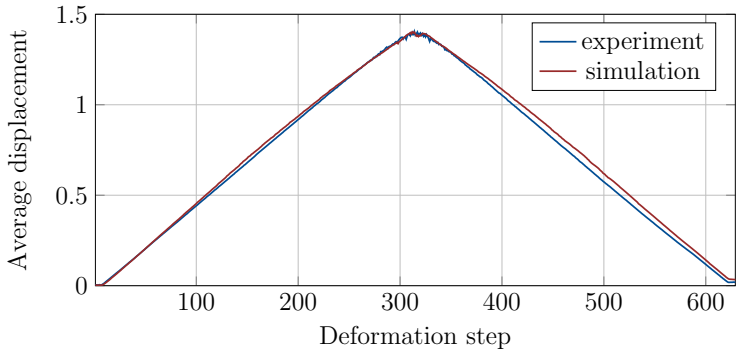


Figure 6.12: Progression of the average displacement regarding the torsion of a cellulose aerogel plate. The blue curve presents the average displacement resulting from the experimental data and the red curve is based on the simulation data.

both global vectors is defined by

$$\|\overline{\boldsymbol{U}}_{sim} - \overline{\boldsymbol{U}}_{exp}\|_2.$$

This deviation value is computed for each deformation state of the torsion test and the progression of the error is presented in the top image of Figure 6.13. The shape of the error curve is almost identical to the curves presented in Figure 6.12. This relationship illustrates the correlation between the magnitude of the displacement and the absolute deviation between the solutions. For the deformation steps that correspond to the states with the strongest torsional deformation, the error curve shows sharp upward spikes. This behavior of the error curve is related to the volatile behavior of the simulation that is also observed in the side-by-side comparison presented in [102, *torsion_displacement_iso.mp4*]. As explained above, this behavior is caused by the missing experimental data in these time steps and the inaccurate definition of Dirichlet boundary conditions.

The graph in the bottom image of Figure 6.13 presents the global measurement of the relative deviation between the experimental and simulation results. The

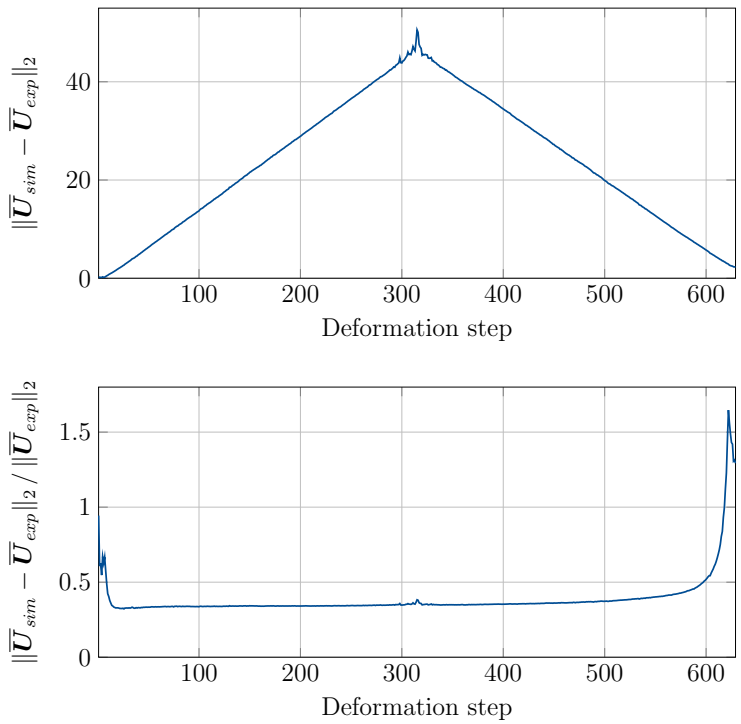


Figure 6.13: Progression of the deviation between the experimental and simulation results with regard to displacement. The top image shows the absolute deviation and the image on the bottom presents the relative deviation values.

relative error value is defined by the expression

$$\frac{\|\overline{\boldsymbol{U}}_{sim} - \overline{\boldsymbol{U}}_{exp}\|_2}{\|\overline{\boldsymbol{U}}_{exp}\|_2}.$$

The graph in Figure 6.13 shows that the relative error measurement remains relatively constant for the majority of the torsion experiment. The relative error curve shows slightly higher values for the first deformation states. However, the error decreases rapidly within a few time steps. Additionally, we see a significant increase of the error value in the last deformation steps. The high relative deviation at the beginning and end of the time frame correspond to the states with the least global deformation. As the norm of the experimental deformation vector $\|\overline{\boldsymbol{U}}_{exp}\|_2$ is close to zero, this leads to high values for the relative error measurement.

To conclude the results regarding the torsion test for a cellulose aerogel plate, the multiscale simulation provides qualitatively very similar deformation behavior compared to the experimental DIC measurements. The relative deviation between simulation and experimental data remains small for the majority of the experiment. Missing data in the states of the experiment with the largest deformations results in inaccurate definition of the Dirichlet boundary conditions. For a more in-depth analysis of the performance of the multiscale approach for the three-dimensional deformation of cellulose aerogels, additional stress data would be required. For a torsion test, the qualitative deformation behavior can be similar for different materials. However, the difference in material behavior is often highlighted by the stress responses of the specimens. With regard to complex three-dimensional deformation test cases, a comparison between the real-world stress behavior of open-porous materials and the stress response computed from the multiscale simulation would be a relevant topic for future research.

Chapter 7

Application-Driven Numerical Results

As many applications benefit from the use of aerogels as insulation materials, it is crucial to investigate the impact of aerogels on the mechanical properties of the components. In this chapter, we investigate the application of aerogel composites as a building material with regard to the mechanical properties. For a mock-up airplane cabin, we simulate the compression of the geometry and evaluate the results based on the deformation and stress data.

The chapter is structured as follows. Section 7.1 describes the general test case regarding the compression of a mock-up airplane cabin geometry. We describe the relevance of aerogel composites for real-world applications and point out their benefits for the mechanical properties of the material. Section 7.2 presents the simulation setup for computing the mechanical response of the cabin geometry. Here, we explain the incorporation of different material models in the FE^2 method and describe the stress-strain relationship that is induced by linear elasticity. The results of the multiscale simulations are presented in Section 7.3. We analyze the deformation and stress results for this specific tension test.

7.1 Aerogel Composite for Mock-Up Airplane Cabin

For aerospace applications, the use of high-performance materials is often of great importance. Besides the need for high-performance insulation, for many of these projects, it is a crucial requirement to build components as light as possible. In these cases, the use of aerogels as insulation material suggests itself and has also proven beneficial for various projects in the past [44]. As presented in Chapter 6, the limit of aerogel components is often given by their mechanical strength. Especially, for the tension test regarding cellulose aerogels, the experimental results in Figures 6.4 and 6.5 show failure of the material even for small tensile deformation. The failure of the material under small deformations can lead to crucial problems, as the structural stability of the components is affected and the insulation properties of the aerogel can deteriorate.

To improve the mechanical stability of aerogel components, the use of composite materials can be considered. This involves the combination of aerogel parts with components made from materials that provide greater mechanical stability. Combining both materials can help to reduce the deformation of the component, when exposed to significant forces. While improving the mechanical properties of the material, the strong insulation properties of the aerogel are retained. With smaller deformations, the failure of the material and the associated deterioration of the insulation properties is reduced. Different approaches can be considered to create an aerogel composite [23]. In addition to conventional methods to combine the materials such as gluing and mechanical fixation, it is possible to use in-situ approaches in which both materials are combined within the synthesis of the aerogel [151].

7.2 Simulation Setup

Given the FE^2 method as described in Chapter 2, the application of the multiscale approach for the simulation of aerogel composites requires only small adjustments to the algorithm. As the method considers individual microscopic problems for each evaluation point within the macroscopic finite element mesh, the FE^2 ap-

proach allows to differentiate between macroscopic elements that are modeled as aerogel and elements modeled as a different material. To allow the simulation of composite materials, we adjust the multiscale method so that it can use different microscopic models for the computation of the macroscopic stress tensor in each Gaussian quadrature point. As the FE^2 method can handle various stress-strain relationships resulting from the microscopic model, we allow to use different material models to define the mechanical behavior of the non-aerogel material. Therefore, for each element of the macroscopic mesh, the model for computing the macroscopic stress from a given deformation gradient can be defined individually.

Based on the idea of using aerogel materials in aerospace applications, we consider a mock-up airplane cabin geometry for the simulation of an aerogel composite. The fuselage of an aircraft imposes special requirements on the materials used, as it must be lightweight in order to reduce fuel consumption, but must withstand the application of significant forces without failure. Therefore, the application of aerogel composites is well suited for these components. We consider a cutout section of the cabin wall of the airplane featuring three windows. The geometry is meshed with P1 finite elements using Gmsh [55]. The macroscopic finite element mesh contains 38,864 elements with 8,503 nodes.

The geometry of the mock-up airplane cabin is separated into two distinct layers. The layout of the cabin wall and its dimensions are presented in Figure 7.1. The holes in the geometry that resemble the windows of the airplane feature a width of 30 cm and a height of 60 cm. Additionally, Figure 7.1 shows discretization of the macroscopic geometry based on tetrahedral finite elements.

For the inner layer of the cabin wall, an open-porous material is considered. In a real-world application this part of the cabin wall would be used for insulation. Thus, the macroscopic problem definition resembles the insulation of an airplane cabin with an open-porous material like aerogel. For the mechanical modeling and the stress-strain relationship, we consider the homogenization approach with the beam frame microstructure as described in Chapter 2 for the finite elements given in this section.

We assume that the outer layer of the cabin wall is made from aluminum. With

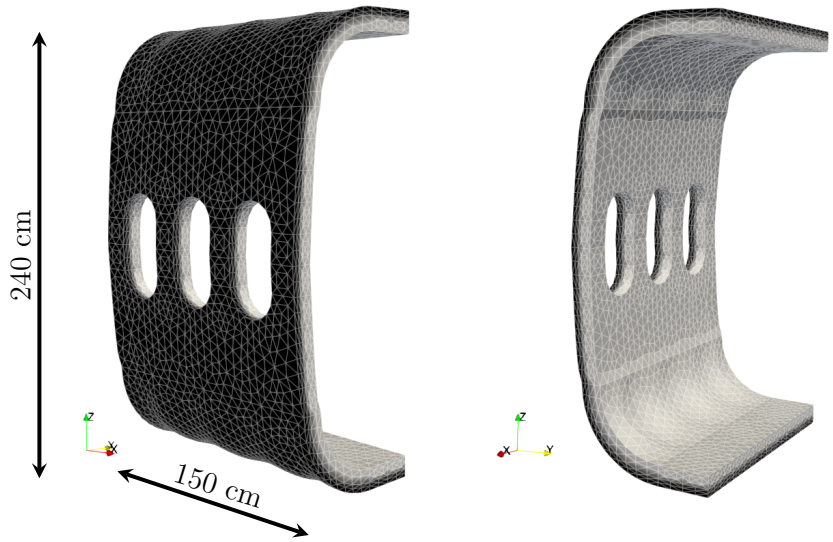


Figure 7.1: Layout and dimensions of the mock-up airplane cabin. The dark colored section refers to the outer layer and the light colored section refers to the inner layer of the cabin wall.

regard to real-world applications, this modeling resembles an aluminum outer shell of the airplane. For the definition of the stress-strain relationship for the outer layer of the cabin wall, we consider the linear elasticity model, which can be used for small deformations [105, chapter 6]. The assumption of linear elasticity for the deformation of solids follows the basic principle of Hooke's law

$$\overline{\sigma} = \frac{\overline{E}}{1+\overline{\nu}} \left(\overline{\varepsilon} + \frac{\overline{\nu}}{1-2\overline{\nu}} \cdot \operatorname{trace}(\overline{\varepsilon}) \cdot I \right), \tag{7.1}$$

which states a linear relation between the Cauchy stress tensor $\overline{\sigma}$ and the strain tensor $\overline{\varepsilon}$ [13, chapter IV]. Here, the material's behavior is defined by Young's modulus $\overline{E} \in \mathbb{R}$ and Poisson's ratio $\overline{\nu} \in \mathbb{R}$. We use the bar notation for each of these entities to indicate that the given relations describe the materials behavior on the macroscopic scale. For an evaluation point $\overline{X} \in \overline{\mathcal{B}}_0$ and a corresponding macroscopic deformation $\overline{\boldsymbol{u}} \in \mathbb{R}^3$, the symmetric strain tensor $\overline{\varepsilon} \in \mathbb{R}^{3\times 3}$ is defined

as

$$\overline{\varepsilon} = \frac{1}{2} \cdot \left(\nabla \overline{\boldsymbol{u}}(\overline{X}) + \nabla \overline{\boldsymbol{u}}(\overline{X})^{\mathsf{T}} \right)$$

with $\nabla \overline{\boldsymbol{u}}(\overline{X})$ being the Jacobian matrix of the deformation vector with respect to the local coordinate [13, chapter IV, §1].

For the three-dimensional case that we consider, it is possible to rewrite Equation (7.1) as a simple matrix multiplication

$$\begin{pmatrix} \overline{\sigma}_{xx} \\ \overline{\sigma}_{yy} \\ \overline{\sigma}_{zz} \\ \overline{\sigma}_{xy} \\ \overline{\sigma}_{xz} \\ \overline{\sigma}_{yz} \end{pmatrix} = \boldsymbol{D} \cdot \begin{pmatrix} \overline{\varepsilon}_{xx} \\ \overline{\varepsilon}_{yy} \\ \overline{\varepsilon}_{zz} \\ \overline{\varepsilon}_{xy} \\ \overline{\varepsilon}_{xz} \\ \overline{\varepsilon}_{yz} \end{pmatrix} \tag{7.2}$$

with the matrix $\boldsymbol{D} \in \mathbb{R}^{6\times 6}$ depending on the material parameters $\overline{E}, \overline{\nu} \in \mathbb{R}$. In the literature, the relation presented in Equation (7.2) is also referred to as Voigt notation [13, chapter IV, §3]. Based on Equation (7.1), the matrix $\boldsymbol{D}$ is defined as

$$\boldsymbol{D} = \frac{\overline{E}}{(1+\overline{\nu}) \cdot (1-2\overline{\nu})} \begin{pmatrix} 1-\overline{\nu} & \overline{\nu} & \overline{\nu} & & & \\ \overline{\nu} & 1-\overline{\nu} & \overline{\nu} & & \mathbf{0}_{3\times 3} & \\ \overline{\nu} & \overline{\nu} & 1-\overline{\nu} & & & \\ & & & 1-2\overline{\nu} & 0 & 0 \\ & \mathbf{0}_{3\times 3} & & 0 & 1-2\overline{\nu} & 0 \\ & & & 0 & 0 & 1-2\overline{\nu} \end{pmatrix}.$$

For a given strain tensor $\overline{\varepsilon}$, Equation (7.2) reduces the computation of the components of the Cauchy stress tensor to a simple matrix-vector multiplication in the case of linear elasticity.

As mentioned above, the multiscale approach for open-porous materials is considered for describing the mechanical properties of the elements located within the inner layer of the airplane cabin. This means that within the affected elements,

the stress-strain relationship is computed based on a given microscopic beam frame model. The RVE used on the microscopic scale is created based on the sphere packing algorithm and Voronoi tessellation as described in Section 2.2. As input for the creation method, we use experimental data that is obtained from the analysis of the nanostructure of cellulose aerogels. We apply the creation method based on the porous fraction and the pore size distribution obtained from gas adsorption methods that are used in combination with the Barrett–Joyner–Halenda (BJH) model [7]. As input of the sphere packing algorithm, we randomly generate 500 pore diameters according to the pore size distribution. The resulting beam frame model consists of 6,064 beam elements connected at 3,711 nodes.

The macroscopic boundary value problem considered for the airplane cabin is based on the momentum balance equation described in terms of the Cauchy stress tensor. Details regarding this formulation are given in Section 2.3. This means that in each Gaussian quadrature point, the homogenized Cauchy stress tensor is computed based on the microscopic model. For the elements within the outer layer of the cabin, the relationship between the macroscopic Cauchy stress tensor and the strain is given by Equation (7.2). For these elements, the components of the Cauchy stress tensor can be directly computed as the matrix-vector product. Regarding the evaluation points of the elements that are located in the inner layer of the cabin, the Cauchy stress tensor is computed based on the RVE on the microscopic scale. As described in Section 3.4, for the linear beam frame problem, the homogenized Cauchy stress tensor can be computed from the single matrix multiplication that is independent from the size of the RVE. The respective matrix, which is derived from the beam frame structure, is multiplied with a vectorization of the macroscopic deformation gradient to obtain the respective homogenized Cauchy stress tensor. This allows a fast computation of the stress components within the FE^2 method and eliminates the need for a machine learning-based surrogate model.

Note that we describe the composition of the macroscopic stress-strain relationship from the linear elasticity model and the linear computation of the Cauchy stress tensor resulting from the beam frame model. Since both of these models provide a linear relationship between the local displacement and the Cauchy stress

tensor, this formulation yields a linear global relationship for the macroscopic finite element problem. Despite this linear relationship, we use Newton's method to solve the macroscopic problem. In general, the application of a non-linear solver is not necessary because a solver for systems of linear equations would also optimize the macroscopic residual. When Newton's method is applied to the linear problem and the linear solver for the update step converges, we expect a significant reduction in the residual within the first iteration.

7.3 Simulation Results

Regarding the testing of the mock-up airplane cabin, we consider a compression of the geometry in z-direction. To apply the deformation, Dirichlet boundary conditions are applied to the nodes within the top and bottom surfaces of the macroscopic mesh. The respective nodes are shifted in z-direction towards the center of the geometry and the deformation in x and y-direction are set to zero. The total deformation is defined to enforce a compression of 10% relative to the height of the cabin. Newton's method is applied as non-linear solver on the macroscopic scale. The simulation is computed with the parallel implementation based on the PETSc framework, which is described in Chapter 5. This means that the microscopic computations for each macroscopic quadrature point are conducted in parallel. The results of the simulation are presented in Figure 7.2. The image on the left shows the front of the geometry with the surface of the outer layer exposed. The image on the right shows the inner layer of the airplane cabin that is modeled based on cellulose aerogel. The coloring of the geometries refer to the norm of the deformation vector. The figures show which parts of the geometry exhibit the highest deformation in the simulation result. Although the geometry of the mock-up airplane cabin is symmetric, the distribution of the deformation magnitudes appears non-symmetric. However, this behavior does not appear to be caused by the solver not converging. The macroscopic Newton solver terminates with a relative residual of $2.18e-8$ after a single iteration.

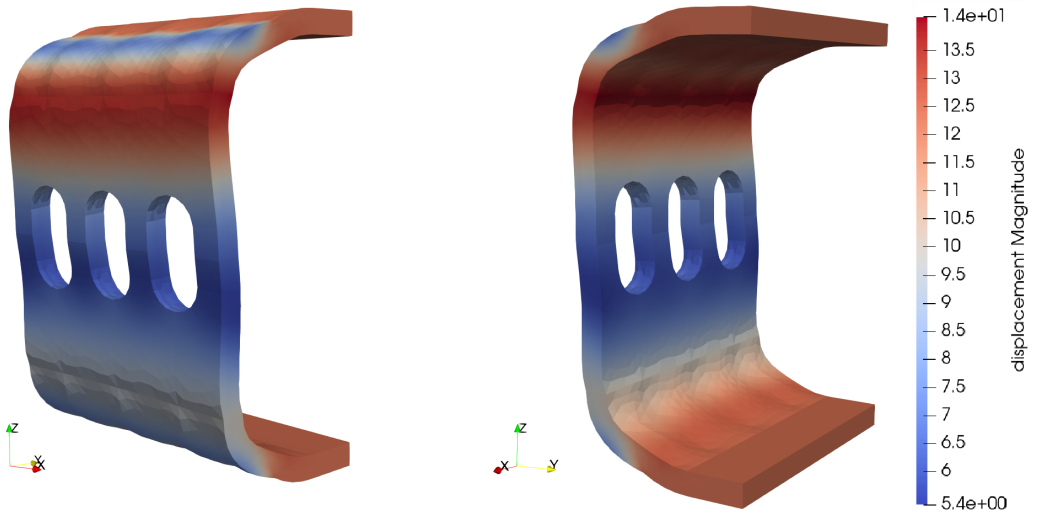


Figure 7.2: Simulation results for the compression of the mock-up airplane cabin based on the composite material. Colors represent the norm of the deformation vectors in centimeters (cm).

Generally, in Figure 7.2, the distribution of the norm of the deformation vectors does not indicate any difference between the inner and outer layer of the cabin wall. Based on these results, the linear elasticity model and the beam frame model do not cause different deformation behavior in the respective sections of the geometry.

To emphasize the different mechanical behavior between both layers of the cabin wall, we generate a representation of the distribution of the von Mises stress within the geometry. The von Mises stress distribution for the computed solution is presented in Figure 7.3. Here, the left image shows the distribution on the outer surface of the cabin wall and the right image shows the stress distribution on the inner surface. For both images, the coloring of the geometry refers to the von Mises stress. With regard to the stress values, we see a clear difference between the parts of the geometry in which the stresses are computed based on linear elasticity and the parts of the geometry based on the beam frame model. The stress values that occur within the outer layer of the cabin wall are significantly higher than the stresses computed for the inner section. A similar relation between the stresses resulting

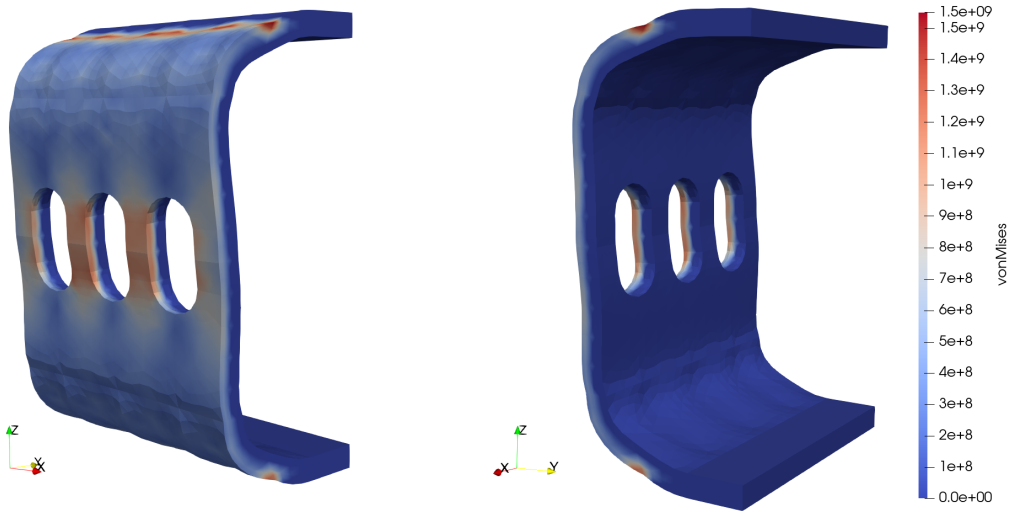


Figure 7.3: Distribution of the von Mises stress values for the compression of the mock-up airplane cabin based on the composite material.

from the porous material and the stresses resulting from a continuous solid would also be expected in a real-world experiment. Note that the distribution of the von Mises stresses in Figure 7.3 show some regions with high stress values in the upper and lower parts of the cabin wall geometry. These peaks in the von Mises stress are largely influenced by the design of the mock-up airplane cabin. In a real-world application, the cabin wall of an airplane would be significantly more rounded in shape to prevent such peaks in the stress values.

In addition to the previous analyses of the simulation results regarding the composite aerogel material, we compare the results to a simulation that is fully based on linear elasticity. This means that we consider the same geometry of the mock-up airplane cabin and for each finite element given in the macroscopic mesh, the stress-strain relationship is computed based on the linear elasticity model. For the resulting macroscopic problem, Newton's method shows direct convergence within one iteration with a relative residual of $7.43e-9$. The results of this simulation are presented in Figure 7.4. The left image shows the front of the geometry and

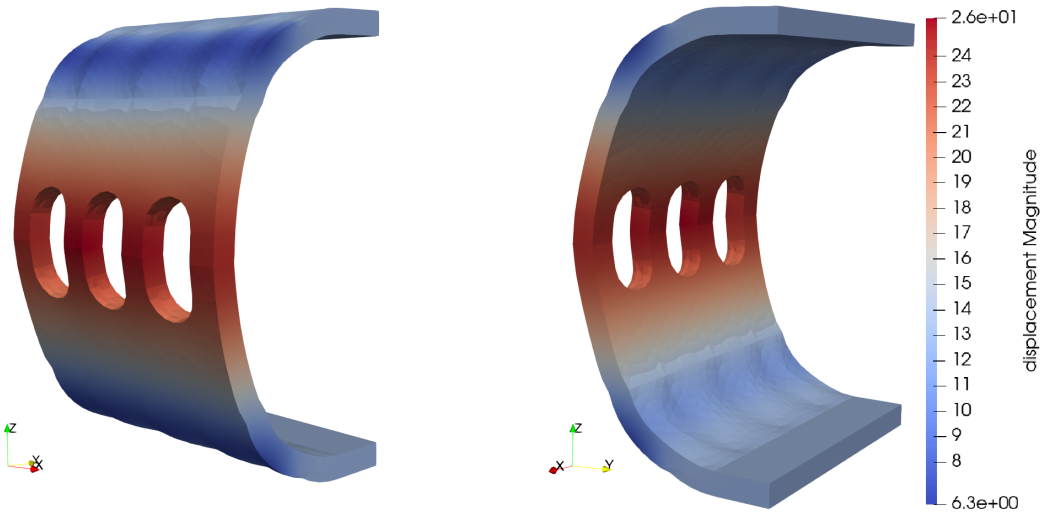


Figure 7.4: Simulation results for the compression of the mock-up airplane cabin based fully on linear elasticity. Colors represent the norm of the deformation vector in centimeters (cm).

the right image shows the back of the cabin. The coloring of the deformed geometry illustrates the distribution of the norm values computed from the deformation vectors. In comparison to the results presented in Figure 7.2, we see significantly higher deformation values close to the center of the geometry in which three rounded holes represent the windows of the airplane. This center section shows relatively large deformations in y-direction which lead to a pronounced rounded shape of the cabin wall in the deformed state. This behavior is very different from the behavior of the composite material, as the outer wall of the cabin appears relatively flat in Figure 7.2.

The results of compressing the mock-up airplane cabin presented in this section show that the multiscale simulation approach is well suited to compute the mechanical behavior of the composite materials. With the individual computation of the stress values within each finite element, it is possible to differentiate between distinct microscopic models for different sections of the macroscopic geometry. For the application of compressing the mock-up airplane cabin, we described the lin-

ear elasticity model. The combination of this basic material model and the beam frame-based multiscale modeling yield direct convergence behavior with Newton's method as non-linear macroscopic solver. The comparison to the simulation result that is solely based on linear elasticity has shown, that the deformation behavior is strongly influenced by including the beam frame model for the inner layer of the geometry. As the macroscopic geometry is not based on real-world components, the insight that the deformation results provide for the specific real-world application regarding airplane design is only limited. However, the description of the multiscale approach provides a general basis for computing the mechanical behavior of aerogel composites in more realistic test cases. The FE^2 framework would also allow the integration of more sophisticated material models than linear elasticity for the simulation of composite materials.

Chapter 8

Conclusion

With regard to the efficient simulation of open-porous materials, we have developed a computational multiscale approach that considers a beam frame structure on the microscopic scale. The developed method is based on the FE^2 framework and requires the solution of a beam frame problem in each macroscopic quadrature point. We have described the creation of the representative volume element (RVE), presented a homogenization approach for the microscopic beam frame model, and described the coupling between the two scales.

In order to reduce the computation time required for the multiscale simulation, we have demonstrated that the microscale computations can be carried out efficiently in parallel and placed a strong focus on the use of machine learning-based surrogate models. Regarding the development of a proper surrogate model that is capable of achieving a significant reduction of the computation time, we have trained a dense feed-forward neural network (NN) to replicate the mechanical behavior of a given beam frame structure. The resulting machine learning model, based on the multi-layer perceptron (MLP) architecture, has been applied within the FE^2 method to replace the beam frame solver and the homogenization computations on the microscopic scale. The performance of the surrogate model has been evaluated based on the deviation of the computed solutions on the macroscopic and on the microscopic scale. The presented results have shown that the MLP architecture is well-suited to reproduce the mechanical properties of a beam

frame structure. Furthermore, the application of the MLP as a surrogate model in the FE^2 framework has resulted in only minor deviations from the original beam frame approach. Additionally, this surrogate approach has achieved a significant reduction of the computational effort required for a simulation.

In addition to the MLP machine learning models that each replace the computations for a single RVE, we considered further NN architectures for training a machine learning-based surrogate model with the stress data resulting from multiple RVEs. These refined models are based on MLP, Mixture of Experts (MoE), and graph neural network (GNN) architectures. We discussed the behavior of the surrogate approaches with regard to the deviation to the beam frame model on the microscopic and macroscopic scale. Although we saw mixed results with regard to the accuracy of the predictions for open-porous materials that have not been included in the training data, the MoE approach provided significant benefits compared to the surrogate model that uses an MLP based on a single RVE. Despite the weak training performance of the GNN architecture within the MoE model, we expect significant potential of this approach to predict the mechanical behavior of open-porous material while taking the pore structure into account. With regard to future work, a refinement of the GNN approach could be investigated to improve the performance of the surrogate model.

As an additional approach to reduce the computation time for the multiscale simulation, we described the parallel implementation of the multiscale method and analyzed the strong scaling behavior of the algorithm. As there are no dependencies between the microscopic computations in each quadrature point for a given iteration of the macroscopic solver, results have shown near-optimal strong scaling of the computation times, consistent with the theoretical expectations.

To evaluate the performance of the derived multiscale approach, we have compared the simulation results to corresponding experimental data obtained from mechanical testing of cellulose aerogels. Various materials and test cases were taken into account for this validation of the numerical method. Additionally, based on real-world aerospace applications, we discussed the use of the multiscale approach for mechanical simulations of an airplane cabin. As aerogel materials do not feature

sufficient mechanical stability for many applications, the use of composite materials can be beneficial. For the purpose of simulating the behavior of a mock-up airplane cabin, we described a computational method for the simulation of composite materials.

The results in this work have shown that the multiscale approach based on the beam frame model on the microscopic scale is well suited to predict the mechanical behavior of open-porous materials like cellulose aerogels. The accuracy of the model decreases when large deformations are considered, which can also include plastic material behavior. To target this deviation between simulation and real-world behavior, a refined definition of the microscopic problem would be required. The development of sophisticated models that include plastic material behavior could be the subject of future research.

Appendix A

Results for Aerogel-based Hole Plates

The two figures presented in this appendix show several snapshots from the simulation of the hole plate geometries. The RVEs used for the simulations are based on the experimental data of cellulose aerogels. The images in Figure A.1 show the simulation results that refer to the cellulose aerogel created from three percent cellulose in the sol-gel process. Similarly, Figure A.2 shows the simulation results for the cellulose aerogel created by using six percent cellulose in the sol-gel process. More details regarding these simulations can be found in Section 6.1.

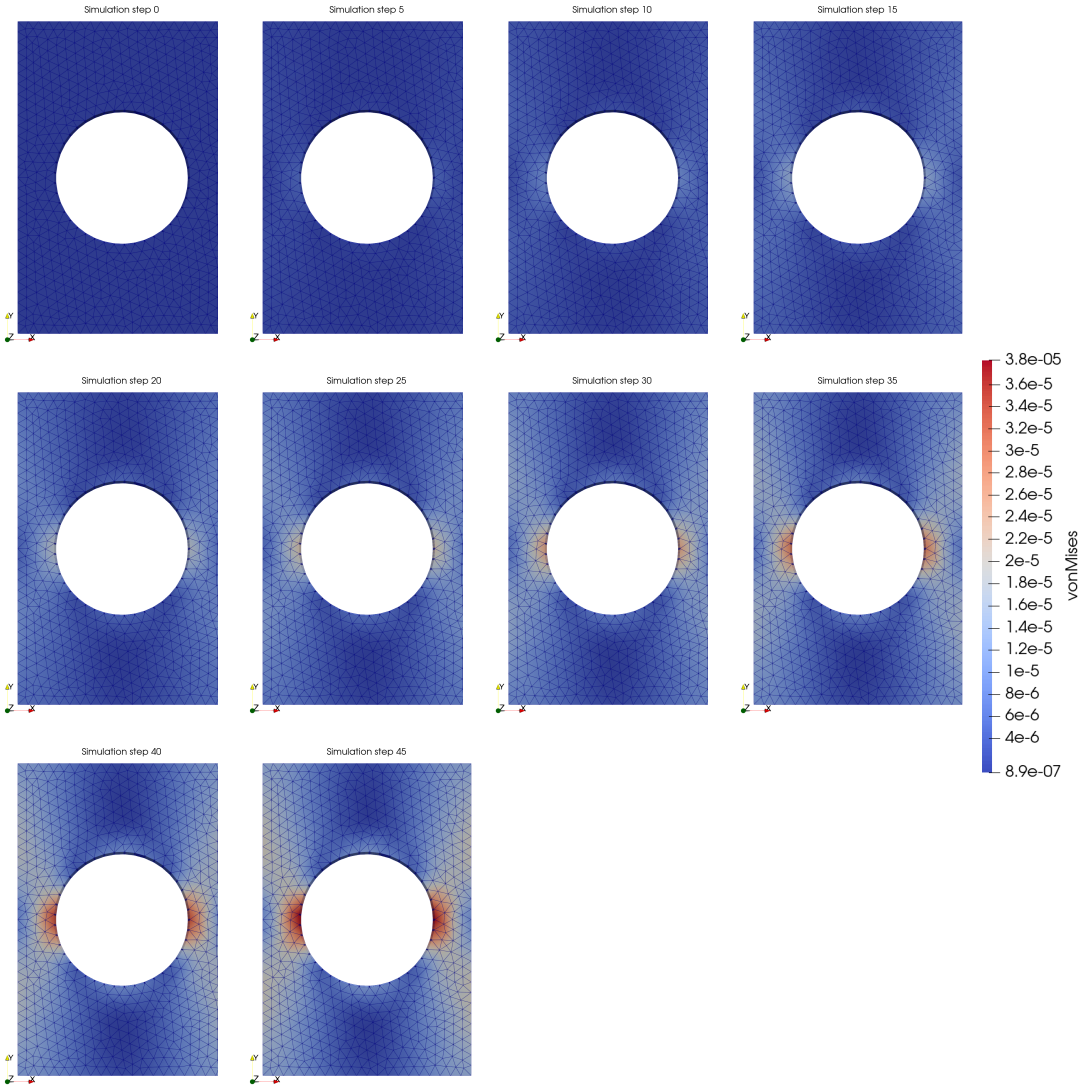


Figure A.1: Simulation results for the hole plate geometry based on the aerogel with three percent cellulose in the starting solution. Ten different time steps of the simulation are presented in the image with the time progressing from left to right. Colors represent the von Mises stress resulting from the deformation in megapascal (MPa).

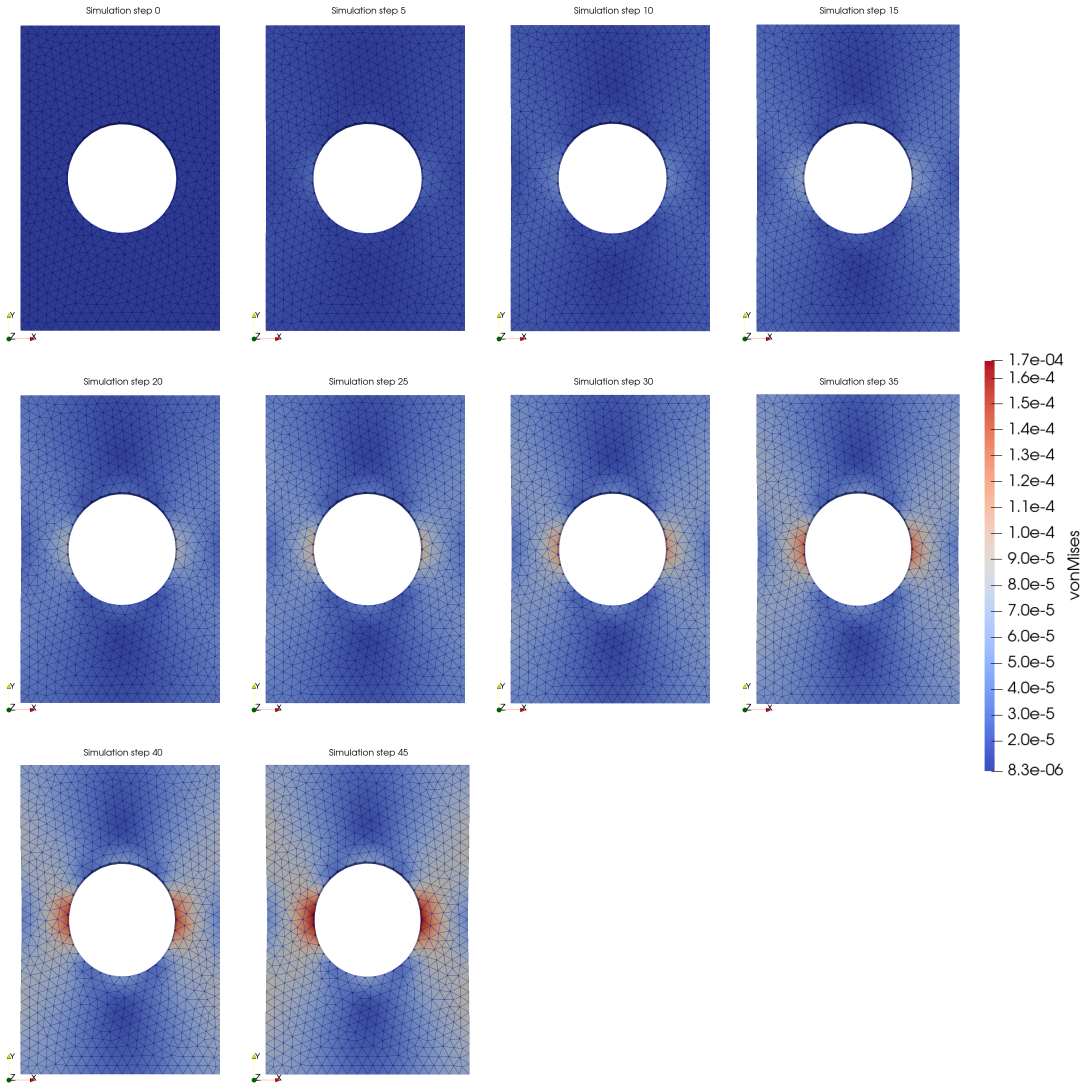


Figure A.2: Simulation results for the hole plate geometry based on the aerogel with six percent cellulose in the starting solution. Ten different time steps of the simulation are presented in the image with the time progressing from left to right. Colors represent the von Mises stress resulting from the deformation in megapascal (MPa).

Appendix B

Results for the Torsion of a Cellulose Aerogel Plate

In this appendix, we present the experimental as well as the simulation results for the torsion of a cellulose aerogel plate. Figure B.1 shows multiple snapshots of the experiment. Each state of the experimental sample is computed with DIC software. The snapshots presented in Figure B.2 show the progression of the simulation results for the torsion test. More details regarding the experimental data and the simulation results can be found in Section 6.2.

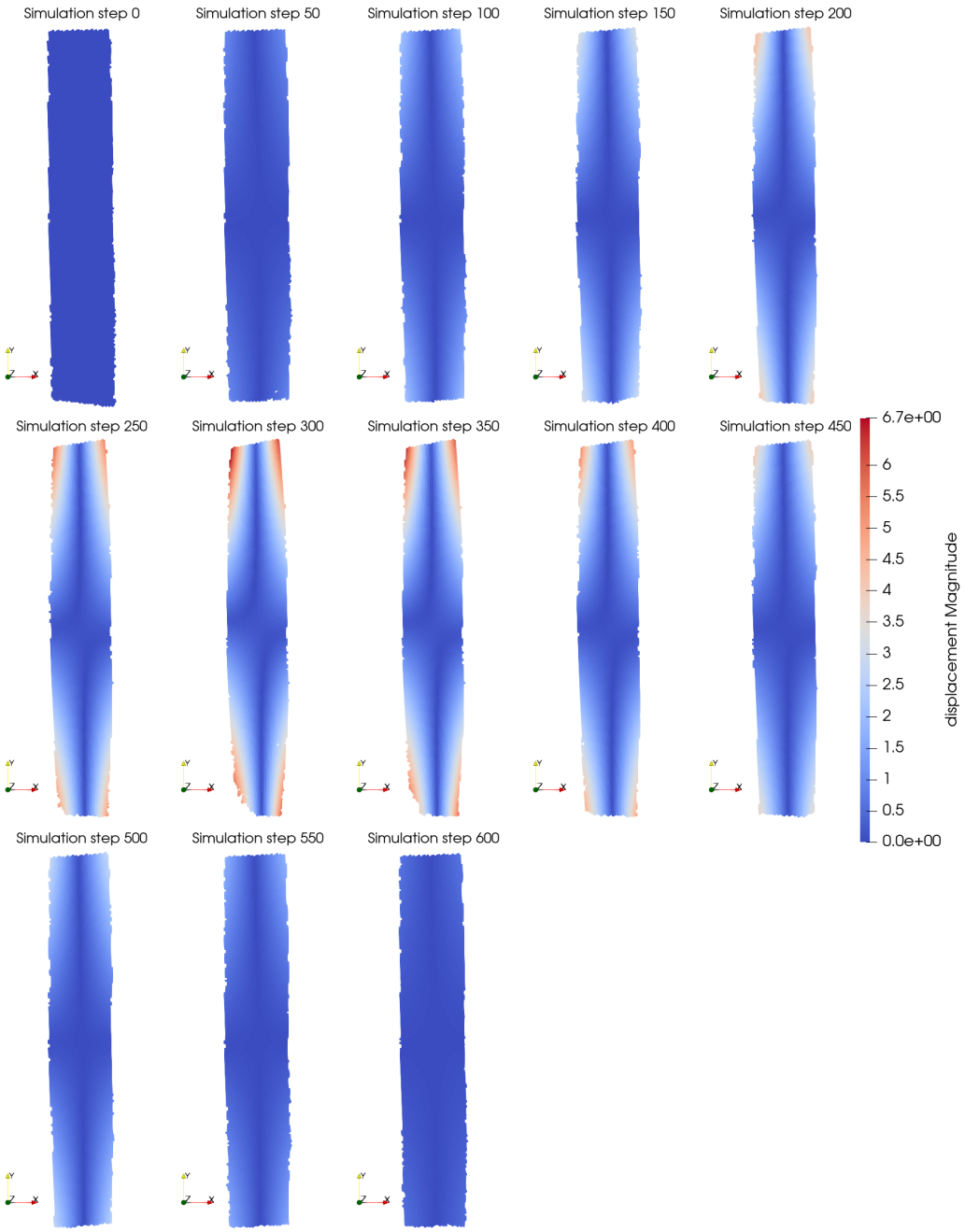


Figure B.1: Experimental results for the torsion of a cellulose aerogel sample. Thirteen snapshots of the experiment are presented in the figure. The coloring of the results refers to the norm over the displacement vector that is given in millimeters (mm).

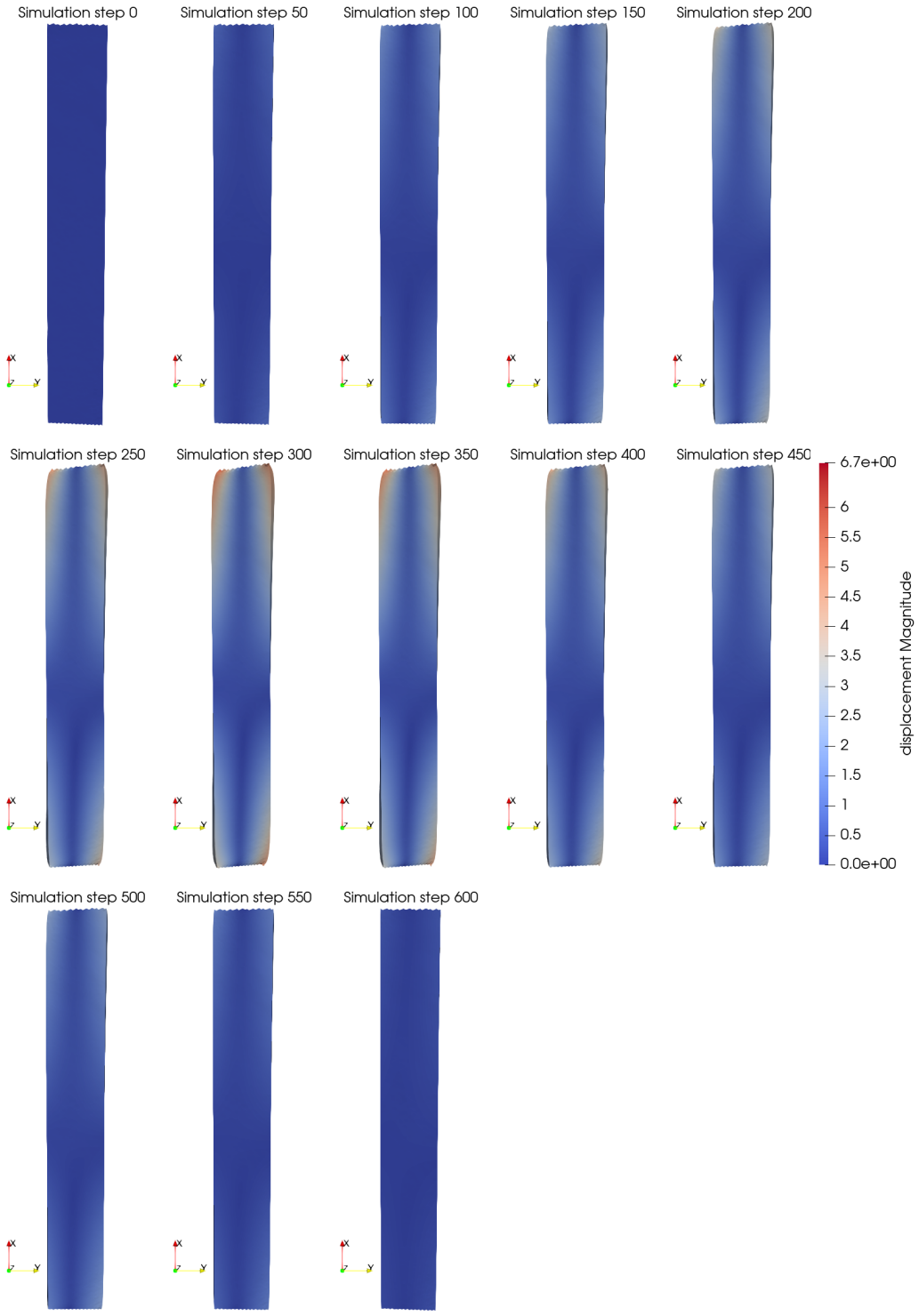


Figure B.2: Simulation results for the torsion of a cellulose aerogel sample. Thirteen snapshots of the simulation are presented in the figure. The coloring of the results refers to the norm over the displacement vector that is given in millimeters (mm).

Bibliography

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. arXiv preprint arXiv:1603.04467. 2015.

[2] A. Abdulle, W. E, B. Engquist, and E. Vanden-Eijnden. "The Heterogeneous Multiscale Method". In: *Acta Numerica* 21, pp. 1–87, May 2012.

[3] A. Altmann, L. Toloşi, O. Sander, and T. Lengauer. "Permutation Importance: A Corrected Feature Importance Measure". In: *Bioinformatics* 26.10, pp. 1340–1347, Apr. 2010.

[4] L. Armijo. "Minimization of Functions Having Lipschitz Continuous First Partial Derivatives." In: *Pacific Journal of Mathematics* 16, pp. 1–3, 1966.

[5] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. Curfman McInnes, K. Rupp, B. F. Smith, S. Zampini, and H. Zhang. *PETSc Users Manual*. Tech. rep. ANL-95/11 - Revision 3.6. Argonne National Laboratory, 2015.

[6] S. Balay, W. D. Gropp, L. C. McInnes, and B. F. Smith. “Efficient Management of Parallelism in Object Oriented Numerical Software Libraries”. In: *Modern Software Tools in Scientific Computing*. Birkhäuser Press, pp. 163–202, 1997.

[7] E. P. Barrett, L. G. Joyner, and P. P. Halenda. “The Determination of Pore Volume and Area Distributions in Porous Substances. I. Computations from Nitrogen Isotherms”. In: *Journal of the American Chemical Society* 73.1, pp. 373–380, 1951.

[8] R. Becker and A. Sokolow. “Stress Averaging for a Beam Network for Use in Hierarchical Multiscale Framework”. In: *ARL-MR-0887*, 2015.

[9] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. “Algorithms for Hyper-Parameter Optimization”. In: *Advances in Neural Information Processing Systems (NIPS)*. Vol. 24. Curran Associates, Inc., pp. 2546–2554, 2011.

[10] R. Bischof and M. Kraus. “Mixture-of-Experts-Ensemble Meta-Learning for Physics-Informed Neural Networks”. In: *33. Forum Bauinformatik*. Munich, Germany, pp. 317–324, Sept. 2022.

[11] L. S. Blackford, A. Petitet, R. Pozo, K. Remington, R. C. Whaley, J. Demmel, J. Dongarra, I. Duff, S. Hammarling, G. Henry, et al. “An Updated Set of Basic Linear Algebra Subprograms (BLAS)”. In: *ACM Transactions on Mathematical Software* 28.2, pp. 135–151, 2002.

[12] F. E. Bock, R. C. Aydin, C. J. Cyron, N. Huber, S. R. Kalidindi, and B. Klusemann. “A Review of the Application of Machine Learning and Data Mining Approaches in Continuum Materials Mechanics”. In: *Frontiers in Materials* 6, 2019.

[13] D. Braess. *Finite Elemente: Theorie, schnelle Löser und Anwendungen in der Elastizitätstheorie*. 5th. Springer Spektrum Berlin, 2013.

[14] M. Brändel, D. Brands, S. Maike, O. Rheinbach, J. Schröder, A. Schwarz, and D. Stoyan. “Effective Hyperelastic Material Parameters from Microstructures Constructed Using the Planar Boolean Model”. In: *Computational Mechanics* 69.6, pp. 1295–1321, 2022.

[15] R. G. M. Breuls, B. G. Sengers, C. W. J. Oomens, C. V. C. Bouten, and F. P. T. Baaijens. “Predicting Local Cell Deformations in Engineered Tissue Constructs: A Multilevel Finite Element Approach”. In: *Journal of Biomechanical Engineering* 124.2, pp. 198–207, Mar. 2002.

[16] J. S. Bridle. “Probabilistic Interpretation of Feedforward Classification Network Outputs, with Relationships to Statistical Pattern Recognition”. In: *Neurocomputing*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 227–236, 1990.

[17] S. Brody, U. Alon, and E. Yahav. “How Attentive are Graph Attention Networks?” In: *10th Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

[18] C. G. Broyden. “The Convergence of a Class of Double-rank Minimization Algorithms: 2. The New Algorithm”. In: *IMA Journal of Applied Mathematics* 6.3, pp. 222–231, Sept. 1970.

[19] S. Brunauer, P. H. Emmett, and E. Teller. “Adsorption of Gases in Multimolecular Layers”. In: *Journal of the American Chemical Society* 60.2, pp. 309–319, 1938.

[20] R. H. Byrd, J. Nocedal, and Y.-X. Yuan. “Global Convergence of a Class of Quasi-Newton Methods on Convex Problems”. In: *SIAM Journal on Numerical Analysis* 24.5, pp. 1171–1190, 1987.

[21] W. Cai, J. Jiang, F. Wang, J. Tang, S. Kim, and J. Huang. “A Survey on Mixture of Experts in Large Language Models”. In: *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–20, 2025.

[22] R. Chandrasekaran, M. Hillgärtner, K. Ganesan, B. Milow, M. Itskov, and A. Rege. “Computational Design of Biopolymer Aerogels and Predictive Modelling of Their Nanostructure and Mechanical Behaviour”. In: *Scientific Reports* 11, 2021.

[23] E. B. Chemere, T. L. Mhlabeni, W. Mhike, M. L. Mavhungu, and M. B. Shongwe. “A Comprehensive Review of Types, Synthesis Strategies, Advanced Designing and Applications of Aerogels”. In: *Royal Society Open Science* 12.5, May 2025.

[24] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. 4th. Cambridge: The MIT Press, 2022.

[25] J. Dan, J. Sun, J. Li, and S. Shi. “An Adaptive Optimal Selection Approach of the Mixture-of-Experts Model Embedded with PINNs for One-dimensional Hyperbolic Conservation Laws”. In: *Communications in Nonlinear Science and Numerical Simulation* 149, May 2025.

[26] P. Dantu and J. Mandel. *Contribution à l'étude théorique et expérimentale du coefficient d'élasticité d'un milieu hétérogène, mais statistiquement homogène*. Vol. 133. École nationale des ponts et chaussées, pp. 115–146, 1963.

[27] T. A. Davis. “Algorithm 832: UMFPACK V4.3—An Unsymmetric-Pattern Multifrontal Method”. In: *ACM Transactions on Mathematical Software* 30.2, pp. 196–199, June 2004.

[28] DeepSeek-AI. *DeepSeek-V3 Technical Report*. arXiv preprint arxiv:2412.19437. 2024.

[29] A. G. Dement'ev and O. G. Tarakanov. “Effect of Cellular Structure on the Mechanical Properties of Plastic Foams”. In: *Polymer Mechanics* 6.4, pp. 519–525, 1970.

[30] S. Deng, S. Hosseinmardi, L. Wang, D. Apelian, and R. Bostanabad. “Data-driven Physics-constrained Recurrent Neural Networks for Multiscale Damagemodeling of Metallic Alloys with Process-induced Porosity”. In: *Computational Mechanics*, 2024.

[31] P. Deuflhard and A. Hohmann. *Numerische Mathematik 1: Eine algorithmisch orientierte Einführung*. 5th. Berlin, Boston: De Gruyter, 2019.

[32] M. W. M. G. Dissanayake and N. Phan-Thien. "Neural-Network-based Approximations for Solving Partial Differential Equations". In: *Communications in Numerical Methods in Engineering* 10.3, pp. 195–201, 1994.

[33] J. Duchi, E. Hazan, and Y. Singer. "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization". In: *Journal of Machine Learning Research* 12, pp. 2121–2159, July 2011.

[34] B. Eidel. "Deep CNNs as universal predictors of elasticity tensors in homogenization". In: *Computer Methods in Applied Mechanics and Engineering* 403, 2023.

[35] D. Eigen, M. Ranzato, and I. Sutskever. "Learning Factored Representations in a Deep Mixture of Experts". In: *Proceedings of the International Conference on Learning Representations (ICLR)*, 2013.

[36] H. Eivazi, J.-A. Tröger, S. Wittek, S. Hartmann, and A. Rausch. "FE^2 Computations with Deep Neural Networks: Algorithmic Structure, Data Generation, and Implementation". In: *Mathematical and Computational Applications* 28.4, 2023.

[37] European Space Agency. *Report on the Space Economy 2025*. Tech. rep. European Space Agency, Mar. 2025.

[38] Z. Fan, Y. Wu, X. Zhao, and Y. Lu. "Simulation of Polycrystalline Structure with Voronoi Diagram in Laguerre Geometry Based on Random Closed Packing of Spheres". In: *Computational Materials Science* 29.3, pp. 301–308, 2004.

[39] R. Farr and R. Groot. "Close packing density of polydisperse hard spheres". In: *The Journal of Chemical Physics* 131, Dec. 2009.

[40] W. Fedus, J. Dean, and B. Zoph. *A Review of Sparse Expert Models in Deep Learning*. arXiv preprint arXiv:2209.01667. 2022.

[41] Y. Fei, G. Rong, B. Wang, and W. Wang. "Parallel L-BFGS-B algorithm on GPU". In: *Computers & Graphics* 40, pp. 1–9, 2014.

[42] N. Feng, G. Zhang, and K. Khandelwal. "Finite Strain FE^2 Analysis with Data-driven Homogenization Using Deep Neural Networks". In: *Computers and Structures* 263, 2022.

[43] O. Ferludin, A. Eigenwillig, M. Blais, D. Zelle, J. Pfeifer, A. Sanchez-Gonzalez, W. Lok Sibon Li, S. Abu-El-Haija, P. Battaglia, N. Bulut, J. Halcrow, F. M. G. de Almeida, P. Gonnet, L. Jiang, P. Kothari, S. Lattanzi, A. Linhares, B. Mayer, V. Mirrokni, J. Palowitch, M. Paradkar, J. She, A. Tsitsulin, K. Villela, L. Wang, D. Wong, and B. Perozzi. *TF-GNN: Graph Neural Networks in TensorFlow*. arXiv preprint arxiv:2207.03522. 2023.

[44] J. E. Fesmire. "Aerogel insulation systems for space launch applications". In: *Cryogenics* 46.2. 2005 Space Cryogenics Workshop, pp. 111–117, 2006.

[45] F. Feyel. "Multiscale FE^2 Elastoviscoplastic Analysis of Composite Structures". In: *Computational Materials Science* 16, pp. 3233–3244, 1999.

[46] G. Foray, J. H. Randrianalisoa, J. Adrien, and E. Maire. "X-ray Tomography Coupled with Finite Elements, A Fast Method to Design Aerogel Composites and Prove Their Superinsulation Experimentally". In: *Gels* 8.11, 2022.

[47] A. Fout, J. Byrd, B. Shariat, and A. Ben-Hur. "Protein Interface Prediction using Graph Convolutional Networks". In: *Advances in Neural Information Processing Systems (NIPS)*. Vol. 30. Curran Associates, Inc., 2017.

[48] J. Fricke and A. Emmerling. "Aerogels". In: *Journal of the American Chemical Society* 75.8, pp. 2027–2036, 1992.

[49] Friedrich-Alexander-Universität Erlangen-Nürnberg. *Fritz*. visited on 2026-02-20. 2025. URL: `https://doc.nhr.fau.de/clusters/fritz/`.

[50] F. Fritzen, M. Fernandez, and F. Larsson. "On-the-Fly Adaptivity for Nonlinear Twoscale Simulations Using Artificial Neural Networks and Reduced Order Modeling". In: *Frontiers in Materials* 6, 2019.

[51] B. Fu, H. Luo, F. Wang, G. Churu, K. T. Chu, J. C. Hanan, C. Sotiriou-Leventis, N. Leventis, and H. Lu. "Simulation of the Microstructural Evolution of a Polymer Crosslinked Templated Silica Aerogel Under High-Strain-Rate Compression". In: *Journal of Non-Crystalline Solids* 357.10, pp. 2063–2074, 2011.

[52] S. Gajek, M. Schneider, and T. Böhlke. "An FE-DMN method for the multiscale analysis of short fiber reinforced plastic components". In: *Computer Methods in Applied Mechanics and Engineering* 384, 2021.

[53] S. Gajek, M. Schneider, and T. Böhlke. "An FE-DMN method for the multiscale analysis of thermodynamical composites". In: *Computational Mechanics* 69, pp. 1087–1113, 2021.

[54] A. N. Gent and A. G. Thomas. "The Deformation of Foamed Elastic Materials". In: *Journal of Applied Polymer Science* 1.1, pp. 107–113, 1959.

[55] C. Geuzaine and J.-F. Remacle. "Gmsh: A Three-dimensional Finite Element Mesh Generator with Built-in Pre- and Post-processing facilities". In: *International Journal for Numerical Methods in Engineering* 79.11, pp. 1309–1331, 2009.

[56] L. J. Gibson and M. F. Ashby. "The mechanics of three-dimensional cellular materials". In: *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* 382.1782, pp. 43–59, 1982.

[57] L. J. Gibson, M. F. Ashby, G. S. Schajer, and C. I. Robertson. "The mechanics of two-dimensional cellular materials". In: *Proceedings of the Royal Society of London. Series A: Mathematical, Physical Engineering Sciences* 382, pp. 25–42, 1982.

[58] G. H. Golub and C. F. Van Loan. *Matrix Computations*. 4th. Philadelphia, PA: Johns Hopkins University Press, 2013.

[59] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[60] M. Gori, G. Monfardini, and F. Scarselli. “A New Model for Learning in Graph Domains”. In: *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.* Vol. 2, pp. 729–734, 2005.

[61] R. Gulakala, B. Markert, and M. Stoffel. “Graph Neural Network Enhanced Finite Element Modelling”. In: *PAMM* 22.1, Mar. 2023.

[62] A. Gupta, A. Bhaduri, and L. Graham-Brady. “Accelerated multiscale mechanics modeling in a deep learning framework”. In: *Mechanics of Materials* 184, 2023.

[63] J. L. Gurav, I.-K. Jung, H.-H. Park, E. S. Kang, and D. Y. Nadargi. “Silica Aerogel: Synthesis and Applications”. In: *Journal of Nanomaterials* 2010.1, 2010.

[64] G. H. Hardy and E. M. Wright. *An Introduction to the Theory of Numbers.* 6th. Oxford, England: Oxford University Press, July 2008.

[65] F. Hendriks, V. Menkovski, M. Doskar, M. G. D. Geers, and O. Rokos. “Similarity Equivariant Graph Neural Networks for Homogenization of Metamaterials”. In: *Computer Methods in Applied Mechanics and Engineering* 439, 2025.

[66] M. R. Hestenes and E. Stiefel. “Methods of Conjugate Gradients for Solving Linear Systems”. In: *Journal of research of the National Bureau of Standards* 49, pp. 409–435, 1952.

[67] R. Hill. “A self-consistent mechanics of composite materials”. In: *Journal of the Mechanics and Physics of Solids* 13.4, pp. 213–222, 1965.

[68] R. Hill. “Elastic properties of reinforced solids: Some theoretical principles”. In: *Journal of the Mechanics and Physics of Solids* 11.5, pp. 357–372, 1963.

[69] G. Holzapfel. *Nonlinear Solid Mechanics: A Continuum Approach for Engineering.* John Wiley & Sons, Jan. 2000.

[70] T. Hou and Y. Efendiev. *Multiscale Finite Element Methods: Theory and Applications.* New York, NY: Springer New York, 2009.

[71] X. Hou, J. Chen, X. Fang, R. Xia, S. Zhu, T. Liu, K. Zhu, and L. Chen. "Thermal Insulation and Fireproof Aerogel Composites for Automotive Batteries". In: *Gels* 11.10, 2025.

[72] T. J. Hughes. *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Dover Publications Inc., 2000.

[73] H. Imai, M. Iri, and K. Murota. "Voronoi Diagram in the Laguerre Geometry and Its Applications". In: *SIAM Journal on Computing* 14.1, pp. 93–105, 1985.

[74] R. Jacobs, M. Jordan, S. Nowlan, and G. Hinton. "Adaptive Mixtures of Local Experts". In: *Neural Computation* 3, pp. 79–87, Mar. 1991.

[75] W.-Y. Jang, A. M. Kraynik, and S. Kyriakides. "On the Microstructure of Open-Cell Foams and Its Effect on Elastic Properties". In: *International Journal of Solids and Structures* 45.7-8, pp. 1845–1875, 2008.

[76] J. Jeffers and J. Reinders. *Intel Xeon Phi Coprocessor High Performance Programming*. Boston: Morgan Kaufmann, 2013.

[77] A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D. S. Chaplot, D. de las Casas, E. B. Hanna, F. Bressand, G. Lengyel, G. Bour, G. Lample, L. R. Lavaud, L. Saulnier, M.-A. Lachaux, P. Stock, S. Subramanian, S. Yang, S. Antoniak, T. Le Scao, T. Gervet, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed. *Mixtral of Experts*. arXiv preprint arxiv:2401.04088. 2024.

[78] X. Jiang, K. Huang, L. Li, C. Liu, and X. Xiang. "Enhanced Thermal Insulation Capabilities of SiO_2 Aerogel Composite Felts and Their Simulation in Mitigating Thermal Runaway in Power Lithium Batteries". In: *Colloids and Surfaces A: Physicochemical and Engineering Aspects* 720, 2025.

[79] M. Jordan and R. Jacobs. "Hierarchical Mixtures of Experts and the EM Algorithm". In: *Neural Computation* 6, pp. 181–214, Jan. 1994.

[80] D. P. Kingma and J. Ba. "Adam: A Method for Stochastic Optimization". In: *3rd International Conference on Learning Representations (ICLR)*, 2015.

[81] A. Klawonn, S. Köhler, M. Lanser, and O. Rheinbach. “Computational Homogenization with Million-Way Parallelism Using Domain Decomposition Methods”. In: *Computational Mechanics* 65.1, pp. 1–22, 2020.

[82] A. Klawonn, M. Lanser, L. Mager, and A. G. Rege. “Computational Homogenization for Aerogel-like Polydisperse Open-Porous Materials Using Neural Network-based Surrogate Models on the Microscale”. In: *Computational Mechanics*, 2025.

[83] A. Klawonn, M. Lanser, L. Mager, A. G. Rege, and J. Weber-Hamacher. “Mixture of Experts Surrogate Model for the Homogenization of Open-Porous Materials”. In preparation. 2026.

[84] A. Klawonn, M. Lanser, and O. Rheinbach. “EXASTEEL - From Micro to Macro Properties”. In: *inSide - Innovative HPC in Germany* 14, pp. 80–85, Nov. 2016.

[85] A. Klawonn, M. Lanser, and O. Rheinbach. “FE^{2}TI: Computational Scale Bridging for Dual-Phase Steels”. In: *Parallel Computing: On the Road to Exascale, Proceedings of the International Conference on Parallel Computing, ParCo 2015*. Vol. 27. Advances in Parallel Computing. IOS Press, pp. 797–806, 2015.

[86] A. Klawonn, M. Lanser, O. Rheinbach, and M. Uran. “Fully-coupled Micro-macro Finite Element Simulations of the Nakajima Test Using Parallel Computational Homogenization”. In: *Computational Mechanics* 68.5, pp. 1153–1178, 2021.

[87] A. Klawonn, M. Lanser, M. Uran, O. Rheinbach, S. Köhler, J. Schröder, L. Scheunemann, D. Brands, D. Balzani, A. Gandhi, G. Wellein, M. Wittmann, O. Schenk, and R. Janalík. “EXASTEEL: Towards a Virtual Laboratory for the Multiscale Simulation of Dual-Phase Steel Using High-Performance Computing”. In: *Software for Exascale Computing - SPPEXA 2016-2019*. Cham: Springer International Publishing, pp. 351–404, 2020.

[88] D. Knoll and D. Keyes. “Jacobian-free Newton–Krylov methods: a survey of approaches and applications”. In: *Journal of Computational Physics* 193.2, pp. 357–397, 2004.

[89] T. F. Korzeniowski and K. Weinberg. “Data-driven Finite Element Computation of Open-Cell Foam Structures”. In: *Computer Methods in Applied Mechanics and Engineering* 400, 2022.

[90] V. Kouznetsova, W. A. M. Brekelmans, and F. P. T. Baaijens. “An Approach to Micro-macro Modeling of Heterogeneous Materials”. In: *Computational Mechanics* 27.1, pp. 37–48, 2001.

[91] N. B. Kovachki, S. Lanthaler, and A. M. Stuart. “Operator Learning: Algorithms and Analysis”. In: *Numerical Analysis Meets Machine Learning*. Vol. 25. Handbook of Numerical Analysis. Elsevier, pp. 419–467.

[92] P. Kraner Zrim, I. B. Mekjavic, and T. Rijavec. “Properties of Laminated Silica Aerogel Fibrous Matting Composites for Footwear Applications”. In: *Textile Research Journal* 86.10, pp. 1063–1073, 2016.

[93] S. Krenk. *Non-linear Modeling and Analysis of Solids and Structures*. 1st. Cambridge University Press, 2009.

[94] A. N. Krylov. “On the Numerical Solution of the Equation by Which in Technical Questions Frequencies of Small Oscillations of Material Systems are Determined”. In: *News of Academy of Sciences of the USSR* 7.4, pp. 491–539, 1931.

[95] M. J. Kuby. “Programming Models for Facility Dispersion: The p-Dispersion and Maxisum Dispersion Problems”. In: *Mathematical and Computer Modelling* 10.10, 1988.

[96] N. Lange, G. Hütter, and B. Kiefer. “A Monolithic Hyper ROM FE^2 Method with Clustered Training at Finite Deformations”. In: *Computer Methods in Applied Mechanics and Engineering* 418, 2024.

[97] B. A. Le, J. Yvonnet, and Q.-C. He. “Computational Homogenization of Nonlinear Elastic Materials Using Neural Networks”. In: *International Journal for Numerical Methods in Engineering* 104.12, pp. 1061–1084, 2015.

[98] Y. Le Cun, L. Jackel, B. Boser, J. Denker, H. Graf, I. Guyon, D. Henderson, R. Howard, and W. Hubbard. “Handwritten Digit Recognition: Applications of Neural Network Chips and Automatic Learning”. In: *IEEE Communications Magazine* 27.11, pp. 41–46, 1989.

[99] D. C. Liu and J. Nocedal. “On the Limited Memory BFGS Method for Large Scale Optimization”. In: *Mathematical Programming* 45.1-3, pp. 503–528, 1989.

[100] L. Lu, P. Jin, G. Pang, Z. Zhang, and G. E. Karniadakis. “Learning Nonlinear Operators via DeepONet Based on the Universal Approximation Theorem of Operators”. In: *Nature Machine Intelligence* 3.3, pp. 218–229, Mar. 2021.

[101] L. Mager. *Simulation and Experimental Results for Tension of Cellulose Aerogel Samples*. visited on 2026-02-20. Feb. 2026. URL: `https://doi.org/10.5281/zenodo.18378781`.

[102] L. Mager. *Simulation and Experimental Results for Torsion of a Cellulose Aerogel Plate*. visited on 2026-02-20. Feb. 2026. URL: `https://doi.org/10.5281/zenodo.18612206`.

[103] D. Marcheggiani and I. Titov. “Encoding Sentences with Graph Convolutional Networks for Semantic Role Labeling”. In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, pp. 1506–1515, Sept. 2017.

[104] S. Markidis, S. W. D. Chien, E. Laure, I. B. Peng, and J. S. Vetter. “NVIDIA Tensor Core Programmability, Performance & Precision”. In: *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pp. 522–531, 2018.

[105] G. T. Mase, R. E. Smelser, and G. E. Mase. *Continuum Mechanics for Engineers*. 4th. Boca Raton: CRC Press, 2020.

[106] W. McGuire, R. H. Gallagher, and R. D. Ziemian. *Matrix Structural Analysis*. 2nd. Wiley, 2015.

[107] Message Passing Interface Forum. *MPI: A Message-Passing Interface Standard Version 5.0*. Tech. rep. June 2025. URL: `https://www.mpi-forum.org/docs/mpi-5.0/mpi50-report.pdf`.

[108] C. Miehe, J. Schröder, and J. Schotte. "Computational Homogenization Analysis in Finite Plasticity. Simulation of Texture Development in Polycrystalline Materials". In: *Computer Methods in Applied Mechanics and Engineering* 171, pp. 387–418, 1999.

[109] R. v. Mises. "Mechanik der festen Körper im plastisch- deformablen Zustand". In: *Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen, Mathematisch-Physikalische Klasse*, pp. 582–592, 1913.

[110] T. M. Mitchell. *Machine Learning*. 1st. USA: McGraw-Hill, Inc., 1997.

[111] C. Moulinec, G. Houzeaux, R. Borrell, A. Quintanas Corominas, G. Oyarzun, J. Grasset, G. Giuntoli, and M. Vazquez. "A Massively Parallel Multi-Scale FE^2 Framework for Multi-Trillion Degrees of Freedom Simulations". In: *Proceedings of the Platform for Advanced Scientific Computing Conference*. PASC '23. New York, NY, USA: Association for Computing Machinery, 2023.

[112] M. A. Nabian and S. Choudhry. *A Mixture of Experts Gating Network for Enhanced Surrogate Modeling in External Aerodynamics*. arXiv preprint arxiv:2508.21249. 2025.

[113] R. Niekamp, J. Niemann, and J. Schröder. "A Surrogate Model for the Prediction of Permeabilities and Flow Through Porous Media: A Machine Learning Approach Based on Stochastic Brownian Motion". In: *Computational Mechanics* 71.3, pp. 563–581, Mar. 2023.

[114] J. Nocedal and S. J. Wright. *Numerical Optimization*. 2nd. New York, NY, USA: Springer, 2006.

[115] A. Öchsner. *Computational Statics and Dynamics: An Introduction Based on the Finite Element Method*. Cham: Springer International Publishing, 2023.

[116] V. J. Oghenekohwo, A. A. Maamoun, S. Zulfiqar, M. J. Forrester, V. Slovák, T.-P. Wang, and E. W. Cochran. "Unlocking the Potential of Polymeric Aerogels from Food and Agricultural Waste for Sustainable CO_2 Capture". In: *ACS Applied Polymer Materials* 6.1, pp. 638–648, 2024.

[117] W. H. Peters and W. F. Ranson. "Digital Imaging Techniques In Experimental Stress Analysis". In: *Optical Engineering* 21.3, p. 427, June 1982.

[118] B. Polyak. "Gradient Methods for the Minimisation of Functionals". In: *Ussr Computational Mathematics and Mathematical Physics* 3, pp. 864–878, Dec. 1963.

[119] S. S. Ravi, D. J. Rosenkrantz, and G. K. Tayi. "Heuristic and Special Case Algorithms for Dispersion Problems". In: *Operations Research* 42.2, pp. 299–310, 1994.

[120] A. Rege, M. Hillgärtner, and M. Itskov. "Mechanics of Biopolymer Aerogels based on Microstructures Generated from 2-d Voronoi Tessellations". In: *Journal of Supercritical Fluids* 151, pp. 24–29, 2019.

[121] A. Rege, I. Preibisch, M. Schestakow, K. Ganesan, P. Gurikov, B. Milow, I. Smirnova, and M. Itskov. "Correlating Synthesis Parameters to Morphological Entities: Predictive Modeling of Biopolymer Aerogels". In: *Materials* 1, pp. 1–19, 2018.

[122] A. Rege, M. Schestakow, I. Karadagli, L. Ratke, and M. Itskov. "Micromechanical Modelling of Cellulose Aerogels from Molten Salt Hydrates". In: *The Royal Society of Chemistry* 12, pp. 7079–7088, 2016.

[123] A. Rege. "Constitutive Modeling of the Densification Behavior in Open-Porous Cellular Solids". In: *Materials* 14.11, 2021.

[124] A. Rege. "Modeling the Structural, Fractal and Mechanical Properties of Aerogels". In: *Springer Handbook of Aerogels*. Cham: Springer International Publishing, pp. 289–305, 2023.

[125] A. Rege, S. Aney, and B. Milow. “Influence of Pore-size Distributions and Pore-wall Mechanics on the Mechanical Behavior of Cellular Solids Like Aerogels”. In: *Physical Review E* 103.4, 2021.

[126] I. B. C. M. Rocha, P. Kerfriden, and F. P. van der Meer. “Machine learning of evolving physics-based material models for multiscale solid mechanics”. In: *Mechanics of Materials* 184, 2023.

[127] I. B. C. M. Rocha, P. Kerfriden, and F. P. van der Meer. “On-the-fly construction of surrogate constitutive models for concurrent multiscale mechanical analysis through probabilistic machine learning”. In: *Journal of Computational Physics: X* 9, 2021.

[128] D. Rumelhart, G. Hinton, and R. Williams. “Learning Representations by Back-Propagating Errors”. In: *Cognitive Modeling*. The MIT Press, pp. 213–222, Sept. 2002.

[129] Y. Saad and M. H. Schultz. “GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems”. In: *SIAM Journal on Scientific and Statistical Computing* 7.3, pp. 856–869, 1986.

[130] Y. Saad. *Iterative Methods for Sparse Linear Systems*. 2nd. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2003.

[131] A. Salvadori, E. Bosco, and D. Grazioli. “A Computational Homogenization Approach for Li-ion Battery Cells: Part 1 – Formulation”. In: *Journal of the Mechanics and Physics of Solids* 65, pp. 114–137, 2014.

[132] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. Battaglia. “Learning to Simulate Complex Physics with Graph Networks”. In: *Proceedings of the 37th International Conference on Machine Learning*. Vol. 119. Proceedings of Machine Learning Research, pp. 8459–8468, July 2020.

[133] F. Scarselli, M. Gori, A. Tsoi, M. Hagenbuchner, and G. Monfardini. “The Graph Neural Network Model”. In: *IEEE Transactions on Neural Networks* 20.1, pp. 61–80, Jan. 2009.

[134] J. Schröder. "A Numerical Two-Scale Homogenization Scheme: The FE^2-Method". In: *Plasticity and Beyond: Microstructures, Crystal-Plasticity and Phase Transitions*. Vol. 550. Springer International Publishing, pp. 1–64.

[135] J. Schröder. "Homogenisierungsmethoden der nichtlinearen Kontinuumsmechanik unter Beachtung von Stabilitätsproblemen". PhD thesis. Institut für Mechanik (Bauwesen), Lehrstuhl I, Universität Stuttgart, 2000.

[136] C. E. Shannon. "A Mathematical Theory of Communication". In: *Bell System Technical Journal* 27.3, pp. 379–423, 1948.

[137] M. V. Shivaditya, J. Alves, F. Bugiotti, and F. Magoulès. "Graph Neural Network-based Surrogate Models for Finite Element Analysis". In: *21st International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES)*, pp. 54–57, 2022.

[138] J. Skibinski, K. Cwieka, S. Haj Ibrahim, and T. Wejrzanowski. "Influence of Pore Size Variation on Thermal Conductivity of Open-Porous Foams". In: *Materials (Basel)* 12.12, pp. 2485–2494, 2019.

[139] R. J. M. Smit, W. A. M. Brekelmans, and H. E. H. Meijer. "Prediction of the Mechanical Behavior of Nonlinear Heterogeneous Systems by Multi-Level Finite Element Modeling". In: *Computer Methods in Applied Mechanics and Engineering* 155.1, pp. 181–192, 1998.

[140] S. A. Steiner III and A. C. Pierre. "The Story of Aerogel". In: *Springer Handbook of Aerogels*. Cham: Springer International Publishing, pp. 1–50, 2023.

[141] P. Stiller, F. Bethke, M. Böhme, R. Pausch, S. Torge, A. Debus, J. Vorberger, M. Bussmann, and N. Hoffmann. "Large-Scale Neural Solvers for Partial Differential Equations". In: *Driving Scientific and Engineering Discoveries Through the Convergence of HPC, Big Data and AI*. Cham: Springer International Publishing, pp. 20–34, 2020.

[142] M. Sun, S. Zhao, C. Gilvary, O. Elemento, J. Zhou, and F. Wang. "Graph Convolutional Networks for Computational Drug Development and Discovery". In: *Briefings in Bioinformatics* 21.3, pp. 919–935, June 2019.

[143] The MathWorks Inc. *MATLAB (R2023b)*. Natick, Massachusetts, United States, 2023. URL: `https://www.mathworks.com`.

[144] S. Timoshenko. *Theory of Elastic Stability*. 1st. New York: McGraw-Hill, 1936.

[145] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. "Attention is All You Need". In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 5998–6008, 2017.

[146] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. "Graph Attention Networks". In: *6th International Conference on Learning Representations (ICLR)*, 2018.

[147] G. Voronoi. "Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Deuxième mémoire. Recherches sur les parallélloèdres primitifs." In: *Journal für die reine und angewandte Mathematik (Crelles Journal)* 134, pp. 198–287, 1908.

[148] P. Wolfe. "Convergence Conditions for Ascent Methods". In: *SIAM Review* 11.2, pp. 226–235, 1969.

[149] Y. Wu, D. Lian, Y. Xu, L. Wu, and E. Chen. "Graph Convolutional Networks with Markov Random Field Reasoning for Social Spammer Detection". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 34, pp. 1054–1061, Apr. 2020.

[150] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu. "A Comprehensive Survey on Graph Neural Networks". In: *IEEE Transactions on Neural Networks and Learning Systems* 32.1, pp. 4–24, 2021.

[151] F. Yang, W. Wang, Y. Yan, and Y. Dong. "In-situ Synthesis of SiO2 Aerogel/Polyethylene Terephthalate Composite Slices and Their Fibers". In: *Pigment & Resin Technology* 54.2, pp. 312–320, Apr. 2024.

[152] M. Yin, E. Zhang, and G. Karniadakis. “Interfacing Finite Elements with Deep Neural Operators for Fast Multiscale Modeling of Mechanics Problems”. In: *Computer Methods in Applied Mechanics and Engineering* 402, May 2022.

[153] D. M. Young. *Iterative Solution of Large Linear Systems*. Academic Press, 1971.

[154] B. Yu, H. Yin, and Z. Zhu. “Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting”. In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI)*. International Joint Conferences on Artificial Intelligence Organization, pp. 3634–3640, June 2018.

[155] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun. “Graph Neural Networks: A Review of Methods and Applications”. In: *AI Open* 1, pp. 57–81, 2020.