TU Delft

Delft University of Technology

Document Version
Final published version

Citation (APA)
Storm, J. (2026). *Reliable and Data-Efficient Machine Learning for Tractable Multiscale Material Simulations*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:9de68e09-8e8c-485a-bc64-11a98dbe8d63

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

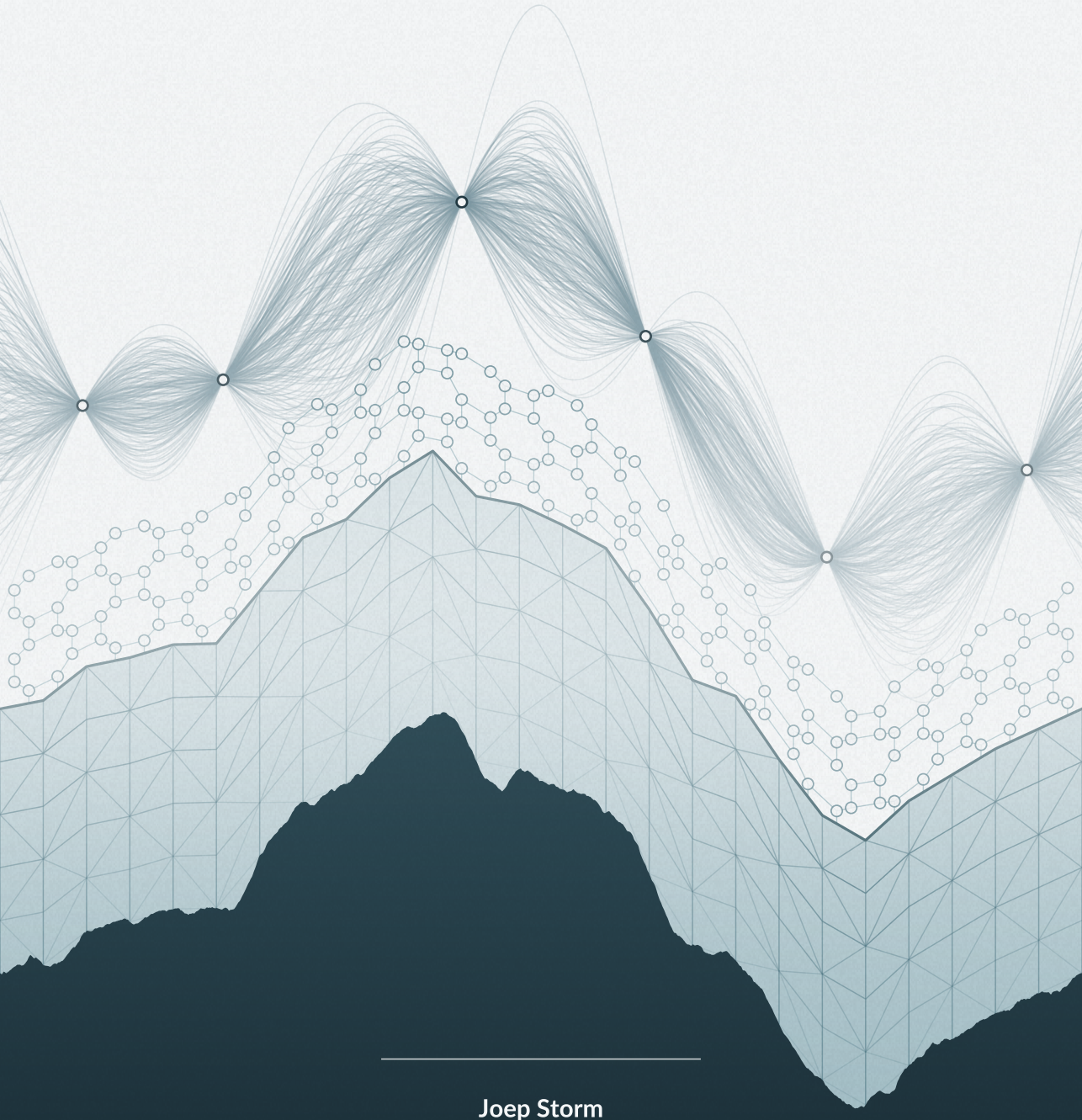
Copyright
In case the licence states “Dutch Copyright Act (Article 25fa)”, this publication was made available Green Open Access via the TU Delft Institutional Repository pursuant to Dutch Copyright Act (Article 25fa, the Taverne amendment). This provision does not affect copyright ownership.
Unless copyright is transferred by contract or statute, it remains with the copyright holder.

Sharing and reuse
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.

Reliable and Data-Efficient Machine Learning for Tractable Multiscale Material Simulations



Joep Storm

Reliable and Data-Efficient Machine Learning for Tractable Multiscale Material Simulations

Reliable and Data-Efficient Machine Learning for Tractable Multiscale Material Simulations

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus, prof. dr. ir. H. Bijl,
chair of the Board for Doctorates
to be defended publicly on
Monday 23 November 2026 at 15:00

by

Joep STORM

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

Rector Magnificus,	chairperson
Dr. ir. F.P. van der Meer,	Delft University of Technology, *promotor*
Dr. I.B.C.M. Rocha,	Delft University of Technology, *copromotor*
Independent members:	
Prof. dr. ir. M. Langelaar	Delft University of Technology
Prof. dr. ir. L. J. Sluys	Delft University of Technology
Prof. dr. J. Bleyer	Ecole nationale des ponts et chaussées, France
Dr. T. Andriollo	Aarhus University, Denmark
Dr. O. Rokoš	Eindhoven University of Technology





Keywords: Multiscale simulations, machine learning, surrogate modeling, solid mechanics
Printed by: Ipskamp printing

Copyright © 2026 by J. Storm

ISBN 978-94-6518-470-8

An electronic copy of this dissertation is available at
`https://repository.tudelft.nl/`.

CONTENTS

Summary **xi**

Samenvatting **xiii**

1. Introduction **1**
1.1. The finite element method 2
1.2. Multiscale simulations 4
1.3. Data-driven simulations 6
1.4. Objectives and outline 8

2. A microstructure-based surrogate model **11**
2.1. Introduction 12
2.2. Background 14
2.3. GNN-based surrogate model 17
2.4. Results - monotonic elasto-plasticity 21
2.5. Non-monotonic elasto-plasticity 26
2.6. Conclusion 38

3. Reliable simulations with inadequate surrogates **41**
3.1. Introduction 42
3.2. Finite Element Analysis 44
3.3. Mixture of Constitutive Models 45
3.4. Results 49
3.5. Conclusion 62
A. Gaussian Process regression 64

4. Microscale-conditioned surrogates for designing multiscale materials **67**
4.1. Introduction 68
4.2. Inspiration: mycelium-woodchip composites 70
4.3. Finite strain mechanics 71
4.4. Microstructure-conditioned PRNNs 72
4.5. Results: conditioning on microscale variables 76
4.6. Results: conditioning on manufacturing variables 85
4.7. Conclusion 88
A. Hyperparameter configurations 89
B. FE^2 validation stress paths 89

5. Conclusion **93**
5.1. Contributions 94
5.2. Future research directions 95

Acknowledgements **101**

Curriculum Vitæ **113**

SUMMARY

Advanced materials promise to cut material use and enable structural designs that conventional materials cannot achieve. Composite materials — materials that combine two or more constituents — such as fiber-reinforced polymers, are an example of an advanced material whose properties can exceed those of traditional building materials. However, developing a new material for a specific structure is a costly process that requires vast experimental trial and error. Virtual testing using computational models has the potential to reduce this cost and accelerate material innovation. Computational methods have been developed for these advanced materials that account for phenomena at a very small scale (the microscale) while simulating a large structure (the macroscale). The high fidelity of these multiscale methods imposes a significant computational burden. Running one such multiscale simulation could scale up to take years, and designing and optimizating a structure often requires simulating many variations. To enable virtual testing and optimization of these materials, we thus require acceleration techniques. A popular avenue to accelerate these simulations is using machine learning.

Data-driven machine learning techniques allow for approximating the simulation outcome at a fraction of the computational cost. The microscale material behaviour can be replaced by a data-driven surrogate model that is then integrated in the macroscale simulation. These data-driven surrogate models can make accurate predictions by learning from precomputed microscale simulations. While promising, this approach faces several challenges: rich microscale information is lost, predictions become unreliable, and training an accurate surrogate model requires large quantities of training data. This dissertation develops methods aimed at addressing each of these challenges.

To enable optimization, a model should predict well for a range of different microstructures. However, most surrogates avoid the microscale completely, and directly predict homogenized quantities for one specific microstructure. We develop a surrogate that instead predicts the rich microscale response for varying microstructures, using a graph neural network that maps onto any microscale mesh. Regardless of the surrogate used, when little data is available, it can become unreliable and lead to inaccurate simulations. We therefore introduce a method that adaptively switches between a fast surrogate and the slow ground-truth model, using the surrogate only in the range on which it has been trained. The uncertainty of a probabilistic surrogate informs the switch, and a phase-field at the macroscale smoothens the transition to increase stability. This approach increases the reliability of data-driven multiscale simulations, leading to accurate simulations even for inadequately trained surrogates. Finally, we enrich an existing surrogate, a physically recurrent neural network, with microstructural descriptors. The strong inductive bias in this model enables it to learn with very few training samples. With this model, we demonstrate that data-driven multiscale simulations can capture high-fidelity microscopic effects and are sufficiently tractable to enable multiscale optimization.

SAMENVATTING

Geavanceerde materialen beloven materiaalgebruik te verminderen en constructieve ontwerpen mogelijk te maken die met conventionele materialen niet haalbaar zijn. Composietmaterialen — materialen die twee of meer bestanddelen combineren — zoals vezelversterkte kunststoffen, zijn een voorbeeld van een materiaal waarvan de eigenschappen die van traditionele bouwmaterialen overtreffen. Het ontwikkelen van een nieuw materiaal is echter een kostbaar proces dat vele experimenten vereist. Virtueel testen met behulp van computermodellen kan deze kosten verlagen en materiaalinnovatie versnellen. Er zijn methoden die rekening houden met verschijnselen op zeer kleine schaal (de microschaal) terwijl een grote constructie (de macroschaal) wordt gesimuleerd, maar de hoge nauwkeurigheid van deze multischaalmethoden vergt veel rekenkracht. Eén zo'n multischaalsimulatie kan jaren duren, en het ontwerpen en optimaliseren van een constructie vereist vaak het simuleren van vele varianten. Om dit toch mogelijk te maken zijn methoden nodig die de rekentijd verkorten. Een populaire route om deze simulaties te versnellen is het gebruik van machine learning.

Datagedreven machine-learningtechnieken maken het mogelijk de uitkomst van de simulatie te benaderen tegen een fractie van de rekenkosten. Het materiaalgedrag op de microschaal kan worden vervangen door een datagedreven surrogaatmodel dat vervolgens wordt geïntegreerd in de macroschaalsimulatie. Deze datagedreven modellen kunnen nauwkeurige voorspellingen doen door te leren van vooraf berekende microschaalsimulaties. Hoewel veelbelovend, kent deze aanpak verschillende uitdagingen: microschaalinformatie gaat verloren, voorspellingen worden onbetrouwbaar, en het trainen van een nauwkeurig surrogaatmodel vereist grote hoeveelheden trainingsdata. Dit proefschrift ontwikkelt methoden die elk van deze uitdagingen aanpakken.

Om optimalisatie mogelijk te maken, moet een model een reeks aan verschillende microstructuren kunnen voorspellen. De meeste surrogaten vermijden de microschaal echter volledig en voorspellen direct gemiddelde waarden voor één specifieke microstructuur. Wij ontwikkelen een surrogaat dat in plaats daarvan het volledige microschaalgedrag voorspelt voor variërende microstructuren, met behulp van een *graph neural network*. Elk surrogaat kan, wanneer weinig data beschikbaar is, onbetrouwbaar worden en tot onnauwkeurige simulaties leiden. Daarom introduceren we een methode die adaptief schakelt tussen een snel surrogaat en het langzame referentiemodel, waarbij het surrogaat alleen wordt gebruikt in het bereik waarop het is getraind. Hiervoor gebruiken we een surrogaat dat zijn eigen onzekerheid schat, en een *phase-field* op de macroschaal om de overgang geleidelijker te maken. Deze aanpak vergroot de betrouwbaarheid van datagedreven multischaalsimulaties, wat leidt tot nauwkeurige simulaties, zelfs voor surrogaten die onvoldoende getraind zijn. Ten slotte verrijken we een bestaand surrogaat, een *physically recurrent neural network*, met microstructurele informatie. De ingebouwde fysische kennis stelt het in staat te leren met zeer weinig trainingsvoorbeelden. Met dit model tonen we aan dat datagedreven simulaties rekening kunnen houden met de microschaal, en snel genoeg zijn om een multischaalontwerp te optimaliseren.

1
INTRODUCTION

A significant part of how we remember societies is by their ability to manipulate materials. Perhaps the best examples of this are the Bronze Age and Iron Age, literally named after material innovations. The Romans are partly remembered for their mastery of concrete, the recipe of which is still being written about to this day (see, for example, Seymour *et al.*, 2023). Similarly, positive visions of the future often depict elegant, slender structures and the ability to shape matter into any desired structure. Also in the present, one can find ample reason for material innovation. Advances that improve the efficiency of designs can reduce material use and emissions, moving us closer to the Sustainable Development Goals (United Nations, 2015).

Consider, for example, the design of a new environmentally friendly material such as a mycelium-woodchip composite (Gantenbein *et al.*, 2022; Wang *et al.*, 2024). By combining two materials, properties can be obtained that exceed those of either constituent. The mycelium can act as a lightweight binding material, and the woodchip can carry compressive loads. Together, they can form material that has a high stiffness-to-weight ratio.

Apart from the properties of each constituent, such as coming from the type of wood used for the chips, the behavior of the composite is influenced by the interactions between the two materials. The size, shape, and orientation of the woodchips all affect the resulting overall material properties. Physically creating and testing each configuration requires significant trial and error and is very time-consuming. In the case of mycelium, the growth of each specimen takes weeks and, as a living material, develops unpredictably. Alternatively, computational tools can be used to accelerate this process. If the properties of each constituent are known, the behavior of the composite can be simulated for any configuration, avoiding the need for extensive material tests.

However, simulating material behavior poses a fundamental tradeoff between fidelity, the level of detail included in the model, and tractability, the ease of handling the simulation. A first-principles approach, where the response is derived from fundamental physics and scaled up to engineering dimensions, remains computationally intractable for practical applications. Instead, we rely on phenomenological *constitutive models*: descriptions of average material behavior, calibrated against experiments rather than derived from the atoms up. In our example, a phenomenological model could be derived from experiments on the mycelium-woodchip composite.

These constitutive models rest on an implicit assumption that the material can be

treated as a homogeneous continuum. While valid for many practical applications, this assumption breaks down when phenomena at lower scales dictate the structural response. This is often the case for inelastic behavior in composites, especially when failure originates from damage, localization, or fracture initiating from interactions of the constituents at a much smaller scale, namely the microscale. These microscale phenomena often make it difficult to accurately capture the constitutive behavior with a closed-form macroscale model. An example of a macroscale structure (in this case a bridge) dependent on the microscale geometry is visualized in Figure 1.1.

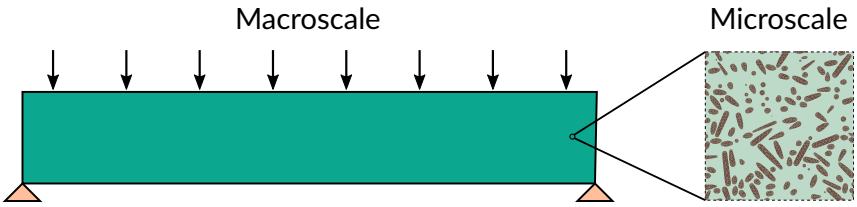


Figure 1.1.: Overview of a multiscale structure.

Accounting for the micromechanical effects in a structural component requires a multiscale approach, analyzing both scales simultaneously. However, this increase in flexibility comes at the cost of tractability, and this method is exceedingly expensive for most practical applications. The aim of this thesis is to make multiscale simulations tractable using data-driven approaches. The rest of this chapter examines the challenges of tractable multiscale simulations in more detail, and discusses how the remaining chapters work towards overcoming them.

1.1. The finite element method

This section briefly summarizes the finite element method (FEM), focusing on the constitutive material models and computational costs. FEM is an approach to solving partial differential equations (PDEs) that discretizes the domain into a mesh of elements. Within each element, the quantity of interest, in our case the displacement field $\hat{\boldsymbol{u}}(\boldsymbol{x})$, is interpolated from nodal values $\boldsymbol{u}$ using shape functions, $\hat{\boldsymbol{u}}(\boldsymbol{x}) \approx \boldsymbol{N}(\boldsymbol{x})\,\boldsymbol{u}$, which reduces the continuous problem to a system of algebraic equations for $\boldsymbol{u}$. For a static problem, this system expresses equilibrium between internal and external forces,

$$\boldsymbol{f}^{\text{int}}(\boldsymbol{u}) = \boldsymbol{f}^{\text{ext}}. \tag{1.1}$$

Assembling $\boldsymbol{f}^{\text{int}}$ requires the stress at quadrature points inside each element. At every such point, the nodal displacements are differentiated into a local strain through the gradient of the shape functions,

$$\boldsymbol{\varepsilon}(\boldsymbol{x}) = \mathcal{B}(\boldsymbol{x}, \boldsymbol{u}), \qquad \mathcal{B} = \frac{\delta \boldsymbol{\varepsilon}(\boldsymbol{x})}{\delta \boldsymbol{u}} \tag{1.2}$$

and a constitutive model $\mathcal{S}$ maps this strain to a stress:

$$\boldsymbol{\sigma} = \mathcal{S}(\boldsymbol{\varepsilon}, \boldsymbol{\alpha}). \tag{1.3}$$

The stresses at the quadrature points are then assembled back into the internal force vector,

$$\boldsymbol{f}^{\text{int}} = \int_{\Omega} \mathcal{B}^{\top} \boldsymbol{\sigma} \, \mathrm{d}\Omega, \tag{1.4}$$

evaluated numerically as a weighted sum over the quadrature points. The constitutive model is where the material behavior enters, and it is evaluated independently at each quadrature point. $\boldsymbol{\alpha}$ represents the internal variables necessary to include complex constitutive behavior, such as for damage or plasticity. For a simple isotropic linear elastic material, $\boldsymbol{\alpha}$ is empty, and this can be evaluated using Hooke's law:

$$\boldsymbol{\sigma} = \boldsymbol{D}\boldsymbol{\varepsilon}, \tag{1.5}$$

where $\boldsymbol{D}$ is a stiffness tensor.

For inelastic behavior, such as plasticity, $\mathcal{S}$ is nonlinear, and the internal variables themselves depend on the strain through a set of evolution equations. Evaluating $\mathcal{S}$ at a quadrature point therefore requires a *local* solve where, given the current strain and the history at the start of the increment, the internal variables are updated by enforcing the constitutive equations at that point. For plasticity, this amounts to returning a trial stress to the yield surface, typically through a return-mapping algorithm (Simo and Taylor, 1986). Because of this nonlinearity, the *global* equilibrium equation must also be solved iteratively, most commonly with a Newton–Raphson scheme. The quadratic convergence rate that makes a Newton–Raphson scheme attractive requires the derivative of the stress with respect to the strain. We compute this consistent tangent at every quadrature point,

$$\boldsymbol{D} = \frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\varepsilon}} \tag{1.6}$$

which linearizes the stress response around the current state, and assemble them into the global tangent stiffness matrix. The resulting order of operations is summarized in Figure 1.2.

Computational cost For most analyses, the initialization and assembly costs are relatively low. The cost of evaluating the constitutive model $\mathcal{S}$ at every quadrature point differs largely based on the type of material model. The predominant cost of a FEM analysis, however, lies in solving the system of equations, and is repeated for every Newton–Raphson iteration per load step. The interpolated field $\mathbf{u}(\mathbf{x})$ converges to the exact solution as the discretization is refined. However, the cost of solving the system of equations grows more than linearly with the number of degrees of freedom in $\boldsymbol{u}$.

There are two families of solvers for solving sparse systems of the form $\mathbf{Ku} = \mathbf{f}$, namely direct and iterative solvers (Davis, 2006; Saad, 2003). A direct solver typically factorizes the matrix $\mathbf{K}$ into products of simpler matrices, solves the simpler systems, and then combines the solutions. Common factorizations are Gaussian elimination and LU factorization. Direct solvers are generally robust and provide exact solutions, but their scaling can be poor for very large systems. With an efficient ordering such as nested dissection, their complexity ranges from $O(n^{1.5})$ for two-dimensional problems to $O(n^2)$ for three-dimensional problems on well-shaped meshes, where n is the number of degrees of freedom (George and Ng, 1988). Alternatively, iterative solvers such as Conjugate Gradient converge on the solution of the system by successive approximations of $\mathbf{u}$. For large sparse systems, the computational cost is $O(k \cdot nnz)$, where k is the

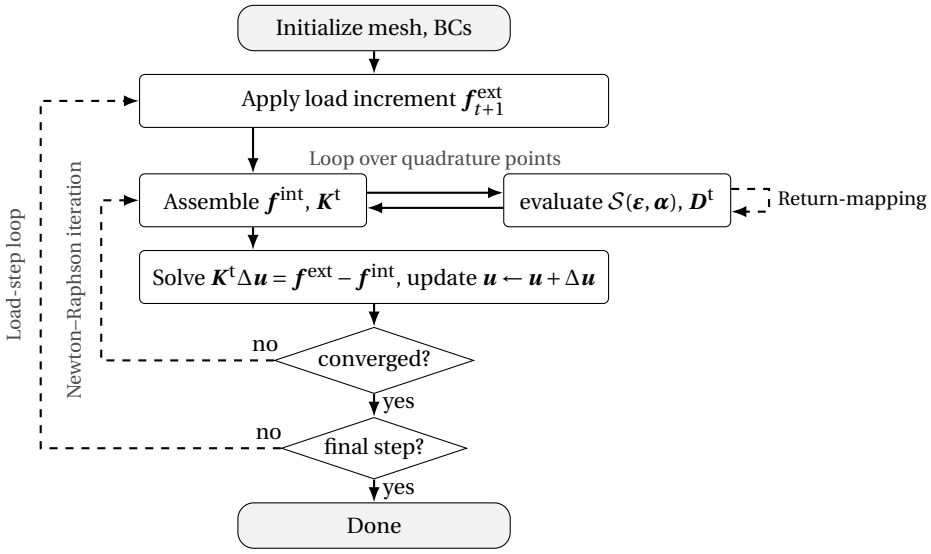


Figure 1.2.: High-level overview of operations of a quasi-static nonlinear FEM analysis.

number of iterations and nnz is the number of non-zero entries in $\mathbf{K}$ (Ferrari, 2024). However, if $\mathbf{K}$ is ill-conditioned, iterative solvers can diverge. To improve the convergence rate of iterative solvers, preconditioners are commonly used to transform the linear system into an equivalent one with more favorable properties. Despite the advantageous scaling of iterative solvers, there remains a limit up to which large systems with many degrees of freedom are tractable.

1.2. Multiscale simulations

For certain materials, such as the mycelium-woodchip composite material introduced earlier, the influence of the microstructure on the macroscale mechanical response is of interest. The initiation of damage at inclusion interfaces and the formation of plastic strain localization bands are examples of microscopic phenomena that influence the macroscopic response, such as the total deformation or when failure occurs. Furthermore, the influence of microscopic parameters on the macroscopic response can be precisely what we want to model for optimization. In this section, we discuss three main routes to include microscopic influences in FEM. We will use Ω to refer to the macroscale domain, and ω to refer to the microscale.

Direct numerical simulation The first approach, direct numerical simulation (DNS), avoids splitting the scales and instead bites the computational bullet by refining the discretization to such an extent that the microstructure features are simply included in the mesh. An example use-case of DNS is to simulate every grain in a steel beam (Bishop *et al.*, 2015). A schematic overview of DNS is visualized in Figure 1.3 b. By directly modeling the microscopic features, DNS allows for the most control with the least assumptions. The microscopic constitutive models are directly included in the

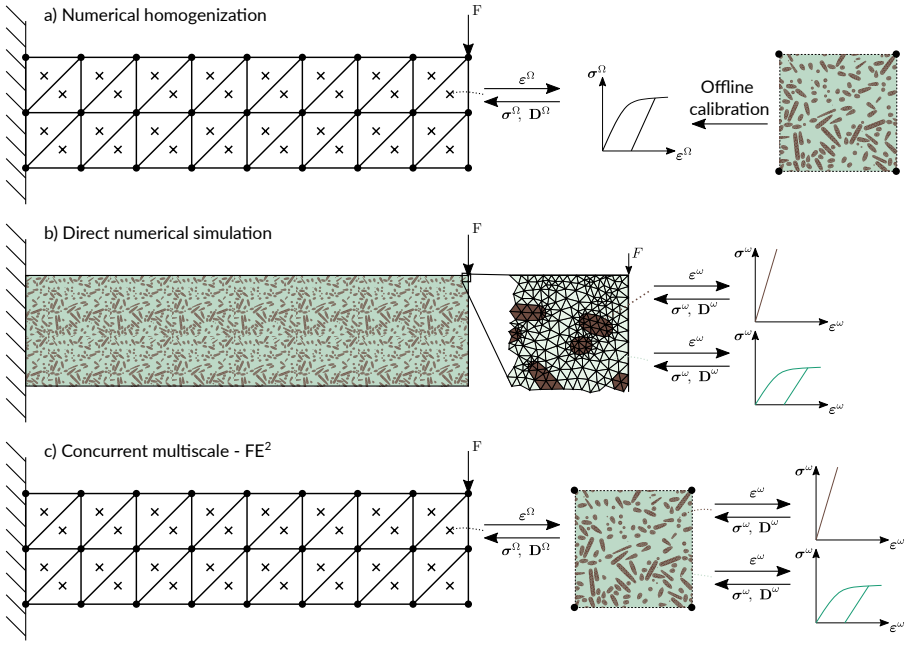


Figure 1.3.: A schematic comparison between numerical homogenization, DNS, and FE^2.

FEM simulation, and additional complex behavior, such as cracking, can easily be implemented. However, the mesh refinement required to model the microscopic features leads to a tremendous number of degrees of freedom. With iterative solvers and *petascale* or *exascale* computing systems, solving this system is still feasible for some specific problems (Vetter, 2017), but the sheer number of degrees of freedom generally makes the problem intractable.

Analytical and numerical homogenization A second approach is to derive the effective macroscopic response from the microstructure. The foundations were laid by mean-field theories for ellipsoidal inclusions (Eshelby, 1957) and the variational bounds for heterogeneous media (Hashin and Shtrikman, 1963). These analytical theories find closed-form estimates of the effective properties directly from the phase properties and inclusion geometry, but extending them to inelastic behavior requires enriching the constitutive model itself. This leads to numerical homogenization: constitutive models whose parameters are calibrated against simulations of microscopic models. Some examples include accounting for the porosity of ductile media (Gurson, 1977), a yield-function dependent on the ellipsoidal shapes of inclusions (Madou and Leblond, 2012), and a homogenized orthotropic plasticity model for fiber-reinforced composites (Vogler *et al.*, 2013). While this is possible to derive for certain properties, it is a laborious process. These models are often fitted to a single set of microscale parameter values, and the derived relation has to be recalibrated if the microstructure is altered.

Concurrent multiscale – FE^2 The third approach, a concurrent multiscale simulation,

keeps the macroscale and microscale separate, as visualized in Figure 1.3 c. Similar to DNS, it also assigns separate constitutive models to fiber and matrix at the microscale, but at the macroscale, the material is still assumed to be homogeneous. This apparent contradiction is resolved by solving the scales separately and passing the averaged microscale response to the macroscale. This is known as a concurrent multiscale simulation, specifically FE^2 (finite-element squared), and works when the macroscale and microscale are well-separated.

FE^2 is more tractable than DNS because, rather than solving a single system with an enormous number of degrees of freedom, many smaller sets are solved instead. In each iteration, the microscale simulations can be independently evaluated in parallel. Domain decomposition brings a similar principle to DNS, but FE^2 remains much more efficient due to the separation of scales. However, FE^2 is still too expensive for most applications. We therefore turn our attention to data-driven techniques that promise to alleviate the computational bottleneck. This is the main aim of this thesis: accelerating FE^2 without sacrificing the benefits obtained from multiscale modeling.

1.3. Data-driven simulations

Data-driven computational mechanics encompasses a broad range of techniques, and there are many ways to categorize them. Figure 1.4 provides an overview of some of the aspects that are relevant to data-driven simulations. Instead of providing an all-encompassing list of techniques, we will highlight a few notable methods relevant for accelerating multiscale simulations.

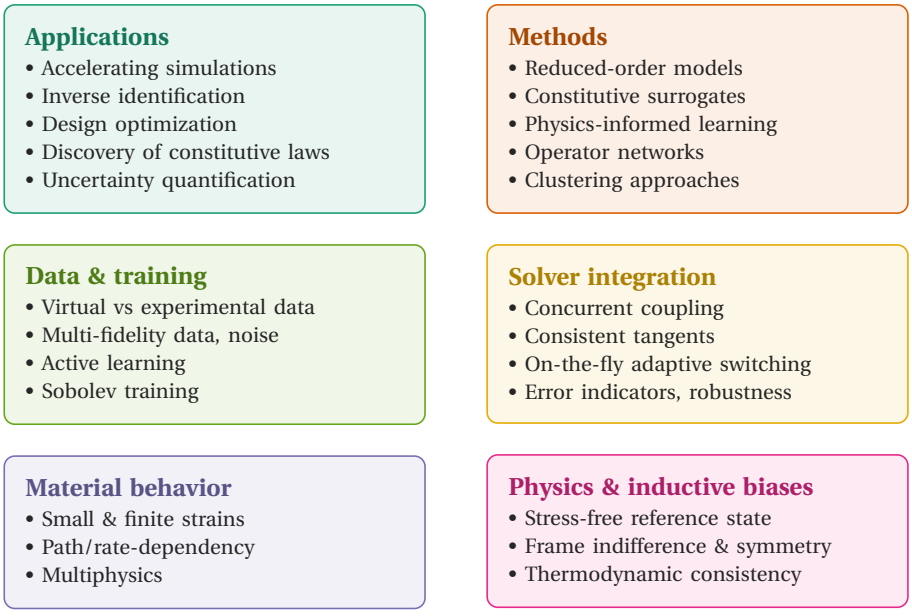


Figure 1.4.: An overview of some aspects in data-driven computational mechanics.

Nearly all data-driven approaches add an *offline* precomputation step to create an approximate model which can then be used *online* during the simulation. They often differ in the assumptions made about the underlying solution, and this leads to different models. For instance, one observation is that the microscale does not typically explore all possible solution fields. This has motivated reduced-order models (ROMs), which assume that the solutions lie close to a low-dimensional manifold which can be found by a linear combination from a handful of precomputed modes using, for example, Proper Orthogonal Decomposition (POD) (Yvonnet and He, 2007). For nonlinear problems, such as when dealing with plasticity, this can still be expensive because every original integration point is used. Therefore, a second hyper-reduction step is often used, which approximates the result based on a subset of cleverly chosen points (Hernandez *et al.*, 2017; Peherstorfer *et al.*, 2014; Ryckelynck, 2009). For highly nonlinear problems, ROMs require many modes for acceptable accuracy, and their speedup is reduced, motiving nonlinear manifold techniques (Lee and Carlberg, 2020). Clustering approaches are similar to hyper-reduction, but can better handle nonlinear behavior by avoiding the POD basis and instead clustering material points that behave similarly and evaluating one point per cluster (Cheng *et al.*, 2019; Liu *et al.*, 2016). What hinders the adoption of these approaches is that they are intrusive: they require fundamental changes to the residual and integration loops of the FEM code.

A non-intrusive alternative is offered by neural network (NN) surrogates, which learn the nonlinear microscale response directly from a dataset of RVE simulations. By directly replacing the constitutive model $\mathcal{S}^{\Omega}$ (Equation 1.3), they require no modification to the macroscale solver. For path-dependent materials, architectures that carry an internal memory are a natural choice, and recurrent neural networks have been applied widely (Borkowski *et al.*, 2022; Ghavamian and Simone, 2019; Mozaffar *et al.*, 2019; Wu *et al.*, 2020b). Convolutional and transformer approaches have been explored as alternatives that process the loading history without a recurrent state (Abueidda *et al.*, 2021; Wang *et al.*, 2020; Wang *et al.*, 2022). A shortcoming of purely data-driven surrogates is that they are data hungry, and the training data is expensive to produce precisely because generating it requires the simulations that the surrogate is meant to replace. Furthermore, they tend to fail when evaluated outside the distribution of loading paths seen during training.

A common remedy is to reintroduce physical knowledge into the architecture. Constitutive artificial neural networks (Linka *et al.*, 2021), thermodynamics-based neural networks (Masi *et al.*, 2021), and physics-augmented formulations (Kalina *et al.*, 2023; Klein *et al.*, 2022a) constrain the models' inputs, architecture, and loss function to respect known principles, such as thermodynamic consistency or material symmetries, which improves robustness and reduces the data requirement. Deep material networks (Liu *et al.*, 2019b), physically recurrent neural networks (Maia *et al.*, 2023), and evolving material models (Rocha *et al.*, 2023) take this idea further, and can be viewed as occupying a middle ground between ROMs and NN-based surrogates. These embed the microscopic constitutive models directly, so that path-dependence and history are handled by the embedded material models rather than learned from data. As a result, they are less data hungry and extrapolate well beyond their training regime. This strong bias comes at the cost of requiring the micromechanical properties and a limited expressivity of the surrogate.

The diversity of models reflects the wide variety of problems that these surrogates aim to address. As a consequence, common benchmarks that fairly compare a range of surrogate models are difficult to establish. The most notable existing benchmarks are in the domain of predicting heterogeneous neo-Hookean material deformation (Lejeune, 2020), fracture (Hamdi and Lejeune, 2026), and broader PDEs (Ohana *et al.*, 2024; Takamoto *et al.*, 2022), but for accelerating multiscale simulations, no good benchmark exists. Despite the lack of benchmarks, several challenges can be identified that hold for a broad range of models. These challenges include handling more complex physics, excessive data requirements, and a lack of robustness. Examples of a lack of robustness are a sensitivity to arbitrary choices during training and a poor performance on loading scenarios unseen during training.

Beyond the choice of architecture, a surrogate only delivers a speedup once it is integrated into a running multiscale solver. A surrogate stands in for the RVE at every macroscale integration point; each macroscale Newton iteration requires not only the homogenized stress but also the consistent tangent. This tangent is typically recovered by automatic differentiation through the network typical for machine learning libraries such as PyTorch (Paszke *et al.*, 2019) and JaX (Bradbury *et al.*, 2018), but integrating these into existing FEM software is not straightforward, and differences in implementation can result in vast differences in speed (Alheit *et al.*, 2026).

Strategies to generate training data also play a crucial role in model performance. Some approaches aim to broadly explore the strain space, such as by using strain paths created using random walks (Wu *et al.*, 2020b) or Gaussian Processes (Logarzo *et al.*, 2021). Alternatively, some work has explored using stress-strain paths from actual multiscale simulations (Ghavamian and Simone, 2019). This significantly reduces the potential sampling space, but restricts the resulting surrogate to only work for the macroscale setup used to generate the data. This can be taken a step further using on-the-fly data construction, where microscale models are simulated for certain integration points during the FE^2 simulation, and their result is used to update the surrogate (Rocha *et al.*, 2021).

Regardless of the method used, the resulting surrogate remains an approximation. Especially for offline generated datasets, a low error on a test set does not necessarily translate to good online performance. For surrogate-based multiscale simulations to be adopted in practice, robustness is necessary. Furthermore, in making multiscale simulations computationally tractable, we should not sacrifice their ability to account for microscopic effects.

1.4. Objectives and outline

The aim of this work is to develop fast and reliable surrogate models that can be trained with little data. These models should still account for the microstructure, making multiscale simulation tractable for demanding applications, such as optimization. Rather than developing one method in vast detail, different methods are introduced that each tackle some current limitation. One such limitation is the inability to account for microstructural variations. For example, if the influence of the microscale geometry is not captured in a surrogate model, then it can't be optimized for. A recurring challenge is

the limited amount of training data. We aim to improve the performance of surrogates in the low-data regime, as well as increase the reliability of a simulation such that accurate results are maintained when using inadequately trained surrogates. The presented methods are tested on various material models, ranging from small-strain plasticity to hyperelasticity with large deformations.

Chapter 2 introduces a graph-based surrogate that predicts the full-field microstructure response, rather than just the homogenized quantities. By predicting how the full-field microscopic strains follow from the macroscopic strain, the microscopic constitutive model and homogenization procedure are retained. A Graph Neural Network is used as it can naturally handle different microstructures.

Chapter 3 aims to give a model-agnostic approach to increase the reliability of surrogate-based simulations. We show that the prediction uncertainty of a probabilistic surrogate, specifically that of a Gaussian Process, can be used as an indicator for when the surrogate will perform poorly. Based on this uncertainty, the simulation locally falls back on the ground truth microscopic model. This adaptive approach is stabilized using a phase-field that smoothens the transition between the different models.

Chapter 4 turns towards multiscale optimization. We enrich a Physically Recurrent Neural Network by conditioning it on microstructural parameters and use it to simulate graded multiscale simulations of a mycelium-woodchip composite. The physics-based architecture introduces a strong bias that allows it to learn with few training samples. Since precise manufacturing of the microscale is challenging, we further show the ability to condition on manufacturing variables instead. The surrogate-based simulation enables efficient optimization of microstructural parameters.

Chapter 5 summarizes the contributions from the preceding chapters. The remaining challenges are discussed and several directions for future research are suggested.

A map is not the territory it represents, but, if correct, it has a similar structure to the territory, which accounts for its usefulness.

Alfred Korzybski, Science and Sanity

2

A MICROSTRUCTURE-BASED SURROGATE MODEL

Simulating the mechanical response of advanced materials can be done more accurately using concurrent multiscale models than with single-scale simulations. However, the computational costs stand in the way of the practical application of this approach. The costs originate from microscale Finite Element (FE) models that must be solved at every macroscopic integration point. A plethora of surrogate modeling strategies attempt to alleviate this cost by learning to predict macroscopic stresses from macroscopic strains, completely replacing the microscale models. In this work, we introduce an alternative surrogate modeling strategy that allows for keeping the multiscale nature of the problem, allowing it to be used interchangeably with an FE solver for any time step. Our surrogate provides all microscopic quantities, which are then homogenized to obtain macroscopic quantities of interest. We achieve this for an elasto-plastic material by predicting full-field microscopic strains using a graph neural network (GNN) while retaining the microscopic constitutive material model to obtain the stresses. This hybrid data-physics graph-based approach avoids the high dimensionality originating from predicting full-field responses while allowing non-locality to arise. In addition, this approach introduces beneficial inductive bias to the model by encoding microscopic geometrical features. By training the GNN on a variety of meshes, it learns to generalize to unseen meshes, allowing a single model to be used for a range of microstructures. The embedded microscopic constitutive model in the GNN implicitly tracks history-dependent variables and leads to improved accuracy. While the microscopic stresses are fully dependent on the microscopic strains, we found it crucial to include both microscopic strains and stresses in the loss function. We demonstrate for several challenging scenarios that the surrogate can predict complex macroscopic stress-strain paths. As the computation time of our method scales favorably with the number of elements in the microstructure compared to the FE method, our method can significantly accelerate FE^2 simulations.

This chapter is published as: Storm, J., Rocha, I. B. C. M., & van der Meer, F. P. (2024). A microstructure-based graph neural network for accelerating multiscale simulations. *Computer Methods in Applied Mechanics and Engineering, 427*, 117001.

2.1. Introduction

There is a global demand for reducing the material cost of structures. One method to achieve this is by creating structures with highly-tailored materials at various scales. However, we currently lack the ability to accurately model the mechanical response of these materials. We typically model these applications using the finite element (FE) method, a technique for solving partial differential equations. The type of material used influences the constitutive relation between stresses and strains. For materials with properties described at a much smaller scale (microscale) than the physical system (macroscale), such as fiber-reinforced polymers, often no accurate analytical relation is known. Instead, concurrent multiscale techniques, such as FE^2, can be used. In FE^2, the constitutive relations are described at the microscale where they are better known, and a full microscopic model is coupled to each integration point of the macroscopic model.

However, multiscale modeling has a high computational cost, therefore data-driven surrogates have been proposed to replace the microscale model directly. Examples of these surrogates are Recurrent Neural Networks (RNNs) (Borkowski *et al.*, 2022; Ghavamian and Simone, 2019; Wu *et al.*, 2020b), temporal convolutional networks (TCNs) (Abueidda *et al.*, 2021; Wang *et al.*, 2022) and attention-based architectures (Wang *et al.*, 2020). These purely data-driven models require a substantial amount of data from expensive microscale simulations to generalize to unseen situations, if they can generalize at all. Introducing physics back into the model can partly overcome this limitation, examples being learning to predict material invariants (Linka *et al.*, 2021), using the invariants themselves as inputs (Klein *et al.*, 2022b), or predicting the internal variables of a constitutive relation (Masi *et al.*, 2021). Including the microscopic material model directly into the data-driven surrogate reduces the necessary training data and enables phenomena such as unloading to be captured without training for it (Liu *et al.*, 2019b; Maia *et al.*, 2023; Rocha *et al.*, 2023).

To obtain training data for these methods, microstructures are simulated for a wide variety of load scenarios. There are different sampling approaches for selecting load paths, such as those from actual FE^2 simulations (Ghavamian and Simone, 2019), monotonically increasing paths (Abueidda *et al.*, 2021), and Gaussian process (GP)-based paths (Abueidda *et al.*, 2021; Maia *et al.*, 2023). Each simulated microstructure contains a rich stress field of data, which is homogenized to a single stress vector to train the models. This means that when using these macroscale-based surrogates, it is impossible to obtain full-field microscopic quantities. Having full-field information could for example be used in an active-learning setting by switching to an FE solver for a few time steps. However, incorporating full-field information in the surrogate model is not trivial with traditional data-driven models.

Next to developments of surrogates predicting a homogenized response for multiscale simulations, numerous works on data-driven techniques focus on predicting full-field single-scale FE problems. Two related challenges here are making a model capable of handling arbitrary domains and avoiding the curse of dimensionality. Physics-informed neural networks (PINNs) embed the partial differential equation (PDE) equilibrium in the loss function, learning based on a few examples and generalizing well (Karniadakis *et al.*, 2021; Raissi *et al.*, 2019). PINNs and graphs can be combined to

get a method closely resembling the FE method, making it possible to minimize the residuals of the weak form of a PDE for linear elasticity (Gao *et al.*, 2022). Recently, the development of operators has gained a lot of attention. Operators are analogous to learning an entire family of PDEs rather than a single instance of the equation. DeepONet, the Fourier neural operator, and the graph neural operator are notable examples (Li *et al.*, 2020a,b; Lu *et al.*, 2019). These operators could have been suitable alternatives for this study.

Alternatively, convolutional neural networks (CNNs) and U-net are other data-driven models that have been used to predict quantities based on a full-field domain (Abueidda *et al.*, 2019; Cang *et al.*, 2018) and generate a full-field solution for elastic cases (Aldakheel *et al.*, 2023; Gupta *et al.*, 2023; Krokos *et al.*, 2022b). Reduced order models have recently been combined with RNNs to recover the full-field solution in a multiscale simulation (Vijayaraghavan *et al.*, 2023). While CNNs are limited to grid-structured data, graph neural networks (GNNs) can deal with arbitrary graphs with varying mesh resolutions, making them well-suited for mesh-based structures (Krokos *et al.*, 2022a). Pfaff et al. propose MeshGraphNets, a GNN with an Encode-Process-Decode architecture followed by an integrator, and use it to predict the dynamics of physical systems such as aerodynamics and structural mechanics (Pfaff *et al.*, 2020). This GNN-based method predicts the relevant quantity for each element based on information of the surrounding elements. Spreading this information around the mesh happens using Message Passing Layers (MPLs). By increasing the number of MPLs, information spreads faster, but too many MPLs can lead to all representations becoming very similar, a problem known as over-smoothing (Li *et al.*, 2018). Both dynamic and static time-independent problems can require fast propagation of information, and adding multiple resolutions of nodes or random augmentations allows information to spread with fewer MPLs, improving accuracy (Cao *et al.*, 2022; Fortunato *et al.*, 2022; Gladstone *et al.*, 2023; Li *et al.*, 2020c; Lino *et al.*, 2021). These GNNs for predicting physical systems generally only predict a single time step ahead during training. To prevent the predictions from diverging during inference over multiple time steps, a Gaussian or prediction-based noise can be added to the inputs during training (Brandstetter *et al.*, 2022; Pfaff *et al.*, 2020). In addition, GNNs have been used to capture multiple steps of linear hardening plasticity under unidirectional tension for simple 1-fiber microstructures (Maurizi *et al.*, 2022), to create a lower-dimensional embedding of the plastic state (Vlassis and Sun, 2023), and for crystal plasticity in 3D microstructures (Frankel *et al.*, 2022). However, none of the aforementioned models is capable of obtaining full-field microscopic quantities for arbitrary inelastic load paths for microstructures with any distribution.

In this work, we propose a surrogate model that makes full-field microscopic predictions for multiscale simulations. We create a dual graph over the mesh of the microstructure — essentially connecting the integration points — and opt for a GNN model. The direct outputs are microscopic strains, and we embed the microscopic material constitutive relations inside the model to obtain stresses. We then in turn use standard computational homogenization to obtain macroscopic stresses. By including the material model, and training on both the strains and the stresses, we learn to predict complex load scenarios for arbitrary meshes more accurately than without the material model. Instead of making one-step ahead predictions during training and adding

noise to the inputs to improve stability, we train on predicting multiple time steps consecutively. The surrogate acts similarly to an autoregressive model by using its strain predictions as inputs for the next time step, being analogous to an RNN with strong physical interpretability due to the material model implicitly updating and preserving history-dependent variables between time steps. Furthermore, by introducing robust geometric inductive bias, our surrogate can successfully generalize to unseen mesh sizes.

We organize the paper as follows. Section 2.2 provides the essential background information on concurrent multiscale simulations and GNNs. In Section 2.3, we introduce our GNN-based surrogate model and provide a comprehensive description of its architecture and the process of training it. Section 2.4 presents a test of the model on a scenario with monotonic loading conditions. To do so, we start the section with an in-depth discussion on model selection, a critical aspect of developing a deep learning surrogate that is frequently left unaddressed. Drawing parallels with the study by (Maia *et al.*, 2023), we examine if the model can predict unloading without being specifically trained for it. This motivates the subsequent Section 2.5, where we train the model on non-monotonic data. Then, we explore the impact of the embedded material model, demonstrate the model's capability to make predictions for various mesh sizes and for more time steps than the model has seen during training, and compare its computational cost to classical FE simulations. Finally, Section 2.6 presents our conclusions.

2.2. Background

This section introduces single-scale FE methods before discussing multiscale FE methods and their challenges, which surrogate models aim to overcome. Then, we present a brief overview of GNNs, which we use as the basis for the surrogate model.

2.2.1. Single-scale FE analysis

We aim to solve a PDE on a domain Ω subject to boundary conditions, finding the resulting displacement field $\mathbf{u}$. The domain Ω is discretized into a number of elements, and the solution in this domain must satisfy:

$$\nabla \cdot \boldsymbol{\sigma}^{\Omega} = \mathbf{0}, \tag{2.1}$$

where $\boldsymbol{\sigma}$ is the stress and $\nabla\cdot$ indicates the divergence operator. We satisfy this relation using the following equations. The displacement is related to the strain ε as:

$$\boldsymbol{\varepsilon}^{\Omega} = \frac{1}{2}\left(\nabla \mathbf{u}^{\Omega} + \left(\nabla \mathbf{u}^{\Omega}\right)^{T}\right), \tag{2.2}$$

and the strain and stress relate via a constitutive model:

$$\boldsymbol{\alpha}_t^{\Omega} = \mathcal{A}^{\Omega}\left(\boldsymbol{\varepsilon}_t^{\Omega}, \boldsymbol{\alpha}_{t-1}^{\Omega}\right) \tag{2.3}$$

$$\boldsymbol{\sigma}_t^{\Omega} = \mathcal{S}^{\Omega}\left(\boldsymbol{\varepsilon}_t^{\Omega}, \boldsymbol{\alpha}_t^{\Omega}\right). \tag{2.4}$$

For history-dependent problems, an internal material variable $\boldsymbol{\alpha}$ is used which evolves over time, for example, to consider elasto-plastic problems. The functions $\mathcal{A}^\Omega$ and $\mathcal{S}^\Omega$ are material-dependent expressions. In the nonlinear case, the resulting displacement $\mathbf{u}$ cannot be solved for directly, and an iterative Newton-Raphson optimization scheme is used instead to find a solution that satisfies Equations (2.1) to (2.4).

2.2.2. Concurrent multiscale analysis

For composite materials with multiple constituents, a direct numerical simulation (DNS) approach involves explicitly modeling the microstructure and assigning different constitutive models for different points in Ω. However, if the microstructural length scales are much smaller than the size of the domain of interest, DNS becomes extremely computationally expensive. In such cases, homogenization is necessary. Finding accurate expressions for $\mathcal{A}^\Omega$ and $\mathcal{S}^\Omega$ can be challenging. Alternatively, a concurrent multiscale analysis can be performed, where we replace the analytical expressions for $\mathcal{A}^\Omega$ and $\mathcal{S}^\Omega$ with solving a boundary value problem on a periodic microscopic domain ω, as depicted in Figure 2.1. This approach is valid if the scales can be separated, *i.e.* if $\omega \ll \Omega$. To connect these scales, we assume that the microscopic displacements $\mathbf{u}^\omega$ can be related by a linear contribution proportional to the macroscopic strains $\boldsymbol{\varepsilon}^\Omega$ and a periodic fluctuation term $\tilde{\mathbf{u}}^\omega$ as:

$$\mathbf{u}^\omega = \boldsymbol{\varepsilon}^\Omega \mathbf{x}^\omega + \tilde{\mathbf{u}}^\omega, \tag{2.5}$$

where $\mathbf{x}^\omega$ is the microscopic coordinate vector. Eqs. (2.1-2.4) then hold for the microscale ω where the relations for each constituent are accurately known. Periodic boundary conditions are constructed by relating the nodal displacements on the boundary of ω with the nodes on the opposite side of the domain. After the microscopic problem is solved, the stresses are homogenized to obtain the macroscopic quantity of interest:

$$\boldsymbol{\sigma}^\Omega = \frac{1}{|\omega|} \int_\omega \boldsymbol{\sigma}^\omega d\omega. \tag{2.6}$$

The microscale problem together with the scale transition in Eqs. (2.5) and (2.6) take the place of the constitutive model (2.3-2.4) in the macroscopic problem.

While the computational cost can be significantly reduced with respect to DNS, simulating these microscale models for all integration points for every macroscopic time step is still prohibitively expensive for most problems. Therefore, attempts have been made to replace the microscale simulations with surrogate models, which are much faster to evaluate. To train these surrogates, data from these microscale models is required. Obtaining this data can be a costly process in and of itself, making it desirable to minimize the amount of data the surrogate requires.

2.2.3. Graph Neural Networks

The model architecture used in this work is based on a Graph Neural Network (GNN). GNNs are used for various applications where graph-type data is available, and numerous different GNN-based architectures exist. Here, we will provide a concise introduction to the message-passing GNN as is used in this work, for a more thorough introduction we refer to (Hamilton, 2020).

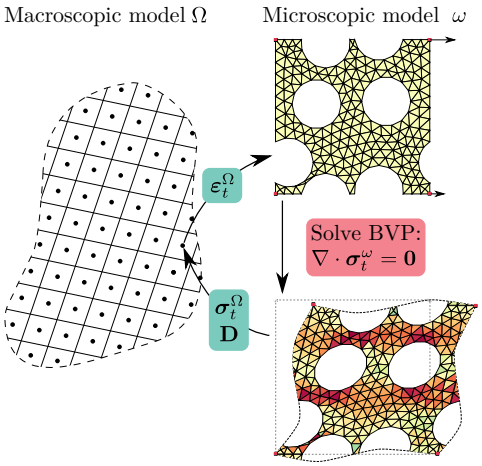


Figure 2.1.: An overview of an FE^2 simulation. A microscopic model ω is solved for every integration point of the macroscopic model Ω.

A graph $\mathcal{G} = (V, E)$ consists of vertices (or nodes) V linked with edges E. The graph can contain information on the edges ($\mathbf{e}_{ij}$), at the nodes ($\mathbf{v}_i$), and at the level of the graph itself, depending on the application. The basic building block of a GNN is a message passing layer (MPL), allowing it to pass information from the nodes and edges around the graph. First, based on all edge states $\mathbf{e}_{ij}$ and nodal states $\mathbf{v}_i$, the corresponding directional edge messages $\mathbf{m}_{ij}$ are computed:

$$\mathbf{m}_{ij} = F^E(\mathbf{e}_{ij}, \mathbf{v}_i, \mathbf{v}_j) \tag{2.7}$$

After updating the messages, every node aggregates the incoming messages from its connected edges (its neighborhood $\mathcal{N}$).

$$\mathbf{m}_i = F^A(\mathbf{m}_{ij} \mid j \in \mathcal{N}) \tag{2.8}$$

The aggregation function F^A is a symmetric operation that can handle incoming messages from an arbitrary number of edges. In practice, this often results in aggregation functions taking the mean or sum of incoming messages. Based on the aggregated messages, the state of each node is updated:

$$\mathbf{v}_i = F^U(\mathbf{v}_i, \mathbf{m}_i). \tag{2.9}$$

The functions F^E, F^A, and F^U are shared between all nodes and edges and generally operate on all node and edge tensors at once. Eqs. (2.7-2.9) represent one MPL, and multiple MPLs can be stacked to process information between more indirectly connected nodes. The number of MPLs is thus a hyperparameter, and when using multiple MPLs, the functions F^E, F^A, and F^U can either be either unique functions or shared between all layers.

2.3. GNN-based surrogate model

In this work, we create a GNN-based surrogate model with the aim of accelerating FE^2 simulations. By using a GNN, we can make full-field predictions while avoiding the high dimensionalities that arise from predicting a full-field response as in a standard NN and giving more flexibility than the grid-based approach of a CNN. Furthermore, the mesh is used by the model as an inductive bias, allowing it to generalize to new meshes. In this section, the details of how we construct this GNN with an embedded material model are presented. We start by introducing the features of the graph, after which we describe the precise architecture and training procedure.

2.3.1. Graph creation

To use a GNN as a surrogate constitutive model for FE^2 simulations, we create a dual graph over the mesh, connecting all the integration points. This is possible because we use elements with a single integration point per element. From this point, we use the word "nodes" to refer to the vertices of the graph, not the nodes of the mesh, unless specified otherwise. Based on this graph, a number of features are inserted for every node. The current microscopic strain state $\boldsymbol{\varepsilon}_t^{\omega}$, the internal variables $\boldsymbol{\alpha}_t^{\omega}$, the macroscopic strains $\boldsymbol{\varepsilon}_{t+1}^{\Omega}$, and any additional geometric features $\mathbf{g}$ are the inputs, which are encoded to $\mathbf{v}_i$. The macroscopic strains $\boldsymbol{\varepsilon}_{t+1}^{\Omega}$ are the same for all nodes. The edge features $\mathbf{e}_{ij}$ are the positional differences in x and y directions between the nodes it connects: $[x_i - x_j, y_i - y_j]$. With bidirectional edges, this thus results in $\mathbf{e}_{ji} = -\mathbf{e}_{ij}$. We are able to retain periodicity by adding extra edges connecting the nodes at opposite sides of the microstructure. The distances for these edges are computed with phantom nodes moved by the length of the microstructure in the corresponding direction, but otherwise, these edges are treated identically to other edges.

Although many GNN-based models for simulating physical processes impose boundary conditions based on forces and displacements on the mesh nodes, this is not straightforward to apply in the context of quasi-static problems as considered here. The boundary condition directly influences the complete microstructure in the first step, requiring an infeasible number of MPLs to propagate this information to all elements. We avoid moving to displacements and forces and instead directly input the macroscopic strains $\boldsymbol{\varepsilon}_{t+1}^{\Omega}$ uniformly to all nodes. This allows us to only consider the microscopic integration points and work with stresses and strains, which is advantageous as embedding the material models is then straightforward. Using this approach, all nodes are directly informed of the load, but a new way to distinguish between the nodes is required, as every point would otherwise always be updated with the same input features, with the undesirable consequence of making all prediction fields uniform.

It is recognized that the cause for stress variations lies in the geometry of the microstructure. Therefore, problem-specific geometric features are provided to the model to allow the locality in the response of the model to arise. In our case, these geometric features $\mathbf{G}$ are the Δx and Δy distance from each node to the center of a fixed number of voids, as visualized in Figure 2.2. For other problems, these features might include the element stiffness, void radius, or volume fraction and should match the application.

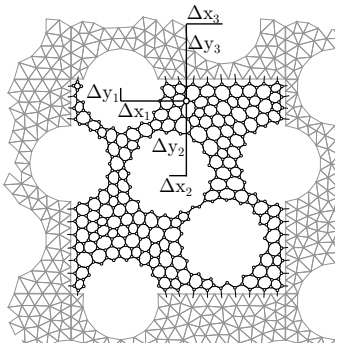


Figure 2.2.: A dual graph is created over the mesh, coinciding with the integration points. Each node of this graph has unique features providing information about a number of the closest surrounding voids through their distances Δx and Δy. We consider the nine closest voids, but we only show distances to three here for clarity. Periodicity is considered in both the void features and the edge connections - nodes at opposite sides of the microstructure are connected.

We are able to consider more voids as features than there are voids in the microscale mesh by accounting for periodicity. In practice, we tile each microstructure in a grid and use a K-nearest neighbors algorithm for each node in the center mesh with respect to the voids. This enables us to include small microstructures with only a few voids, which are cheap to obtain, while still having a sufficient number of features.

2.3.2. Architecture

Based on the graph and the node and edge features described in the previous section, the GNN predicts the strain for the next step $\boldsymbol{\varepsilon}^{\omega}_{t+1}$. The predicted strains are then passed to a purely physics-based constitutive material model $\mathcal{S}^{\omega}$ to compute the stresses $\boldsymbol{\sigma}^{\omega}_{t+1}$, this is the same constitutive model as used in the original micro-model. In this process, the internal variables $\boldsymbol{\alpha}^{\omega}_{t+1}$ are also implicitly obtained. In the multiscale setting, $\boldsymbol{\sigma}^{\omega}_{t+1}$ is homogenized using Eq. (2.6) to obtain $\boldsymbol{\sigma}^{\Omega}_{t+1}$. The updated strains $\boldsymbol{\varepsilon}^{\omega}_{t+1}$, internal variables $\boldsymbol{\alpha}^{\omega}_{t+1}$ and new macroscopic strain input $\boldsymbol{\varepsilon}^{\Omega}_{t+2}$ are then used in the next time step. We present an abstract representation of the complete framework in Figure 2.3. Unlike other GNN-based models that directly predict all stress and displacement-related quantities as network outputs (Gladstone *et al.*, 2023; Pfaff *et al.*, 2020), our proposed method thus retains a physics-based material model. We essentially extend an Encode-Process-Decode architecture Battaglia *et al.*, 2018 to an Encode-Process-Decode-Material architecture, visualized in Figure 2.4. We believe that this offers an advantage by reducing the complexity of the problem to be learned.

We compute the messages as

$$\mathbf{m}_{ij} = F^{E}(\mathbf{e}_{ij}, \mathbf{v}_{j}), \tag{2.10}$$

concatenating the neighboring nodal state with the edge features, and excluding the

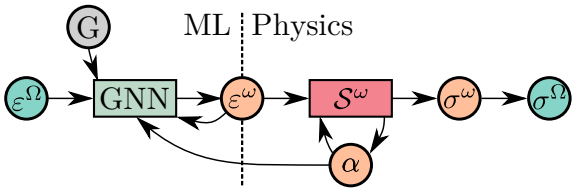


Figure 2.3.: Abstract overview of the multi-step GNN predictions.

state of the current node. We then aggregate the messages by summing them

$$\mathbf{m}_i = \sum_{j \in \mathcal{N}_i} \mathbf{m}_{ij}, \tag{2.11}$$

where $\mathcal{N}_i$ represents the set of neighboring nodes of node i. We concatenate the aggregated message with the current nodal state and pass this through the update function in Eq. (2.9) to obtain the new nodal state. The encoder F^{Enc}, decoder F^{Dec}, edge message F^E, and node update F^U networks are all chosen as multi-layer perceptrons with 2 layers each. In the literature, we observe many slightly varying strategies for using dropout in the GNN architecture. In (Krokos *et al.*, 2022a), dropout is applied after every fully connected layer, (Gladstone *et al.*, 2023) between each MPL block, and in (Vlassis *et al.*, 2020) between every GCN and Dense layer. In our model, we found only minor differences in performance between these strategies, and have chosen to apply dropout inside each update function. We use unique layers in each of the MPLs, as we found this to be more effective than sharing parameters between the layers. Additionally, residual connections are added inside each MPL and from the encoded state to the final MPL output.

2.3.3. Training process

The aim of training the model is to learn the set of parameters of F^{Enc}, F^{Dec}, F^E and F^U that minimizes a loss function $\mathcal{L}$ on the data. The direct output of the GNN-based network is ε^ω, based on which we compute σ^ω through the material model, which we then homogenize to σ^Ω. We introduce the loss function based on these microscopic quantities as:

$$\mathcal{L} = \xi \mathcal{L}_{\sigma^\omega} + (1-\xi)\mathcal{L}_{\varepsilon^\omega}, \tag{2.12}$$

where $\xi \in [0,1]$ is a hyperparameter that allows us to emphasize either the full-field stress or full-field strain. While $\mathcal{L}_{\sigma^\omega}$ and $\mathcal{L}_{\varepsilon^\omega}$ are directly related, we have found it crucial to include both terms in the loss function. If only $\mathcal{L}_{\varepsilon^\omega}$ is considered, small errors for low-value strains can lead to large stress errors. On the other hand, if only $\mathcal{L}_{\sigma^\omega}$ is considered, stresses can potentially still be correct even from considerably wrong strains, e.g if the micromodel exhibits perfectly plastic behavior. This is caused by the nonlinearity of the material model, for a linear elastic microscale material this would not be necessary. Their loss terms are computed as:

$$\mathcal{L}_{\varepsilon^\omega} = \sqrt{\frac{1}{N}\sum_{n=1}^{N} \frac{1}{T_n E_n C} \sum_{t=1}^{T_n}\sum_{e=1}^{E_n}\sum_{c=1}^{C} (\hat{\varepsilon}^{\omega,t}_{n,e,c} - \varepsilon^{\omega,t}_{n,e,c})^2}, \tag{2.13}$$

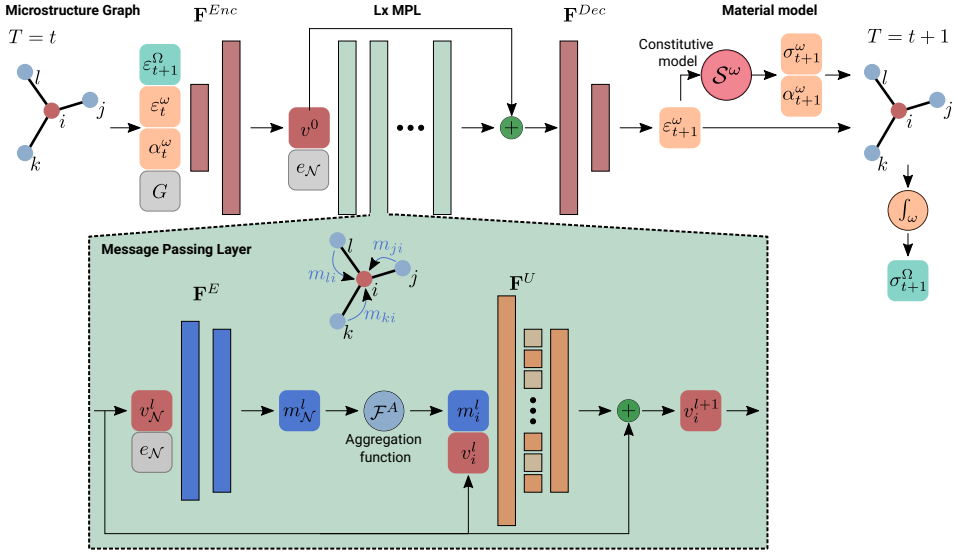


Figure 2.4.: Visualization of the network architecture. The rectangles represent weight layers, where the height roughly indicates their number of weights. The center layer in F^U represents dropout.

$$\mathcal{L}_{\sigma^\omega} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}\frac{1}{T_n E_n C}\sum_{t=1}^{T_n}\sum_{e=1}^{E_n}\sum_{c=1}^{C}(\hat{\sigma}_{n,e,c}^{\omega,t} - \sigma_{n,e,c}^{\omega,t})^2}. \tag{2.14}$$

Here we use N as the number of samples, T_n as the number of time steps, E_n as the number of elements in that sample, and C represents the number of components, which is 3 in the 2D case. In addition to these losses, we introduce the loss of σ^Ω which is the relevant quantity for the macroscale model in a multiscale simulation:

$$\mathcal{L}_{\sigma^\Omega} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\frac{1}{T_n C}\sum_{t=1}^{T_n}\sum_{c=1}^{C}(\hat{\sigma}_{n,c}^{\Omega,t} - \sigma_{n,c}^{\Omega,t})^2}. \tag{2.15}$$

This loss completely depends on σ^ω, and we empirically find it unnecessary to include in the training loss.

Updating the network parameters requires backpropagating the loss $\mathcal{L}$ through all time steps included for the samples. Because the input $\boldsymbol{\alpha}^\omega$ depends on the output of the previous step, and since we also include stress-based targets, the loss needs to be back-propagated through the material model. The constitutive model $\mathcal{S}^\omega$ implicitly evolves $\boldsymbol{\alpha}^\omega$, creating a recurrency. Because of this, the loss at any time step depends on the predictions made for all previous time steps. This leads to a complex computational graph and expensive gradient computations, which scale with the number of time steps during training. An alternative approach is predicting only one time step ahead, but this can lead to unstable rollouts despite low training errors. By artificially inducing noise on the inputs this discrepancy can be reduced (Brandstetter *et al.*, 2022; Pfaff *et al.*, 2020). However, we find this one-step-ahead approach unnecessary here and instead

directly propagate all time steps for a sample during training.

2.3.4. Implementation

The data was generated with an in-house Finite Element software developed using the Jem/Jive (Nguyen-Thanh *et al.*, 2020) open-source C++ library. The microstructures are represented as periodic 2D meshes generated using Gmsh (Geuzaine and Remacle, 2009b). The proposed GNN-based model was implemented using Pytorch Geometric (Fey and Lenssen, 2019). The GNN-based model evaluates all material points in a batch in parallel, and the loss is propagated through them using automatic differentiation. Training and time comparisons for the surrogate are performed on an NVIDIA Tesla V100S 32GB GPU, and FE simulations on an Intel Xeon E5-6248R 24C 3.0GHz, both in the DelftBlue cluster Delft High Performance Computing Centre (DHPC), 2022. We provide code and a dataset to reproduce the results in this paper at `https://github.com/JoepStorm/Microscale-GNN-Surrogate`.

2.4. Results - monotonic elasto-plasticity

We start the analysis by subjecting the micromodel to monotonically increasing strains. To accurately reproduce these predictions, the GNN model needs to demonstrate the ability to predict in an auto-regressive manner for multiple time steps and account for plasticity for a wide variety of microstructures. We discuss the relevance of hyperparameters and perform a model selection study before showing the results for the optimal model. A limitation of the model when only trained on monotonic strain paths is highlighted, motivating the need for training on non-monotonic data, which we discuss in the next section.

2.4.1. Data generation

We generate a different periodic microstructure for each sample (we define a sample as one complete strain path), with a random distribution of one to nine voids (each chosen an equal number of times) but keeping a fixed void volume fraction $V_f = 0.4$. We also sample a random average element size for each mesh, allowing the model to handle a wide range of mesh densities during inference. Then, for each sample, a stress-strain path is generated by selecting a random strain direction and monotonically increasing the strain in this direction, keeping the loading proportional between components. We store the macroscopic and microscopic quantities for each converged time step and discard intermediate Newton-Raphson iterations. We use an elasto-plastic material with von Mises plasticity (J2), a Young's modulus of 3130 $[MPa]$, a Poisson's ratio of 0.37 $[-]$, and yield criterion $\sigma_C = 64.80 - 33.60 \cdot e^{\frac{\varepsilon^p_{eq}}{-0.003407}}$, where ε^p_{eq} is the equivalent plastic strain (the internal variable $\boldsymbol{\alpha}^\omega$ of this material model). In Figure 2.5 we show examples of the resulting macroscopic stress-strain curves after stress homogenization. We generate 4000 samples as training data, 2000 as a validation set, and additional test sets also consist of 2000 samples unless specified otherwise. In sec-

tion 2.5, we will show that 4000 training samples are sufficient by plotting the learning curve for the training sample size for a more complex case.

2.4.2. Model selection

For any deep learning architecture, there are many variables that have a considerable influence on the model performance. Specifically for GNNs, these include the number of MPLs, the number of neurons per layer, the number of layers per MPL, the number and width of layers of the encoder and decoder, the activation functions, normalization layers, residual connections, the aggregation function, dropout layers and their rate, the optimizer, the learning rate, and the batch size. In addition, many of these hyperparameters are interdependent and have many possible values. Performing a thorough model selection study is thus crucial for each problem setting. However, due to the vast number of options and the computational cost of training the model, it is not feasible to consider all combinations. Therefore, a practical approach is necessary for determining the optimal hyperparameters.

Some settings are chosen based on previous work, such as employing two layers per multi-layer perceptron as chosen in MeshGraphNets (Pfaff *et al.*, 2020). For other architectural choices, such as choosing if and where to add normalization layers and residual connections, we compare several options during the initial model creation and then keep this architecture fixed for the rest of the study as described in section 2.3.2. In purely data-driven methods the parameter count, often determined by the amount of neurons per layer, has a significant influence on the inference and training speed of the network. In our case, the computational time is primarily influenced by the material model instead (and scales only with the number of elements). Therefore we can afford a very high parameter count without significantly compromising in computation speed, giving us a highly flexible model, and use the dropout rate as a hyperparameter to regularize the model. The high parameter count follows from using 512 neurons per layer, which is then also the dimensionality of the node state $\mathbf{v}_i$ and messages $\mathbf{m}_i$. For the learning rate, we use 10 warmup epochs to go from $1 \cdot 10^{-5}$ to $2 \cdot 10^{-4}$ and follow this up with an exponential decay with factor 0.998: $2 \cdot 10^{-4} \cdot 0.998^{epoch}$ for a fixed time budget of 105 hours giving around 700 epochs for a typical model. The number of con-

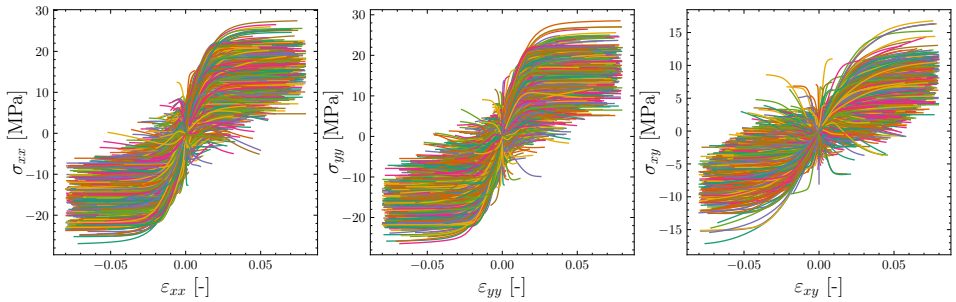


Figure 2.5.: Examples of stress-strain curves obtained from simulating one to nine void microstructures with a monotonic strain increase.

sidered void features is kept at 9, the maximum number of voids included in training samples.

MPLs are the defining feature of GNNs, and the number of MPLs is an important hyperparameter we study. As we are dealing with elasto-plasticity, we find that the tradeoff between strains and stresses using ξ (Eq. 2.12) is also an important hyperparameter to consider. For the number of MPLs, the dropout rate, and ξ we perform several rounds of model selection, where in each round we vary one parameter at a time and use the best value from the previous round as a starting point until the values converge. We train two models per setting, and the resulting errors for each variable in these rounds are shown in Figure 2.6. In both rounds, mainly the value of ξ and the dropout rate have an influence. The small influence of the number of MPLs is surprising and should be investigated further in future work.

Choosing the value for ξ is not straightforward, as the nature of the optimization problem changes as we change the objective by adjusting ξ. Considering only the value of the combined validation loss is therefore insufficient. Instead, we need to consider the components $\mathcal{L}_{\varepsilon^\omega}$ and $\mathcal{L}_{\sigma^\omega}$ separately. Based on these two separate losses, we conclude that a value of $\xi = 0.8$ is a good choice for our problem. Remarkably, we observe that having a value of $0 < \xi < 0.8$ not only decreases $\mathcal{L}_{\sigma^\omega}$ but does so without compromising $\mathcal{L}_{\varepsilon^\omega}$.

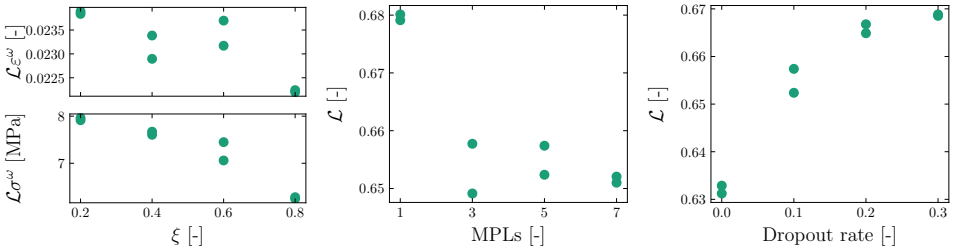
Since the best settings after the second round are the same as used as the base values of the second round, we do not need to perform an additional round. We can use the model with 3 MPLs, $\xi = 0.8$, and a dropout rate of 0.0 for further experiments.

2.4.3. Prediction results

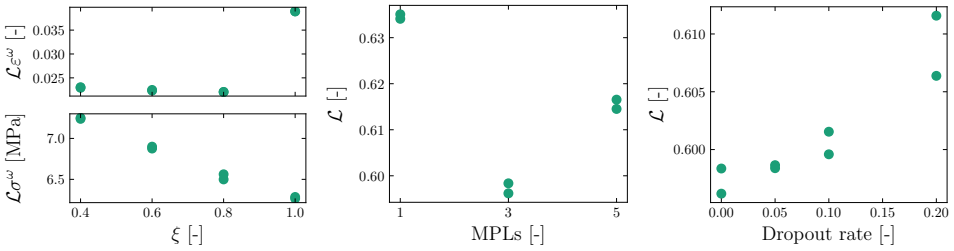
With the model determined, we can now evaluate the prediction results. Figure 2.7 shows all microscopic quantities for a 5-void test sample subjected to a random strain direction after 25 time steps and the resulting macroscopic stress-strain curve, all in comparison with target results from the full micromodel. This example also demonstrates the differences in the strain and stress fields as a result of the nonlinear material model. Even after 25 steps, the surrogate model correctly predicts the areas of high and low strain, and this is well reflected in the equivalent plastic strain ε^p_{eq} field (computed by the physics-based material models $\mathcal{A}^\Omega$ at each integration point). Furthermore, the image highlights the difficulty of the problem to solve: the σ^ω_{xy} field has little recognizable structure, yet the model can predict it well.

2.4.4. Extrapolating for non-monotonic paths

A conventional neural network surrogate, trained solely on the macroscopic strain ε^Ω, cannot differentiate between loading and unloading, resulting in an unloading path identical to the loading path. However, the model has two additional sources of information that could theoretically aid in distinguishing between loading and unloading. Firstly, the model is provided not only with the next macroscopic strain but also with the current microscopic strain and history variables. Secondly, the embedded material model implicitly tracks its history, and recent surrogate models with embedded



(a) Round 1: ξ value (b) Round 1: Number of MPLs (c) Round 1: Dropout rate



(d) Round 2: ξ value (e) Round 2: Number of MPLs (f) Round 2: Dropout rate

Figure 2.6.: The results for two rounds of the hyperparameter study. The initial values chosen as default values for round 1 are $\xi = 0.6$, MPLs=5, and dropout=0.1. For ξ we plot the individual (unnormalized) validation losses $\mathcal{L}_{\varepsilon^\omega}$, $\mathcal{L}_{\sigma^\omega}$ separately, as their combined loss is itself dependent on ξ. Based on round 1, we choose the default values for round 2 as $\xi = 0.8$, MPLs=3, and dropout=0.0.

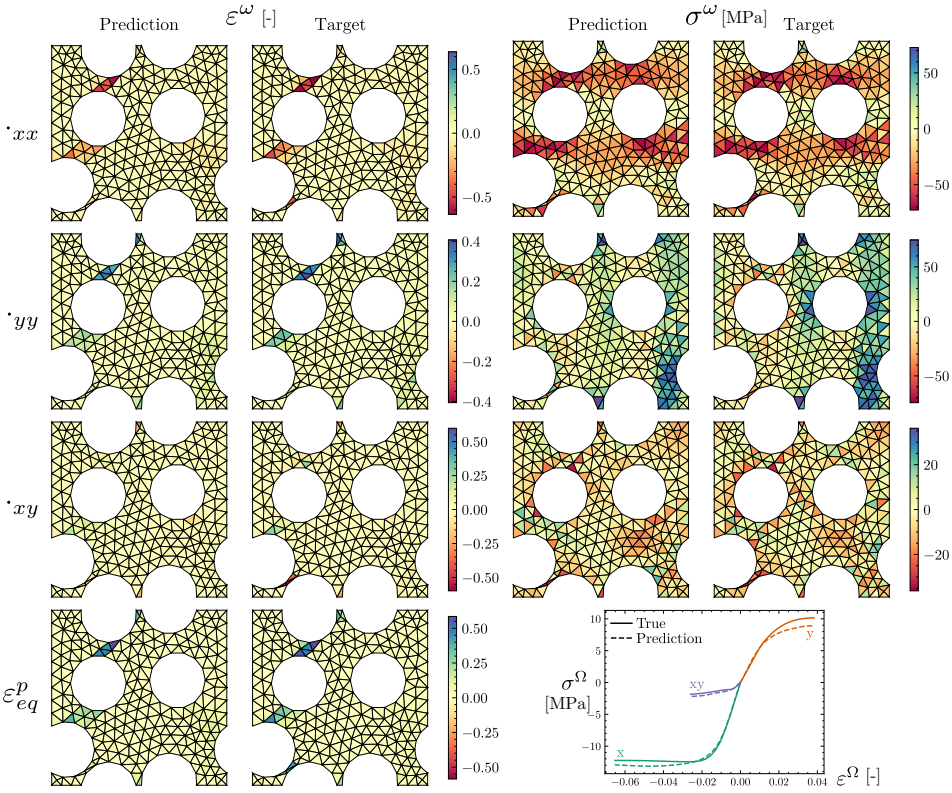


Figure 2.7.: Example of all full-field predictions made after 25 time steps and the complete homogenized stress-strain curve. Due to the elasto-plastic material the strain and stress fields have different patterns - the strain fields are dominated by a few elements with large plastic deformation.

material models (albeit in a very different manner) have been shown to be able to predict unloading without having been trained on it (Liu *et al.*, 2019b; Maia *et al.*, 2023).

To explore the potential of the model, we examine a scenario involving unloading. We create a new test set, but instead of monotonically increasing the strain, we decrease it from steps 15 to 19, unloading the microstructure, before resuming loading from steps 20 to 30 (keeping the magnitude of the strain increment fixed). In Figure 2.8 we show the results for a sample and observe that the model fails to capture unloading, as during the unloading phase the stress predictions are far away from the true unloading branch. Despite the good predictions for monotonic paths, the model does not implicitly learn to predict unloading from monotonic strain paths. In the following section, we therefore train it on non-monotonic data.

2.5. Non-monotonic elasto-plasticity

In an FE analysis, the local strain evolution in individual elements can be highly nonlinear functions of the evolution of boundary conditions. In the previous section, we demonstrated that the model can make accurate predictions for monotonically increasing load paths with elasto-plastic behavior, but it is not yet able to predict the unloading response. Therefore, we now increase the complexity of the training data, to improve prediction results for more general load cases.

2.5.1. Data generation

To create arbitrary load paths, we generate random walks based on Gaussian Processes (GP), as done in several other works (Logarzo *et al.*, 2021; Maia *et al.*, 2023; Mozaffar *et al.*, 2019). We retain a fixed random strain direction and proportional loading, but the norm of the strain vector now follows the magnitude of a GP sample. Specifically, we initialize a squared exponential GP with a variance of $2 \cdot 10^{-3}$, a length scale of 8, and

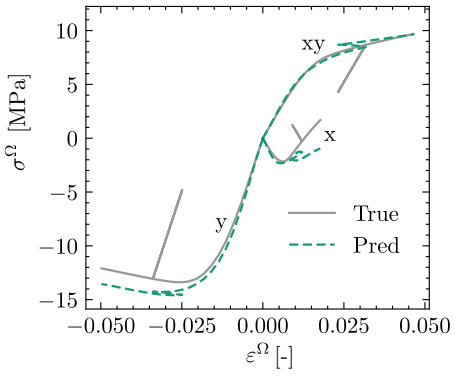


Figure 2.8.: The behavior of a GNN trained on monotonically increasing data, predicting a load case with unloading.

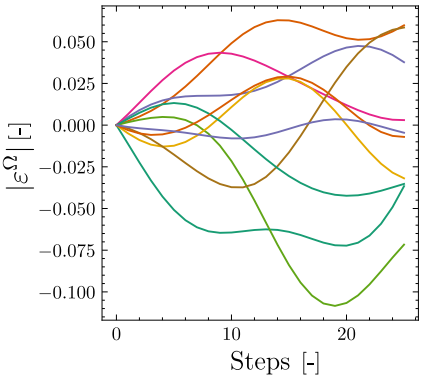


Figure 2.9.: The samples of the GP used for the strain magnitude.

fit through the point [0,0]. We draw random realizations from this GP for 25 time steps per realization; ten of these are shown in Figure 2.9. The generation of the microstructures and the material properties are the same as for the monotonic loading case. In Figure 2.10 the resulting ground-truth stress-strain curves are shown.

2.5.2. Model selection

We perform the same model selection process as in the section 2.4.2, but now for models trained with the GP-based data. The same initial values are used for the first round, namely $\xi = 0.6$, 5 MPLs, and a dropout rate of 0.1, and we vary each of these parameters one at a time. The losses are computed on a validation set of 2000 GP-based samples, therefore these losses cannot be compared directly to those of the monotonic case. We visualize the results in Figure 2.11. From this first round of model selection, we find that the optimal model is obtained with $\xi = 0.6$, 5 MPLs, and a dropout rate of 0.1. Again we find little difference between the errors in the number of MPLs. As these values for the hyperparameters are the same as the initial values, no second round is performed, and the model with these settings will be used for all following experiments.

So far we have been assuming that 4000 training samples are sufficient for the model

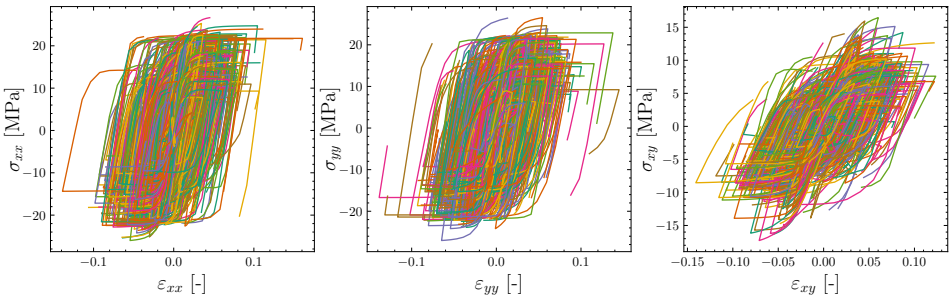
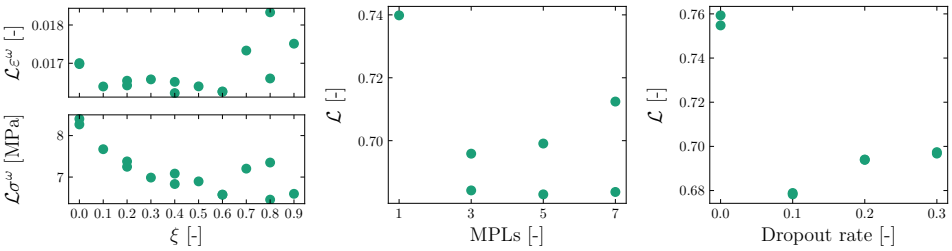


Figure 2.10.: Examples of stress-strain curves obtained from simulating 1 to 9-void microstructures following a non-monotonic, GP-based strain magnitude.



(a) Round 1: ξ value (b) Round 1: Number of MPLs (c) Round 1: Dropout rate

Figure 2.11.: The results for the hyperparameter study. The initial values chosen as default values for round 1 are ξ=0.6, MPLs=5, and dropout=0.1. Values for $\xi = 1$ are omitted from the plots as they lead to a very high $\mathcal{L}\varepsilon^\omega$. Since the optimal values are the same as the base values, we do not perform a second round.

and that providing more data would not significantly improve the model accuracy. Here we show that this is indeed the case using a learning curve by training models with a varying number of training samples. When the model accuracy does not increase significantly with more data, we take it as an indication the dataset size is sufficiently large. We show the test error of the various models in Figure 2.12. The trend has not completely converged and a lower error could possibly be obtained with more samples, but this also comes at the cost of requiring more expensive data. We therefore consider that using 4000 samples is sufficient. As this GP-based loading is more complex than monotonic loading, we argue that using the same amount of samples is more than enough for the monotonic case considered before.

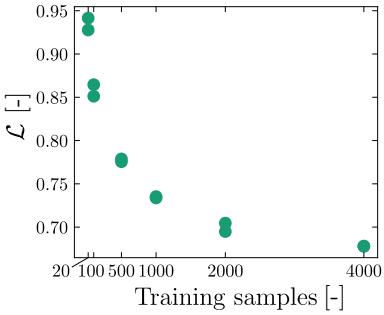


Figure 2.12.: The learning curve for models trained based on GP-based load paths, indicating that using 4000 training samples is sufficient. For each setting of training samples, we show the result of two training runs with random initialization.

2.5.3. Prediction results

Now, we return to the manually defined unloading case from Section 2.4.4 and compare the performance of the model trained on the GP-based data to that of the model trained only on the monotonic data. In Figure 2.13 we qualitatively show the resulting $\boldsymbol{\varepsilon}^{\Omega} - \boldsymbol{\sigma}^{\Omega}$ curves for both models. The initial loading phase predictions are very similar for both models, but the model trained on the GP-based data can predict the unloading phase much better. Still, predictions are far from perfect, and alternative surrogates only trained to predict σ^{Ω} can outperform this model.

In addition to these qualitative results, we also compare the test errors for the two models. The results in Figure 2.14 show an improved accuracy when training on the GP-based models. Notably, there is a larger improvement in the loss associated with σ^{Ω} than with σ^{ω}. This indicates that while predicting which elements are subject to how much unloading is still challenging, the overall unloading pattern of the field is predicted much better. It is clear from these experiments that training on non-monotonic data is necessary. With a model able to make full-field predictions for complex load scenarios, we now want to better understand the influence of the physics-based material model retained by the surrogate.

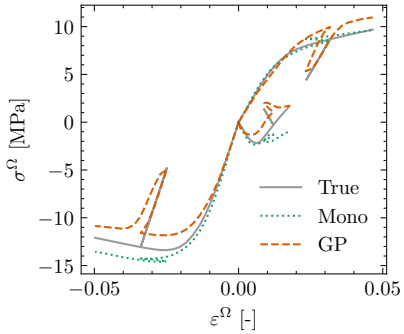


Figure 2.13.: Comparison between a model trained on GP data and a model trained on monotonic data in predicting the homogenized stress-strain curve of an unloading test sample.

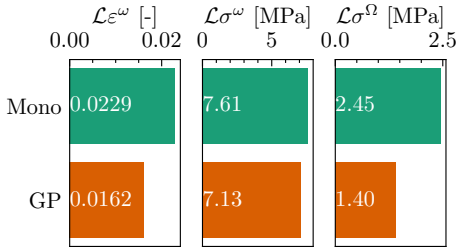


Figure 2.14.: Comparison of the test errors for unloading between the models trained with monotonic and GP-based data.

2.5.4. Material model

One of our main contributions is including a purely physics-based material model inside the surrogate to model nonlinear material behavior. In this section, we investigate the influence of the added material model on surrogate performance. The material model provides two additions, which we attempt to study separately as best as possible. First, we consider a model for which we remove the input feature of the equivalent plastic strain ε^p_{eq}, which is obtained when computing local stresses. Second, we study the influence of directly computing the stresses by removing the material model and instead predicting the stresses as GNN outputs while keeping the loss function the same. This data-driven option without the material model is closest to most GNN-based surrogate alternatives, and as it increases the complexity of the model, we expect it to require more training data to learn. We compare one additional model where we change the nature of the problem by training only for strains but still using the material model to compute stresses during inference. We achieve this by setting $\xi = 0$, and since we do not compute the material model during training, we also do not use the ε^p_{eq} feature. We thus consider three alternatives to the base model, which we summarize in Table 2.1.

Table 2.1.: Overview of alternative models.

Model	Material model during training	Material model during inference	ε^p_{eq} feature	ξ
A: Base model	✓	✓	✓	0.6
B: No ε^p_{eq}	✓	✓	✗	0.6
C: No material	✗	✗	✗	0.6
D: Strain-based	✗	✓	✗	0.0

In Figure 2.15 we compare a $\varepsilon^{\Omega} - \sigma^{\Omega}$ curve between the different models. Since model B performs similarly to the base model (shown in Figure 2.13), it is unclear whether the ε^{p}_{eq} feature is beneficial, at least for the current architecture. We observe that model C, without the material model, predicts the stresses well during loading but cannot predict the unloading phase. In this example, model D already shows a large error in the monotonic phase and does not predict the unloading slope well either.

We also quantitatively compare the models on two different test sets, one with GP-based paths, and the other is the unloading test set introduced in Section 2.4.4. Starting with the GP-based test set, we compare the learning curves of the different models in Figure 2.16. The error is significantly lower for model A than for model C, regardless of the number of training samples. This indicates that the material model directly results in better predictions rather than increasing the rate at which it learns. The improvement in overall accuracy predominantly originates from the stress predictions.

Model B produces very similar results to model A for the GP-based dataset, indicating that the ε^{p}_{eq} feature only represents a minor contribution to performance. We show the errors for the unloading test set in Figure 2.17, where we observe a slightly increased difference between models A and B. Still, we expected the model to rely more on this feature to accurately predict the stresses, reflecting what happens in the actual FE micromodel. Across both test sets, the losses of models C and D are higher than the base model, indicating that there is a significant benefit from including the material model. Model D outperforms model C, indicating that training with the stresses directly as a model output can be detrimental to the performance compared to only computing them with the actual material model during inference. We observe that the strain errors are similar between all models, and the difference in the error is mainly due to the stress predictions.

The contribution of the material model does come at a significant computational cost. To start, the history dependency of the material model complicates the computational graph for backpropagation. But more significant is the computational cost of evaluating the material model itself. As the material model is evaluated for each element in the mesh in the same way as when using the FE method, it scales with the number of elements and is therefore independent of the GNN hyperparameters. This can become a significant computational bottleneck for expensive material models, as this operation

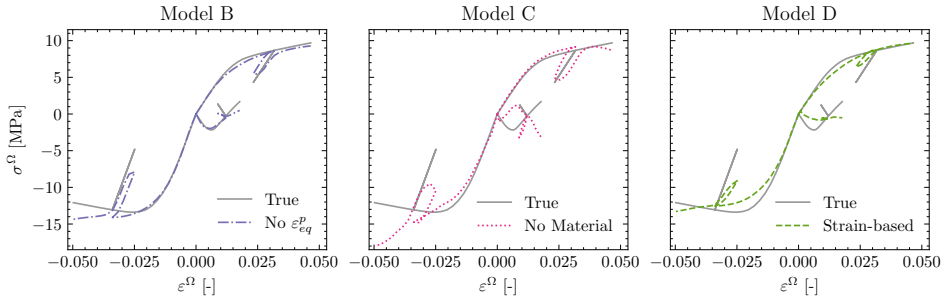


Figure 2.15.: A comparison between homogenized stress-strain curve predictions of the three alternative models (cf. Figure 2.13 for model A).

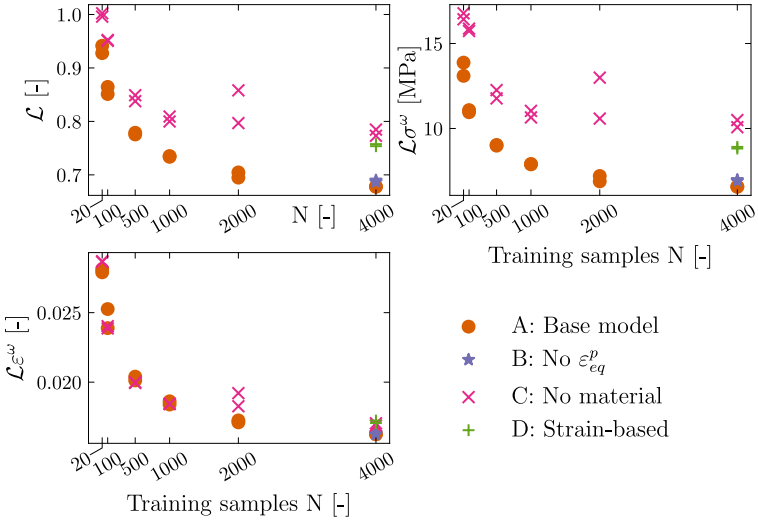


Figure 2.16.: The learning curves on a GP-based test set for the various alternative models. The individual components show the unnormalized losses.

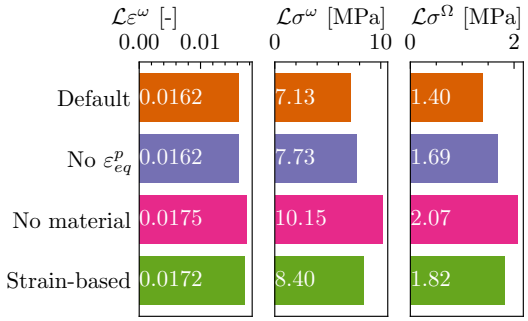


Figure 2.17.: A comparison of the test errors for unloading between the models with various alternatives to the material model.

needs to be performed many times during training.

2.5.5. Predicting for more time steps

In autoregressive models, errors can accumulate over time as their inputs depend on their previous outputs. This can potentially lead to out-of-distribution inputs, which can cause exponentially increasing errors to occur. We have only focused on the behavior of the model during the first 25 time steps, which was also the length of the training paths, and now investigate how the model behaves when extrapolating beyond this point. We generate a new test set of 2000 load curves with 50 time steps. For the model trained on samples with 25 time steps, we compute a mean and maximum error for each time step on this test set and average over all samples. We compute the norm of the difference between the ground truth and predicted fields as errors that can be interpreted well:

$$\text{Mean error} = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{E_n}\sum_{e=1}^{E_n}||\boldsymbol{\varepsilon}^{\omega}_{n,e} - \hat{\boldsymbol{\varepsilon}}^{\omega}_{n,e}||, \tag{2.16}$$

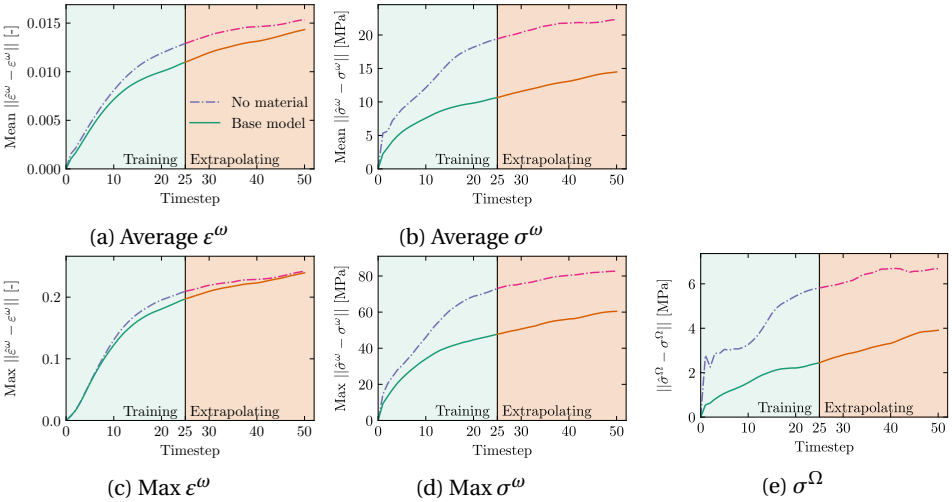
$$\text{Max error} = \frac{1}{N}\sum_{n=1}^{N}\max(||\boldsymbol{\varepsilon}^{\omega}_{n,e} - \hat{\boldsymbol{\varepsilon}}^{\omega}_{n,e}|| \;\forall\; e \in E_n). \tag{2.17}$$

We compute similar errors for σ^{ω} and σ^{Ω} and compare the performance of the model A to model C (without a material model). The results are shown in Figure 2.18, from which we can make two main observations. Firstly, after the errors increase quickly in the first few steps, the errors do not increase excessively as we go beyond the number of training steps but instead follow a near-linear trend. Secondly, comparing the model to the one without a material model, we again observe that while the strain errors are similar, a large discrepancy exists for the stress errors. Especially for the homogenized stress, the difference is already significant after the first few steps, where it fails to correctly predict the fast increase in stress during the elastic phase without the material model. This difference then stays relatively constant as we extrapolate beyond the number of training steps. The linear increase in error indicates that the model is stable when extrapolating to more time steps. We show several representative examples in Figure 2.19, where we can qualitatively observe that the models indeed still follow the correct trend.

2.5.6. Microstructure scaling

By training the model on various microstructures with a different number of voids, the model is encouraged to learn to predict general microstructures. Here we investigate how well the model can extrapolate to larger microstructures. We demonstrate the model predictions for a representative 49-void[1] sample by plotting the homogenized stress-strain curve in Figure 2.20 and the full-field stresses in Figure 2.21. While the microstructure size is never explicitly given to the model, the homogenized curves match very well. We observe that the model can predict the general trend of the full-field

[1] As a reminder, training is performed with microstructures with no more than nine voids.



(a) Average ε^{ω} (b) Average σ^{ω}

(c) Max ε^{ω} (d) Max σ^{ω} (e) σ^{Ω}

Figure 2.18.: Average errors based on 2000 test samples, where we extrapolate beyond the number of time steps seen during training. For each sample, the errors are computed for each time step. In the top row we plot the average error over all samples, in the bottom row we compute the maximum error for each sample and time step, before averaging over these maxima. We plot the quantities for base model A and model C trained without the embedded material.

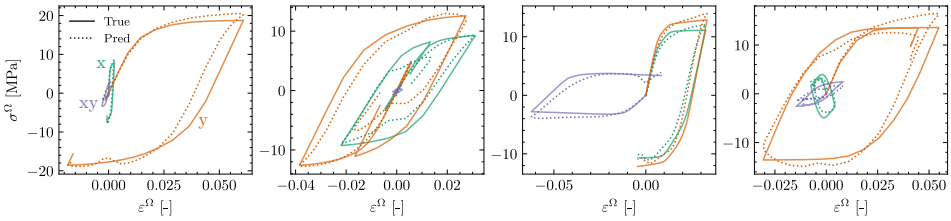


Figure 2.19.: Examples of homogenized stress-strain curves for 50 time steps.

stresses, although the predictions are not as accurate as for the smaller microstructures. We do not plot the strains as they are dominated by a few elements with very high strains and therefore difficult to compare visually.

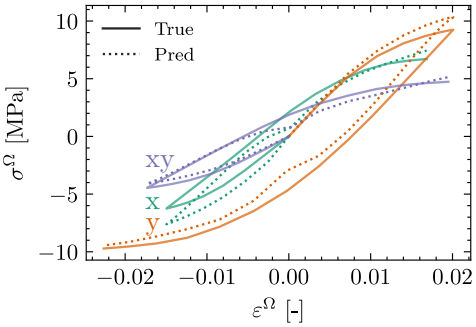


Figure 2.20.: Homogenized stress-strain curve for a 49-void sample.

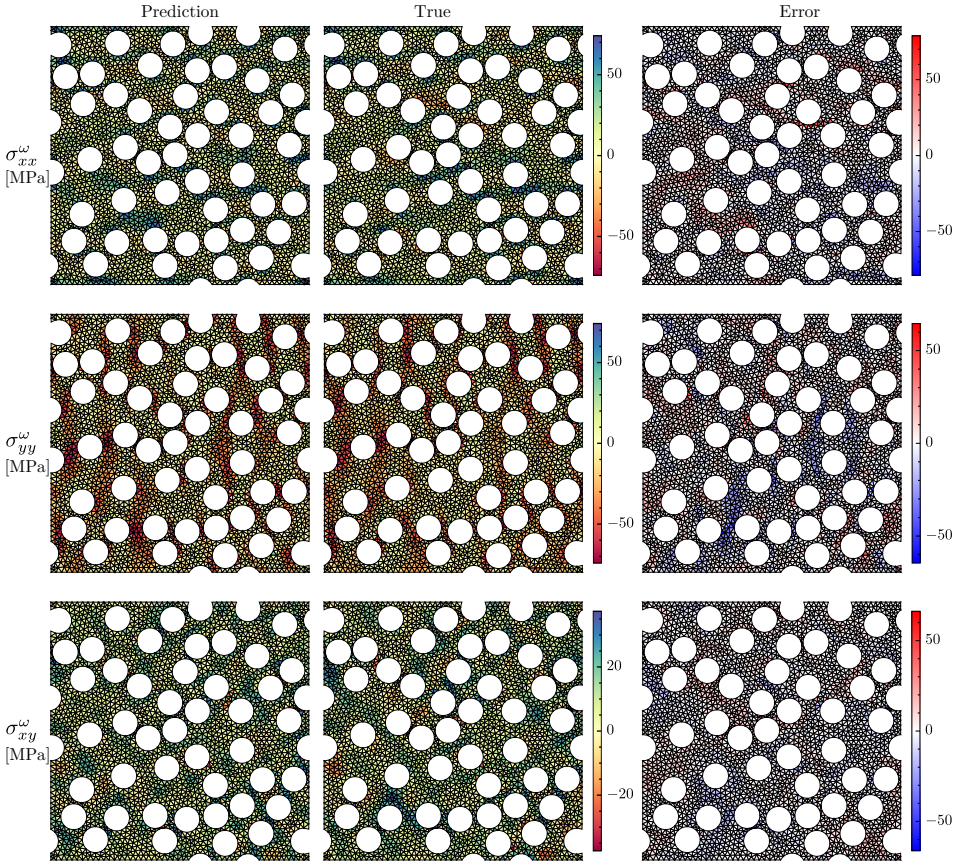


Figure 2.21.: Full-field stresses after 25 time steps for a 49-void sample.

One quantity that depends on the microstructure size is the initial stiffness: for small microstructures, the stiffness is not representative of the real behavior. When using the FE method, the homogenized stiffness is explicitly computed using the probing method. For our model, we can compute the stiffness as

$$\frac{\partial \sigma^{\Omega}}{\partial \varepsilon^{\Omega}} = \frac{\partial \sigma^{\Omega}}{\partial \sigma^{\omega}} \frac{\partial \sigma^{\omega}}{\partial \varepsilon^{\omega}} \frac{\partial \varepsilon^{\omega}}{\partial \varepsilon^{\Omega}}, \tag{2.18}$$

which we perform using automatic differentiation. In theory, this can be achieved by applying automatic differentiation after predicting with input $\varepsilon^{\Omega} = 0$. Since our GNN is a highly nonlinear model, using only $\varepsilon^{\Omega} = 0$ could potentially give a distorted result, and we instead average over several very small random ε^{Ω} values.[2]

We create test datasets with varying microstructure sizes to test the ability to extrapolate to larger unseen geometries. The size is determined by the number of voids (n_v) in the microstructure, as the volume fraction and void size are kept constant. For each size category, we generate 500 samples and compute the stiffness with an FE analysis and our model. We present the resulting stiffnesses in Figure 2.22. Our models, trained on data with an equal distribution of [1,2,3,4,5,6,7,8,9]-void microstructures, closely follow the FE result, for both the average stiffness but also the variance. Since the stiffness is already almost converged after 9 voids, which the model has seen during training, we train another model where we only include [1,2,3]-void microstructures during training. This model still follows the trend closely, although the mean stiffness stays relatively constant beyond four voids, instead of showing a slight downward trend towards nine voids. Nevertheless, both models give stiffness predictions that closely follow the FE results. Moreover, the decrease in variance for increasing microstructure size that is observed in the FEM predictions is reproduced also for unseen microstructure sizes. With this, we show that there is a positive inductive bias in the microstructure geometry that our model benefits from.

2.5.7. Volume fraction generalization

To further analyze the model capability, we study its capacity to predict microstructures with different void volume fractions (V_f) than the one seen during training. Keeping the size of the microstructure constant at that equivalent to a 9-void microstructure with $V_f = 40\%$, we vary V_f by changing the number of voids (n_v). This is different from our training set, where we would change the microstructure size when changing the number of voids to keep V_f constant. The model thus needs to generalize to unseen volume fractions, having only seen examples of one value. We generate 500 samples for each volume fraction and compute the different loss terms.

The results are shown in Figure 2.23. A U-shaped curve around the training regime is observed for the strains, with V_f around 40% still performing well. However, the performance degrades the further the V_f of the samples is from that observed in training. The difference in performance between stresses and strains becomes clear when looking at the full-field plastic strain predictions in Figure 2.24. For small volume fractions, the model predicts strains close to zero over the full domain, similar to what is seen

[2] In practice, we find the difference from doing this to be negligible.

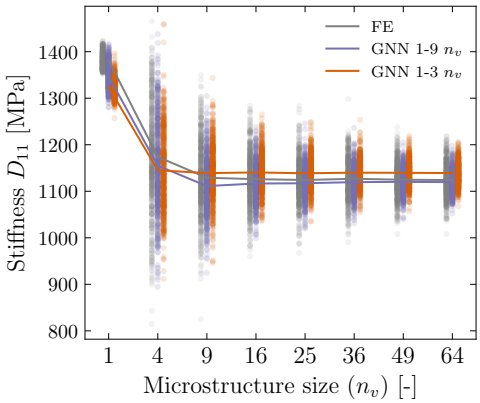


Figure 2.22.: A comparison of the stiffnesses obtained from the surrogate and an FE analysis for various microstructure sizes. The scatter shows 500 samples for each n_v, with the line giving the mean for each type. The surrogate models are trained on a limited number of voids, beyond which they are extrapolating.

around isolated voids in the training set. For high volume fractions, too many bands with high strain are predicted. The nonlinearity of the material model explains how underestimating strains leads to larger stress errors than overestimating strains.

Training on a dataset with varying volume fractions could improve the performance for unseen volume fractions. As this is one specific case of many possible generalizations in geometry (e.g. voids with non-circular shapes, phases with different material properties), we leave this for future work. In general, geometric features and training data should be carefully curated depending on the application.

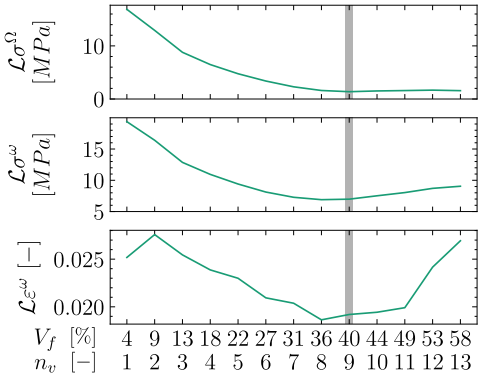


Figure 2.23.: Volume fraction extrapolation results for the unnormalized homogenized stress, full-field stress, and full-field strain losses. The gray band indicates V_f = 40%, the value used for the training samples. Each result is an average based on 500 test samples.

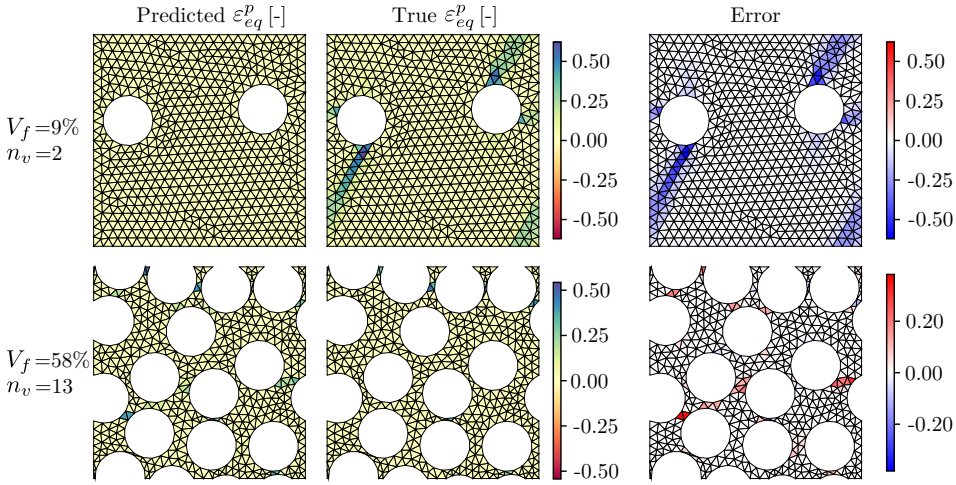


Figure 2.24.: Full-field ε^p_{eq} predictions for a small and large volume fraction, showing the model's bias to underpredict strains for small volume fractions and overpredict for large volume fractions.

2.5.8. Computational cost

Making a fair time comparison between using the GNN surrogate and a full FE^2 simulation is challenging for several reasons. The GNN surrogate is implemented in Python using PyTorch, created for GPU execution, while our in-house FE solver is written in C++ and executed on a CPU. We therefore focus on the relative scaling of both methods with the size of the microscale mesh, rather than on the absolute differences. We compute all steps sequentially for both the FE and GNN approaches, without using parallelization or batching techniques.

We perform FE multiscale simulations by applying tension to a macroscopic dogbone structure with holes. We vary the microstructure size between runs to study the scaling of the computation time. For each simulation, the strain paths and computational cost of solving the microscale model for all time steps are recorded, considering only the final Newton-Raphson steps that lead to macroscopic convergence. We subject the surrogate model (base model A) to the same strain paths and record the resulting execution time.[3] The resulting computation times corresponding to the microstructure size are shown in Figure 2.25.

The results match the expectation that the GNN surrogate scales linearly with respect to the microstructure size, whereas FEM scales exponentially. We also observe that computing the tangent stiffness using automatic differentiation is costly and scales worse compared to when only computing $\boldsymbol{\sigma}^{\Omega}$. Therefore, other methods to compute the tangent stiffness, such as finite differences, could be considered instead. Some hyperparameters of the GNN model, such as the number of MPLs, will also influence the

[3]This method allows us to compare the computational cost of the methods, but it is not equivalent to running full multiscale simulations. Similar to other surrogate models, embedding the GNN in an FE^2 simulation might lead to unphysical behavior and cause instabilities that have not been studied in this work.

computation time. However, these are not expected to change how the GNN scales favorably with the microstructure size compared to full FE^2 simulations.

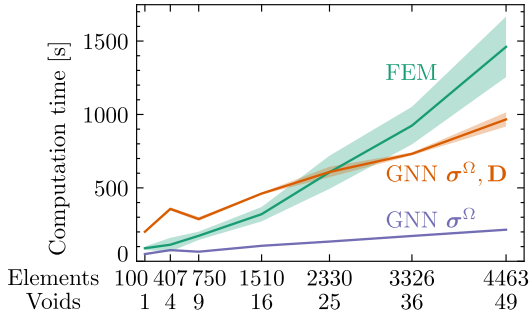


Figure 2.25.: Scaling of the computational cost between the GNN-based surrogate and FEM for an increasing microstructure size. We plot the mean and 95% confidence interval over 5 runs.

2.6. Conclusion

State-of-the-art surrogate models for multiscale simulations directly replace the complete microscale simulation with a machine-learning model. By doing so, the microscale physics and geometry are lost, and a large number of simulations from RVEs are required for training. We presented a GNN-based surrogate that replaces only the solving of the microscopic boundary-value problem while keeping the microscopic material models and computational homogenization intact. By obtaining all full-field quantities of interest the surrogate could be used interchangeably with an FE solver for any time step in a multiscale model.

We demonstrated the ability of our model to accurately predict microscopic strain and stress fields for monotonically increasing strain paths with nonlinearity caused by considering elasto-plastic materials. We found that embedding the material model in the GNN is insufficient to capture complex load scenarios such as unloading while only being trained on monotonic loading. Instead, we trained our model on a dataset with non-monotonic load paths, which improves its ability to predict unloading scenarios. Embedding the material model inside the surrogate architecture leads to a significant performance increase. While the stress directly depends on the strain, we found that including the stress in the loss function (in addition to the strain) not only leads to a significant improvement in the stress predictions but also generally improves the strain predictions. The model is autoregressive and therefore potentially sensitive to instability, but we show that the error does not increase excessively when extrapolating beyond the number of time steps seen during training. By training on arbitrary microstructures and predicting the stiffness of larger microstructures than when training the model, we show the potential for GNNs to extrapolate to larger microstructures. This is possible by the inductive bias of the GNN, which learns to use the geometry of the microstructure to make predictions. While trained on data with a fixed volume fraction, the model can predict the response of samples with other volume fractions close to the training

value well, although recovering a wider range of volume fractions would require augmenting the training dataset. The proposed surrogate has potential to be faster than an FE analysis by scaling better with the number of elements in the microstructure. We also expect these gains to be higher once our GNN code is further optimized and better integrated within existing FE code.

By keeping the multiscale nature of the problem while being faster than full FE^2 simulations, this model shows promise in accelerating simulations of multiscale mechanical models. This is the first surrogate to integrate an elasto-plastic material into a GNN while retaining all microstructure quantities for a multiscale simulation. With countless developments on GNNs, we expect that they will play an increasingly important role in the field of multiscale simulations, as they are naturally suited to model mesh-based systems. We further expect that purely data-driven models will not be optimal, and that hybrid data-driven and physics-based models such as the one we show here will provide more powerful surrogates for multiscale simulations.

If a man will begin with certainties, he shall end in doubts; but if he will be content to begin with doubts, he shall end in certainties.

Francis Bacon, *The Advancement of Learning*

3

RELIABLE SIMULATIONS WITH INADEQUATE SURROGATES

There is a high interest in accelerating multiscale models using data-driven surrogate modeling techniques. Creating a large training dataset encompassing all relevant load scenarios is essential for a good surrogate, yet the computational cost of producing this data quickly becomes a limiting factor. Commonly, a pre-trained surrogate is used throughout the computational domain. We introduce an alternative adaptive mixture approach that uses a fast probabilistic surrogate model as constitutive model when possible, but resorts back to the true high-fidelity model when necessary. The surrogate is thus not required to be accurate for every possible load condition, enabling a significant reduction in the data collection time. We achieve this by creating phases in the computational domain corresponding to the different models. These phases evolve using a phase-field model driven by the surrogate uncertainty. When the surrogate uncertainty becomes large, the phase-field model causes a local transition from the surrogate to the high-fidelity model, maintaining a highly accurate simulation. We discuss requirements for accuracy and numerical stability and compare the phase-field model to a local approach that does not enforce spatial smoothness in phase mixing. Using a Gaussian Process surrogate for an elasto-plastic material, we demonstrate the potential of this mixture of models to accelerate multiscale simulations.

This chapter is published as: Storm, J., Sun, W., Rocha, I. B. C. M., & van der Meer, F. P. (2025). Mixing Data-Driven and Physics-Based Constitutive Models Using Uncertainty-Driven Phase Fields. *International Journal for Numerical Methods in Engineering*, *126*(20), e70162.

3.1. Introduction

Advanced manufacturing techniques allow for the creation of materials with properties tailored to their use. Multiscale modeling can accurately capture the behavior of these materials, yet its use is limited by the high computational cost of simulations. The need for more computationally efficient models, combined with the progress in machine learning, has resulted in numerous data-driven models (Fuhg *et al.*, 2024; Ghavamian and Simone, 2019; J. Ghaboussi, 1991; Lu *et al.*, 2021; Raissi *et al.*, 2019; Vlassis and Sun, 2023). Given sufficient data, these models often show an impressive ability to capture the behavior of complex history-dependent load paths. They can therefore be used as surrogates for the representative volume element (micromodel) in a multiscale analysis. However, obtaining the necessary training data from microscale simulations is very costly. Because purely data-driven methods often extrapolate poorly, the data required to ensure fidelity tends to be large. In addition, it is often challenging to predict which loading scenarios will be relevant. The surrogate, therefore, needs to be trained across a broad range of inputs to generalize for unseen data, many of which may never be relevant during the target simulation.

We can broadly categorize two (non-exclusive) approaches to dealing with this problem. The first is to embed a prior in the surrogate design, allowing the model to extrapolate constitutive responses from small datasets. Examples include leveraging geometry or embedding physics into the model as an inductive bias (Dornheim *et al.*, 2024; Klein *et al.*, 2024; Liu *et al.*, 2021; Maia *et al.*, 2023). These models often have intermediate quantities or additional outputs with a physical meaning that can be leveraged during the training and inference process.

The second approach is to use active learning rather than rely solely on a pre-trained surrogate. In active learning, the surrogate is continuously retrained whenever it makes a poor prediction, avoiding the need for an all-encompassing dataset. Authors in Rocha *et al.*, 2021 propose using the uncertainty of a Gaussian Process (GP) surrogate to inform on-the-fly adaptivity. There, several fully-solved anchor models are used throughout the simulation to update the GP, triggered by the GP uncertainty. A similar approach, FE^{ANN}, stores all deformation gradients and, by comparing invariants of each new deformation to this dataset, it detects unseen deformations (Kalina *et al.*, 2023). In that case, the method runs new simulations, enriches the dataset, and retrains the network. Adaptively updating the surrogate during the simulation avoids the necessity for all scenarios to be covered in an extensive pre-training dataset. However, active learning does require the surrogate to be suitable for retraining. Furthermore, updating the surrogate can cause instabilities or slower convergence of the macroscopic solver.

Instead of using the high-fidelity model just to update the surrogate, a mixture of several models can be applied. A common strategy is clustering, where elements are grouped by response after an initial solve of the system (Li *et al.*, 2025). Alternatively, a mixture of a pre-trained neural network (NN) and a reduced order model (ROM) can be used throughout the domain (Fritzen *et al.*, 2019). After solving an FE^2 simulation with the NN surrogate at every point, a second run is performed where all points that reached values outside the NN training region during the first run are replaced with the ROM. To minimize unnecessary evaluations, the second run can also start with

the NN at all points, and switch to the ROM only when strain values outside the NN training region occur. One of the challenges of these methods is avoiding instability of the finite element method (FEM) caused by stress jumps at boundaries between two models. One option to avoid these jumps between models is to introduce a transition zone in front of a crack based on the thick level set method (Rocha *et al.*, 2020). From a machine learning perspective, these hybrid approaches are analogous to a mixture-of-experts model (Jacobs *et al.*, 1991), where we use domain-specific information to choose the expert. Alternatively, this also bears a resemblance to data assimilation methods, such as using a weighted least-squares finite element method to assimilate experimental data with numerical models (Rajaraman *et al.*, 2017).

In this work, we allow for adaptively using multiple constitutive models when and where necessary. Specifically, a probabilistic surrogate and the original model it was trained to replace are aggregated to make predictions. The main idea is to use the uncertainty in surrogate predictions to guide whether the surrogate or the original model should be used. For our motivation of accelerating multiscale simulations, the fast surrogate should be used when it has a low uncertainty, and the original model should be used otherwise. We use the idea of a weighted average transition zone and implement it using a phase field. The phase field promotes numerical stability, as it naturally introduces a spatial transition zone that is used to switch between the models. Phase-field models have been used to simulate the evolution of, among others, solidification (Boettinger *et al.*, 2002), grain growth (Chen, 2002), and fracture (Wu *et al.*, 2020a) for several decades now. In a phase field, a scalar variable ϕ evolves to represent, e.g., liquidsolid transitions or crack formation. Here, we use the phases for the different constitutive models. The uncertainty coming from the probabilistic surrogate drives the phase field evolution. At the diffuse interface between phases, we take a weighted average between model predictions to smoothly transition between the models.

Using the phase field brings the cost of solving an additional partial differential equation (PDE) every time step. In FE^2 simulations, the costs associated with solving microstructures are generally so high that the cost of solving the phase field is negligible. When a surrogate is trained on a well-curated dataset that allows it to predict the full behavior of the simulation with little uncertainty, the phase field remains zero and only the surrogate is used. In this case, this approach only requires the additional computation of the surrogate uncertainty. The mixture of constitutive models preserves accuracy when this condition is not met.

The width of the transition zone can play an important role in the stability of the mechanical and phase-field problems. A narrow transition zone requires fewer expensive model evaluations, making it computationally desirable. However, this might lead to undesirable stress jumps or make the solver require more Newton-Raphson iterations. To study this, we compare the phase-field approach to a purely local approach, which does not require solving another PDE. By letting the uncertainty directly determine which constitutive model is used, we can essentially let the transition zone vanish and study the impact on the stability of the mechanical problem.

The outline of this paper is as follows. In Section 3.2, we discuss using FEM for solving the behavior of structures undergoing loading. We provide a detailed overview of our phase-field-based approach for mixing models in Section 3.3. There, we also discuss

using a GP as a surrogate constitutive model. We present the results of the different approaches in Section 3.4. Finally, we conclude with an overview of our findings in Section 3.5.

3.2. Finite Element Analysis

FEM is used to model the behavior of a structure or material under load. Specifically, the aim is to find the displacement field $\mathbf{u}$ of a structure subject to boundary conditions. We obtain equilibrium in the domain by finding the displacement field $\mathbf{u}$ satisfying

$$\nabla \cdot \boldsymbol{\sigma} = 0. \tag{3.1}$$

Here, $\boldsymbol{\sigma}$ is the stress, and $\nabla\cdot$ indicates the divergence operator. $\mathbf{u}$ is related to the strains $\boldsymbol{\varepsilon}$ as

$$\boldsymbol{\varepsilon} = \frac{1}{2}\left(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T\right), \tag{3.2}$$

and $\boldsymbol{\varepsilon}$ can be related to $\boldsymbol{\sigma}$ through a constitutive relation:

$$\boldsymbol{\sigma} = \mathcal{C}(\boldsymbol{\varepsilon}, \boldsymbol{\alpha}). \tag{3.3}$$

The material-dependent constitutive operator $\mathcal{C}$ is often nonlinear and can depend on the loading history. The internal variables $\boldsymbol{\alpha}$ can account for this loading history. Due to the nonlinear nature of the constitutive model, Equation (3.1) cannot be solved directly. Instead, an iterative Newton-Raphson scheme is employed. To obtain quadratic convergence, we require not just the outcome of the constitutive relation ($\boldsymbol{\sigma}$), but also its derivative ($\boldsymbol{D}$).

To find equilibrium, the FEM discretizes the continuous domain of the structure into a number of elements. The boundary conditions, in the form of forces or prescribed displacements, are applied to the nodes of these elements. Within each element, the displacement field $\mathbf{u}$ is approximated using simple *basis functions*, often polynomials. The Gauss quadrature rule is used to evaluate the integrals arising from the discretization, with strategically placed integration points (IPs) within each element. The constitutive relation from Equation (3.3) is evaluated for each integration point.

We consider quasi-static loading, where the load is applied in consecutive time steps without causing dynamic effects. This allows for obtaining the full load path of the structure of interest. For a single simulation, we thus have to compute the constitutive model for all quadrature points in the elements, for each Newton-Raphson iteration of all loading steps.

For certain materials, such as composites, the constitutive relation can depend on the geometry at the microscale. Finding accurate constitutive relations for these materials is challenging, even if the constitutive relations of each constituent are known. In multiscale simulations using FE^2, a microscopic FEM problem is solved instead, where $\boldsymbol{\varepsilon}$ is imposed as boundary conditions on a microstructure. We solve this system as in FEM since the constituents' properties are well described, then pass the homogenized stress $\boldsymbol{\sigma}$ back to the macroscale. However, solving a boundary value problem at every macroscopic quadrature point and time step brings a considerable computational cost.

This cost is so high that, even with parallelization, FE^2 is used only sparingly in practice, motivating acceleration strategies.

To make it computationally feasible to study the approach in detail, we limit ourselves to single-scale problems in this work. We use an elasto-plastic material with a von Mises yield criterion as our high-fidelity (HF) model, which serves as the ground truth. To quantify the acceleration of our approach, it thus makes little sense to compare the computational time—as evaluating this elasto-plasticity model is about as fast as the surrogate we use to replace it. Instead, we measure the reduction in the number of HF model evaluations. The simplification of using an analytical constitutive model, therefore, does not affect our upcoming conclusions about the approach.

For complex problems, the Newton-Raphson solver can fail to converge for a given load increment. Adaptive load stepping is then required to obtain a solution. In this work, we use an adaptive load step strategy that stays constant as long as the solver converges. If it fails to converge, the load step is reduced by a factor γ. This is repeated until the solver converges, or until a minimum load step value is reached. After reaching the minimum value, increased load steps are attempted, up to a maximum value, after which the simulation is terminated and the problem is considered unsolved. If it converges, the load step remains constant for one step before gradually increasing back to the initial size, using a factor $\frac{1}{\gamma}$.

3.3. Mixture of Constitutive Models

We solve the mechanical problem by combining constitutive models of distinct accuracies and computation speeds in the macroscopic domain. The analytical elasto-plastic material model serves as the high-fidelity model ($\mathcal{C}_{\text{HF}}$), and we use a GP surrogate model ($\mathcal{C}_{\text{GP}}$). Using a GP is a pragmatic choice, as it naturally provides an uncertainty. A limitation of a GP is its inability to handle path-dependency. Other types of constitutive models and surrogates could be used, as long as the surrogate provides an uncertainty in its prediction. The general schematic of the approach in Figure 3.1 shows what information the mixture model obtains from other models. Ideally, $\mathcal{C}_{\text{GP}}$ is used throughout most of the domain, and $\mathcal{C}_{\text{HF}}$ only for a small subset of points. Our hybrid constitutive model depends on a phase-field variable ϕ to determine which constitutive model is used at each quadrature point:

$$\mathcal{C}_{mix} = \begin{cases} \mathcal{C}_{\text{GP}} & \text{if} \quad \phi < \tau, \\ (1-\phi)\mathcal{C}_{\text{GP}} + \phi\mathcal{C}_{\text{HF}} & \text{if } \tau < \phi < 1-\tau, \\ \mathcal{C}_{\text{HF}} & \text{if} \quad \phi > 1-\tau. \end{cases} \tag{3.4}$$

At the interface between the two phases, the hybrid constitutive model is thus a weighted average of $\mathcal{C}_{\text{HF}}$ and $\mathcal{C}_{\text{GP}}$, as visualized in Figure 3.2. In practice, we take weighted averages over both $\boldsymbol{\sigma}$ and $\boldsymbol{D}$. We use a small cutoff value, τ, to avoid simulating $\mathcal{C}_{\text{HF}}$ when the phase field is close to but not exactly zero. The upper boundary of $1-\tau$ is of lesser importance since the cost of solving $\mathcal{C}_{\text{GP}}$ is negligible, and its uncertainty is still required when updating the mixture. We use $\tau = 0.01$ in all numerical experiments. Algorithm 1 shows this procedure in more detail. The high-fidelity model

generally tracks internal variables $\boldsymbol{\alpha}$ to, for example, account for plasticity. These internal variables are only committed when moving to the next time step. When switching from the surrogate model to the full model (when ϕ crosses the threshold value τ), the internal variables of the full model are unknown yet necessary. In this scenario, the history path is recomputed using the full model to obtain the internal variables. We call this process the retracing of history. Although this increases the number of evaluations, we only need to retrace along the converged solutions. Thus, this still requires fewer evaluations than if the full model were used for this point from the start. We could, in principle, use larger time step sizes when reconstructing, but here we pragmatically opt for retracing every increment missed by the full model. This also means we have to store converged strain values at every fully-reduced IP for all time steps in the event a retracing is needed.

Algorithm 1 Mixture of constitutive models

Input: $\boldsymbol{\varepsilon}, \phi$
Output: $\boldsymbol{\sigma}, \boldsymbol{D}$
Initialize: Internal variables $\boldsymbol{\alpha}$
if $\phi < \tau$ **then**
  Use only the GP model: $\boldsymbol{\sigma}, \boldsymbol{D} = \mathcal{C}_{\mathrm{GP}}(\boldsymbol{\varepsilon})$
else
  Retrace $\boldsymbol{\alpha}$ to the current timestep if it is outdated
  if $\phi > 1 - \tau$ **then**
    Use only the high-fidelity model: $\boldsymbol{\sigma}, \boldsymbol{D} = \mathcal{C}_{\mathrm{HF}}(\boldsymbol{\varepsilon}, \boldsymbol{\alpha})$
  else
    $\boldsymbol{\sigma}_{\mathrm{GP}}, \boldsymbol{D}_{\mathrm{GP}} = \mathcal{C}_{\mathrm{GP}}(\boldsymbol{\varepsilon})$
    $\boldsymbol{\sigma}_{\mathrm{HF}}, \boldsymbol{D}_{\mathrm{HF}} = \mathcal{C}_{\mathrm{HF}}(\boldsymbol{\varepsilon}, \boldsymbol{\alpha})$
    $\boldsymbol{\sigma} = \phi \boldsymbol{\sigma}_{\mathrm{HF}} + (1 - \phi) \boldsymbol{\sigma}_{\mathrm{GP}}$
    $\boldsymbol{D} = \phi \boldsymbol{D}_{\mathrm{HF}} + (1 - \phi) \boldsymbol{D}_{\mathrm{GP}}$
  end if
end if

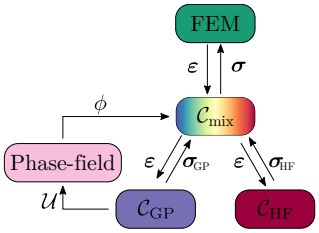


Figure 3.1.: Overview of how the mixture constitutive model interacts with the other models. The tangent stiffness $\boldsymbol{D}$ is omitted for clarity.

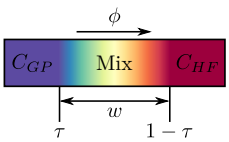


Figure 3.2.: As ϕ increases, the constitutive model $\mathcal{C}_{mix}$ gradually switches from $\mathcal{C}_{\mathrm{GP}}$ to $\mathcal{C}_{\mathrm{HF}}$.

3.3.1. Phase-Field Approach

The main purpose of the phase field is to control how the surrogate uncertainty influences the mixture. Specifically, it allows us to create a controllable interface width and determine the minimum uncertainty required to initiate the switch. While monolithic schemes exist (Gerasimov and De Lorenzis, 2016), staggered schemes are more common (Luo, 2023) and are adopted here. The variational form used in our numerical implementation is as follows:

$$F(\phi, v) = -\underbrace{\int_\Omega [\mathcal{U} v]\, d\Omega}_{\text{driving force}} + \underbrace{b\int_\Omega v\, d\Omega}_{\text{opposing force}} + \underbrace{\int_\Omega \left[\epsilon^2 \nabla\phi \cdot \nabla v\right] d\Omega}_{\text{interface energy}} + \underbrace{\int_\Omega \left[\omega\phi(1-\phi)(1-2\phi)\, v\right] d\Omega}_{\text{double-well}} \tag{3.5}$$

Here, v represents the test function. The first term is the driving force $\mathcal{U}$, causing ϕ to increase as the uncertainty increases. Because we use a staggered approach, the driving force does not necessarily have to be continuous with respect to the strains to maintain stability. The second term is a constant opposing force, acting as a threshold that the driving force needs to overcome. The third term represents the interface energy, penalizing sharp gradients in the phase-field variable. By influencing the interface width, this term may affect the stability of the problem. The fourth term is the derivative of the double-well potential, ensuring the phase-field variable favors values near 0 or 1. The phase-field formulation does not depend on its previous state, as we do not require our phase field to be smooth in time. This avoids issues related to path dependency, where a different loading step size would lead to different behavior.

3.3.2. Staggered Updating Scheme

We solve the phase-field and mechanical problem in a decoupled, staggered manner. In Figure 3.3 a schematic overview of the staggered updating scheme is shown. Although either field can be updated first (Luo, 2023), we choose to first update the phase field based on the last known surrogate model uncertainty. The phase field is then fixed while solving the mechanical problem.

Solving each problem once per time step is desirable, but has two drawbacks. First, the uncertainty from the previous converged step might not be a good indicator of the per-

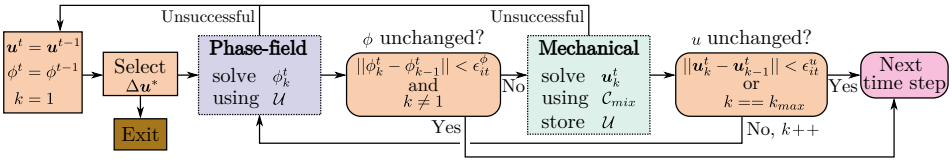


Figure 3.3.: The staggered updating approach used in this work. A prescribed displacement Δu^* is selected using an adaptive time-step function, before updating the phase field based on $\mathcal{U}$. Then the mechanical problem is solved. The phase-field and mechanical problem are updated iteratively until convergence, or k_{max} is reached.

formance at the current step. Second, this can introduce a load step size dependency, where taking different step sizes leads to different solutions. To overcome these issues, we iterate several times over this staggered scheme before moving on to the next step. We consider this approach to be converged when the norm of the change in $\boldsymbol{u}$ is below a threshold value ϵ_{it}^{u}. In certain scenarios, such as for the first few time steps when $\phi = 0$ for all points, the phase field will not change, even when $\boldsymbol{u}$ changes. We then know that $\boldsymbol{u}$ will not change in the next iteration and can consider the current solution converged based on ϕ, avoiding redundant iterations. By employing these iterations over the staggered approach, we are adding additional solves of the mechanical problem in which we generally need to perform costly $\mathcal{C}_{\text{HF}}$ evaluations. However, by using the result of each iteration as the initial prediction for the subsequent iterations, these should converge using fewer Newton-Raphson iterations than they would if starting from $\boldsymbol{u}_{t-1}$.

Instead of relying on the uncertainty coming from the mixed model (which entails expensive high-fidelity simulations), one might instead be inclined to obtain the uncertainty from a preliminary surrogate-only solution. One would first solve the mechanical problem using only the surrogate model, update the phase field based on the resulting uncertainty, and then solve the mechanical problem again with the mixture model. However, we find empirically that this approach is not viable: firstly, it heavily depends on reasonable surrogate performance and can fail to converge for inaccurate predictions; secondly, it reintroduces load step size dependency. Due to these reliability concerns, we do not present results using this approach.

3.3.3. Gaussian Processes as surrogate constitutive models

Bayesian surrogate models provide powerful tools for approximating complex functions. While several surrogate models provide uncertainty estimates (neural network ensembles, dropout networks (Gal and Ghahramani, 2016), bootstrapping approaches) and are therefore suitable for our approach, we focus on GP regression. This is a pragmatic choice to demonstrate our framework for mixing constitutive models. In more realistic applications, GPs might be insufficient as they are unable to handle path-dependency, and scale poorly with the number of training points. Other surrogates that provide a measure of uncertainty could be used instead. In Appendix A we provide a general description of GPs for regression that serves as background to this section. Here we describe how we use GPs specifically as constitutive models.

We create a separate GP for each stress component. Since our experiments are 2-dimensional, we thus have separate GPs for σ_x, σ_y, and σ_{xy}, each taking the full strain tensor as input. The GPs use a radial basis function kernel, and their hyperparameters are individually tuned. Rather than predicting σ directly, we predict a correction term to a linear elastic model:

$$\sigma_i = [\boldsymbol{D}_e \boldsymbol{\varepsilon}]_i + \text{GP}_i(\boldsymbol{\varepsilon}), \tag{3.6}$$

where $\boldsymbol{D}_e$ is the elasticity tensor. For predictions away from the data, the output thus reverts back to a linear elastic model, rather than to the zero prior. The scalar driving force for the phase field is computed as the maximum variance of the components:

$$\mathcal{U} = \max_i(\sqrt{var[\text{GP}_i(\boldsymbol{\varepsilon})]}). \tag{3.7}$$

Since this GP formulation only depends on $\boldsymbol{\varepsilon}$, it cannot capture unloading after plasticity has occurred. If unloading is to occur in the simulation, it is likely that the GP could make inaccurate predictions. For this reason, we focus on scenarios where the global load prescribed by the boundary conditions increases monotonically throughout the simulation.

As in related work, we assume that the surrogate variance correlates with the true error. If a surrogate is confidently wrong, the result can be inaccurate, as it would be if only the surrogate was used. Anomaly detection could be used to identify such scenarios, but this would come at a large computational cost.

3.4. Results

We start by showing the behavior of our mixture of models on a simple dogbone structure loaded with a prescribed displacement in tension. In this dogbone experiment, we focus on the influence of performing several staggered iterations. Then, moving to a more complex notched plate, we study the influence of the different phase-field parameters. We additionally compare the phase-field method to an alternative local approach. Finally, we investigate the potential of our approach to reduce the number of high-fidelity simulations on a plate with holes where a complex phase field is required.

3.4.1. Numerical Setup

As discussed earlier, for computational feasibility we do not perform FE^2 in this study, but instead use an analytical constitutive model as C_{HF}. For this reason, we do not compare the computational time of our approach to the full FE^2 model, but instead focus on the reduction in the number of high-fidelity model evaluations. We use an elasto-plastic material with von Mises plasticity in a plane-stress condition, a Young's modulus of 3130 [MPa], a Poisson's ratio of 0.37 [-], and yield criterion $\sigma_y = 64.80 - 33.60 \cdot e^{\frac{\varepsilon^p_{eq}}{-0.003407}}$, where ε^p_{eq} is the equivalent plastic strain (corresponding to the internal variable $\boldsymbol{\alpha}$ used earlier). These material properties and yield criterion are adopted from (Rocha, 2019). In all experiments, we use quadratic triangular elements (T6) for the mechanical problem and linear triangular elements (T3) for the phase field. For the dogbone structure, a constant displacement increment of 0.001 is sufficient to obtain convergence. In the remaining problems, the adaptive time step handling as described in Section 3.2 is used, with $\gamma = 0.5$.

We study the influence of the phase field with three different GP surrogates, *GP10*, *GP30*, and *GP100*, trained on different datasets with 10, 30, and 100 load curves, respectively. Each load curve consists of 20 monotonically increasing steps in a random strain direction, up to a strain norm of 10%. We choose 10% for pragmatic reasons, assuming we have no prior information about the strains that will occur during the simulation. Just because certain GPs have more data does not guarantee that their predictions during the simulation are always more accurate, since the loading directions of their data are random. Still, it can be reasonably expected that the GPs with more

data improve accuracy and reduce reliance on the full model.

The framework is implemented using the *FEniCSx* finite element library (Baratta *et al.*, 2023) with the *dolfinx_materials* (Bleyer, 2024) package to define arbitrary constitutive models. *PETSc* (Balay *et al.*, 2024) solvers are used to solve the mechanical and phase-field problem. The phase field is solved using a constrained solver with bounds $[0,1]$. All code is available on GitHub at `https://github.com/SLIMM-Lab/phase-field-mixture`.

3.4.2. Time Step Consistency

We start with a simple dogbone structure to study the influence of the staggered approach of Figure 3.3 on the time-step dependency. The dogbone structure is fixed on its left edge, and loaded by incrementally applying a prescribed displacement to the right edge, up to an elongation of 2%. Figure 3.4 shows the full-field responses for the stresses and equivalent plastic strain. The stresses mainly occur in the x-direction and concentrate at the center of the dogbone, where the structure is narrower than at its ends, with ensuing strain localization.

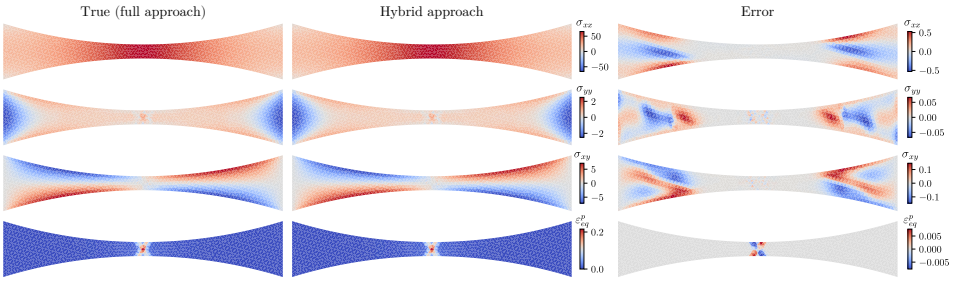


Figure 3.4.: Full-field stress and equivalent plastic strain ε^p_{eq} plots of the dogbone experiment at the maximum prescribed displacement. The full and hybrid model with $k_{max} = 3$ are shown, as well as the difference between them.

For the phase-field mixture we use the *GP10* surrogate with hyperparameters $\epsilon = 10^{-2}$, $\omega = 10^{-3}$, and $b = 1$. Figure 3.4 shows that the hybrid approach closely follows the full model. On the left side of Figure 3.5, we plot the phase-field variable for various time steps using $k_{max} = 1$ (i.e. a single solve of each PDE per load increment, see Figure 3.3). At the start, when ϕ is zero everywhere, only the GP surrogate is used. Starting from time step seven, the uncertainty in the high-strain region at the center of the dogbone drives the phase field locally up to one. The phase field can switch from $\phi = 0$ to $\phi = 1$ in a single load step when the uncertainty increases quickly. This is necessary to prevent large errors when the GP suddenly becomes uncertain.

In Figure 3.6, we present the load-displacement curves for this problem. Using only the GP surrogate results in a poor prediction that does not capture the nonlinear behavior of the material, showing that the surrogate is inadequate for predicting the structural response by itself. The inability to capture non-linearity results from the very limited dataset used for GP10. Still, by switching away from the surrogate at the right time, the hybrid approach closely follows the full model path.

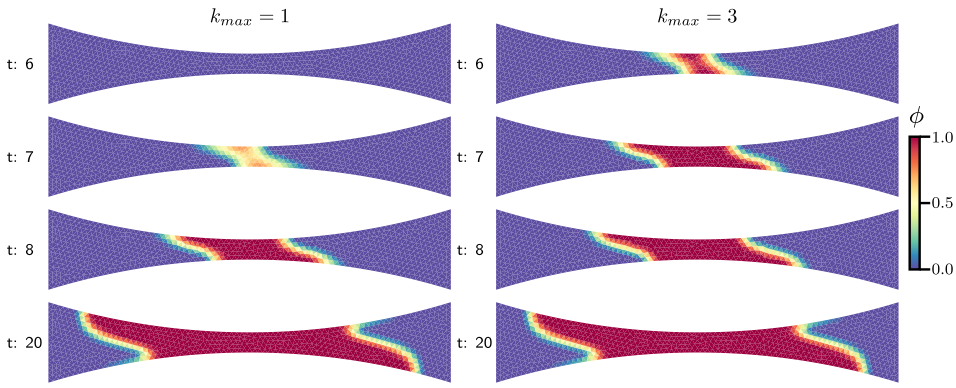


Figure 3.5.: Evolution of the phase field for the dogbone experiment. The phase field gradually grows until time step 17, after which it stays constant due to strain localization. The asymmetrical pattern occurs as a result of the asymmetrical shear stresses.

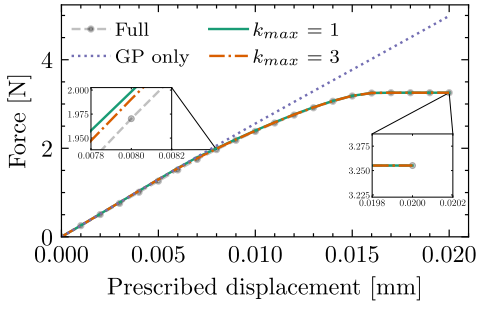


Figure 3.6.: The load-displacement curves for the dogbone experiment. While using only the surrogate model results in poor performance, the hybrid approach shows a large agreement with the baseline full model, with a slightly lower error for $k_{max} = 3$ compared to $k_{max} = 1$.

Figure 3.7 shows the mean absolute error of this hybrid model over the domain compared to using the high-fidelity model everywhere. The errors are computed for two fixed increment sizes Δu. When $k_{max} = 1$, there is a clear difference in errors when using different increments. This difference can be explained by the phase field using uncertainty from the previous time step to determine the mixture of constitutive models. When using a different increment size, the phase field thus updates on different information, creating a time-step dependency. In contrast, we observe that using $k_{max} = 3$ results in the same response for both loading increment sizes at the points where both are evaluated. Performing multiple iterations of the staggered approach per loading increment thus avoids the time-step dependency.

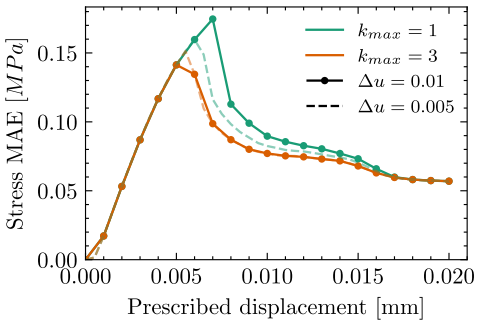


Figure 3.7.: The full-field stress error when using $k_{max} = 1$ or $k_{max} = 3$ for different load step increments.

With multiple iterations, the phase field is updated using information from the current load increment rather than from the previous iteration. In Figure 3.8 the evolution of ϕ throughout the simulation is plotted. It is clear from this plot that $k_{max} = 1$ lags roughly one step behind $k_{max} = 3$. This can also be observed by comparing the left and right sides of Figure 3.5.

The downside of performing several iterations of the staggered approach is that it requires more Newton-Raphson iterations of the mechanical problem, causing more high-fidelity constitutive model evaluations. Figure 3.9 shows the cumulative number of high-fidelity evaluations, where we observe that $k_{max} = 3$ uses more evaluations than $k_{max} = 1$. The reduction in high-fidelity evaluations compared to running the full model is relatively small in this example, as we intentionally use a GP trained on a very limited dataset to show the behavior of the phase field. Since it avoids the time-step-dependent behavior and improves the model's accuracy, we argue that performing several iterations is worthwhile for guaranteeing model robustness.

We further investigate the influence of the number of staggered iterations by running the model for more values of k_{max}. Figure 3.10 shows the results by plotting the error and number of staggered iterations used throughout the simulation. Using two iterations already shows a significant difference in the error compared to using only one. Even with $k_{max} = 4$, the model always converges within three iterations for this experiment. However, it is possible for the problem to oscillate between iterations, never meeting the tolerances, so k_{max} should be set. We set $k_{max} = 3$ for the remainder of this work.

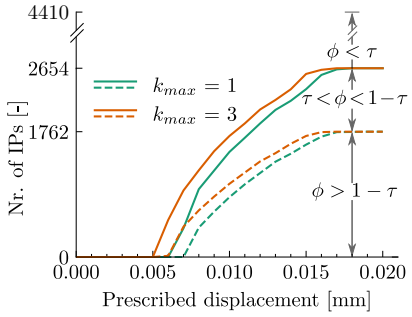


Figure 3.8.: Number of integration points (IPs) that use the GP surrogate ($\phi < \tau$), a mixture of both models ($\tau < \phi < 1-\tau$), or the high-fidelity model only ($\phi > 1-\tau$). The mesh contains 4410 IPs in total.

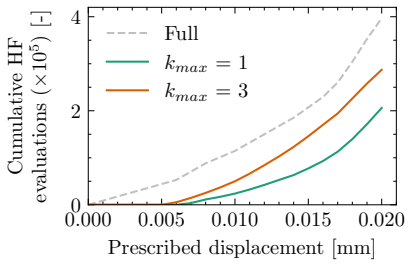


Figure 3.9.: Comparison of the cumulative number of HF evaluations. Retraced history computations are counted in the time step in which they are retraced. The significant difference between $k_{max} = 1$ and $k_{max} = 3$ comes mainly from the additional staggered iterations, requiring more Newton-Raphson steps.

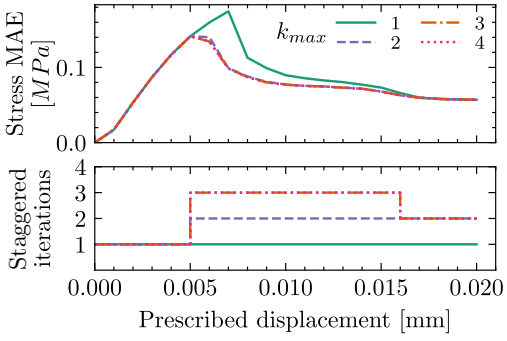


Figure 3.10.: The influence of k_{max} on the simulation. In this example, the model always converges within three iterations.

3.4.3. Phase-Field Parameters

In this section, we explore the influence of the phase-field parameters in more detail. To highlight their influence, we create an experimental setup with a more complex phase-field evolution pattern.

Experimental Setup

A plate with two notches diagonally opposite each other is loaded by incrementally displacing the right boundary by a prescribed amount. This setup is visualized in Figure 3.11, and we plot full-order simulation results in the left plots of Figure 3.12. It can be observed that stress concentrations occur at the tips of the notches, and a plastic strain band develops between these tips. The full-field errors of the hybrid approach with base settings show close agreement with the full-field model. Note how in this case the error with *GP10* is lower than with *GP100*, but this comes at the cost of running more high-fidelity simulations. We vary the element size over the domain to capture the strain localization accurately. Even though the global loading increases monotonically, we observe local unloading in some integration points close to the notch tips. As our GP surrogates cannot capture unloading, switching to the high-fidelity model at these points becomes essential. This is, however, not guaranteed by our method, particularly when using a high opposing force (through parameter b) that allows the GP to be used despite high uncertainty. Still, we empirically find that the limited local unloading has minimal impact on the results and does not affect our conclusions.

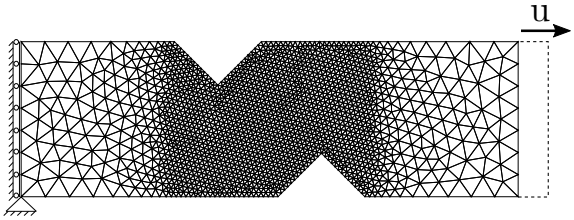


Figure 3.11.: Overview of the notched plate example.

Several quantities are used to compare the influence of the hyperparameters. To evaluate the accuracy of the model, we compare force-displacement curves. Since we now use an adaptive stepper, the points of the force-displacement curve do not coincide if a certain step fails to converge, so we cannot directly compute their differences. Instead, this error is computed by first linearly interpolating each F-u curve to the fixed displacement points that the solution follows if it always converges. To compute the error, we sum the absolute force differences between the approximate model and the true model. In addition, we plot the average number of integration points (IPs) where the high-fidelity model is used (i.e. where $\phi > \tau$). This indicates how much the high-fidelity phase has grown during the simulation. As an indication of the stability, we compare the required number of Newton-Raphson iterations of the mechanical problem, obtained by summing over all adaptive time steps (converged or unconverged), for all staggered approach loops. For the number of high-fidelity evaluations, we track

each time the high-fidelity model is evaluated, including the retracing of load paths to obtain the internal variables (as described in Section 3.3.2).

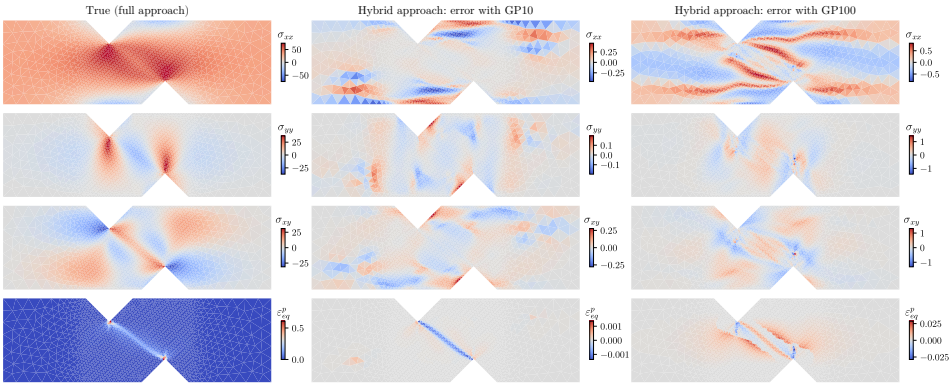


Figure 3.12.: Full-field stress and equivalent plastic strain ε^p_{eq} plots of the notched plate experiment at the maximum prescribed displacement. The full model results are plotted on the left. For the hybrid approaches, we plot the results for *GP10* and *GP100*. The differences are so small that we only plot the errors. Hyperparameters $b = 1$, $\epsilon = 10^{-2}$, and $\omega = 10^{-3}$ are used.

Phase-Field Opposing Force

We study the influence of the phase-field hyperparameter b on the mechanical problem. The parameter b determines the magnitude of the constant opposing force, as shown in Equation (3.5). The larger b is, the larger the uncertainty $\mathcal{U}$ needs to be to drive the phase field up from zero towards one. Since the uncertainty of different surrogate models can behave very differently, the choice of b should be calibrated based on the behavior of the variance of the surrogate. Furthermore, it is challenging to directly compare different surrogates for the same b value. For example, the GPs we use in this work all have different hyperparameters that influence their maximum variance and how quickly the uncertainty increases away from training points. While this maximum variance could in principle be used to scale $\mathcal{U}$, making b an adimensional threshold, other surrogate methods often do not have an upper bound to their uncertainty. Therefore, we let b simply be the unscaled uncertainty.

We vary b while fixing the other phase field parameters from Equation (3.5) to $\epsilon = 10^{-2}$ and $\omega = 10^{-3}$. In Figure 3.13 we show how the phase field evolves throughout the simulation for the different GPs for different values of b. We observe that the uncertainty of GPs with more data grows more slowly and that the larger the value of b, the fewer IPs use the high-fidelity model. We visualize the influence of b on the error, the phase-field evolution, the required number of Newton-Raphson iterations, and the number of high-fidelity evaluations in Figure 3.14. The gray line in the plots refers to the values obtained when running the full model, with the high-fidelity constitutive model used in all IPs. Missing data points, and the crosses in the bottom plot, indicate that the setting failed to converge (black crosses indicate a failure of the mechanical problem,

red crosses indicate a failure of the phase field). The error of the *GP10* model increases consistently as we increase b. This indicates that we are using the insufficiently trained GP model even when it is highly uncertain. As expected, the GP models with more training data result in fewer $\phi > \tau$ elements. As we increase b, the required number of Newton-Raphson iterations of the mechanical problem generally increases, with several instances of *GP10* and *GP30* failing to reach the end of the simulation. This creates a trade-off where a higher b requires fewer high-fidelity evaluations but can increase the error and lead to instabilities.

Phase-Field Double-Well

In Figure 3.15 we show the evolution of the phase-field values throughout a simulation for $\omega = 0$ and $\omega = 10$. We observe that this double-well term of the phase field pushes values away from $\phi = 0.5$ and towards $\phi = 0$ and $\phi = 1$. Increasing ω thus has the

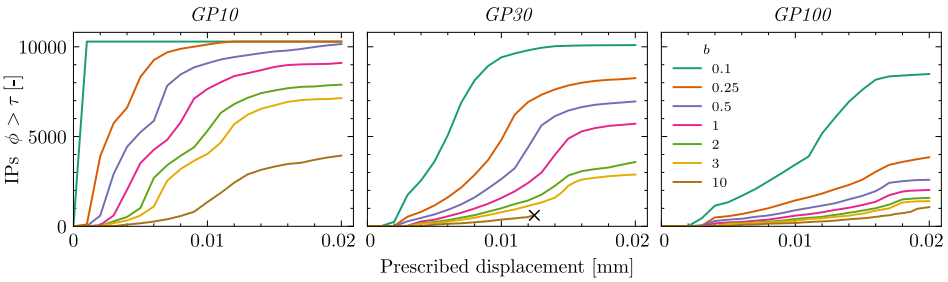


Figure 3.13.: Evolution of the phase field during the simulation for various b values. Default values $\epsilon = 10^{-2}$ and $\omega = 10^{-3}$ are used. The simulation with *GP30*, $b = 10$, failed to converge.

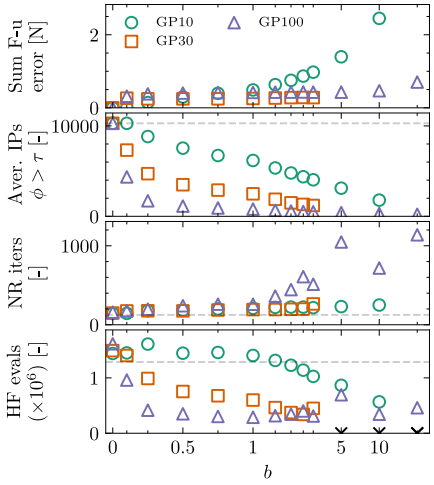


Figure 3.14.: Influence of the opposing force b on the mechanical problem. Note that the x-axis is linear from zero to one, and then scales logarithmically.

potential to increase the number of IPs with $\phi < \tau$, avoiding the high-fidelity model evaluation. Having fewer elements at the interface reduces the number of high-fidelity evaluations, leading to a larger acceleration. However, by setting the parameter ω too high, solving the phase field itself can become challenging. The difference for the $\phi = 1 - \tau$ boundary is smaller, indicating that there are fewer points with ϕ values close to that boundary.

The influence of ω on the result of the simulation for different GPs is shown in Figure 3.16. For *GP10*, the phase field reaches the part of the domain with larger mesh elements. The combination of *GP10* being inaccurate and ω being high makes convergence more difficult. Values below $\omega < 10^{-1}$ appear to have little effect on any of the quantities. For larger values of ω, slightly fewer $\mathcal{C}_{\mathrm{HF}}$ evaluations are required, yet the phase field also becomes unstable, failing to converge for some cases, or requiring significantly more Newton-Raphson iterations. As the impact of this double-well term appears small, we conservatively set $\omega = 10^{-3}$ for the remainder of this work.

Phase-Field Interface Width

The phase-field parameter ϵ influences the width of the interface between the phases and, therefore, the width of the transition region between the constitutive models (the interface width is equal to approximately 4ϵ). To study the influence of the interface more carefully, we perform tests with a constant element size throughout the domain. In Figure 3.17, the influence of ϵ on the required number of Newton-Raphson iterations is shown for different element sizes. A few corresponding full-field plots of the phase-field value for different mesh sizes are shown in Figure 3.18. We can observe two main trends in these figures. Firstly, finding solutions for small ϵ values requires smaller mesh sizes. A minimal interface width is thus required to solve the phase field (but the wider the interface, the more high-fidelity models are evaluated in each iteration). As we decrease ϵ such that the mesh element size is smaller than 4ϵ, the interface width stays one element until the phase field fails to converge. Secondly, the number of Newton-Raphson iterations of the mechanical problem decreases slightly as we increase ϵ. When the phase field fails for some steps, but the overall simulation

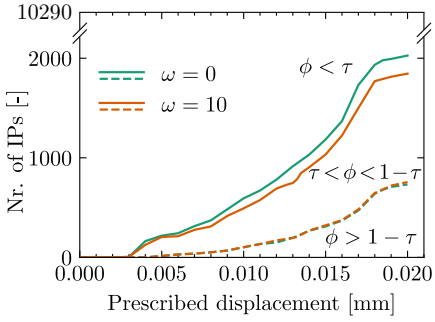


Figure 3.15.: Influence of the double-well term ω on the phase-field values. A higher ω leads to more elements having a ϕ value of either below τ or above $1-\tau$. We use *GP100*, $\epsilon = 10^{-2}$, and $b = 1$.

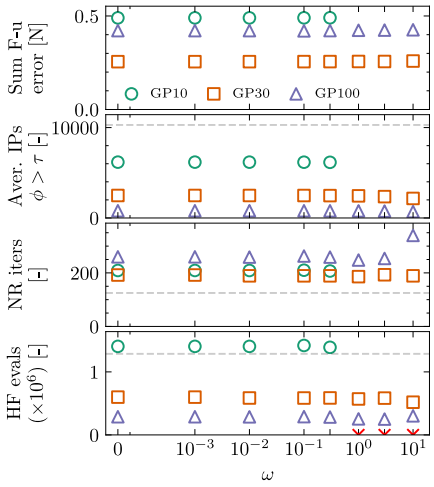


Figure 3.16.: Influence of the double-well term ω on the mechanical problem.

still converges with adaptive steps, it causes the spikes observed for element size 0.004. The general trend indicates that having a transition region spanning multiple elements improves the numerical stability of the mechanical problem.

The mesh used for the other notched plate experiments in this work has a variable mesh element size to capture strain localization with fine elements (0.005) while having coarse elements (up to 0.04) in the rest of the domain, as shown in Figures 3.11 and 3.12. When using such a variable mesh, a separate mesh with a different characteristic element size could be implemented for the phase field as in (Goswami *et al.*, 2019). This brings additional challenges and is not considered here.

Figure 3.19 shows the influence of ϵ for the various GP surrogates on the domain with a variable mesh size. As ϵ decreases, the interface becomes narrower, leading to fewer

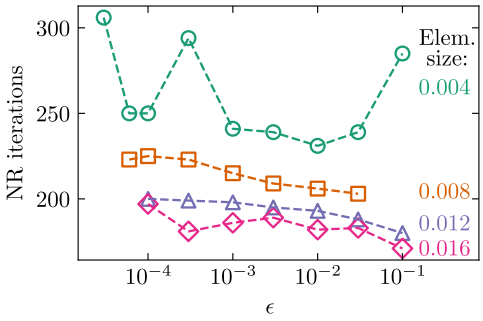
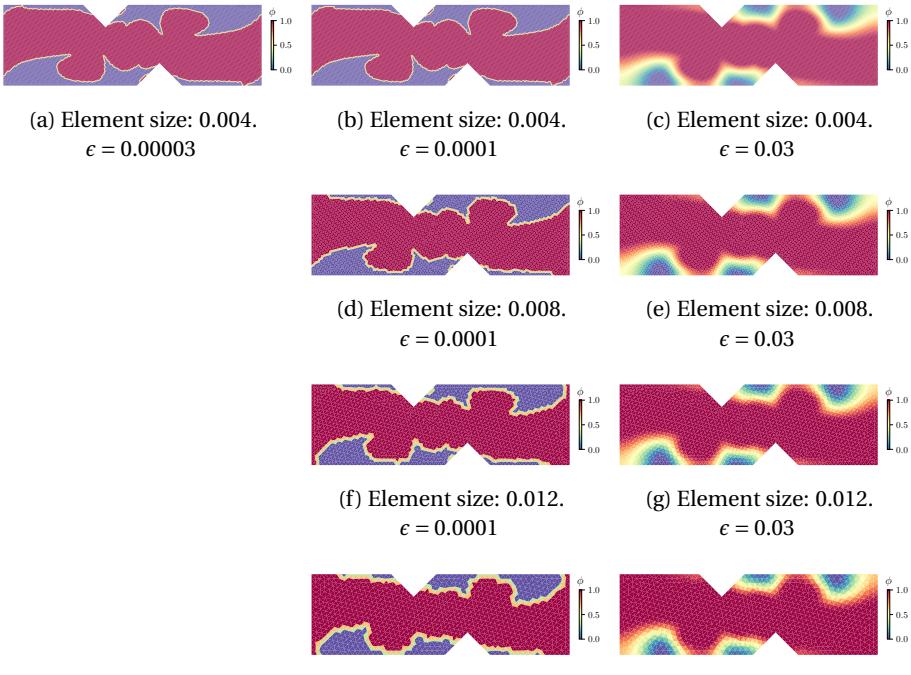


Figure 3.17.: Stability of the phase-field and mechanical problem concerning the element size and phase-field variable ϵ on the notched plate problem. All problems are evaluated on an $\epsilon \in [10^{-5}, 10^{-1}]$ interval, but only converged points are shown. We have used $\omega = 10^{-3}$ and the *GP10* model for all simulations.



(a) Element size: 0.004. $\epsilon = 0.00003$

(b) Element size: 0.004. $\epsilon = 0.0001$

(c) Element size: 0.004. $\epsilon = 0.03$

(d) Element size: 0.008. $\epsilon = 0.0001$

(e) Element size: 0.008. $\epsilon = 0.03$

(f) Element size: 0.012. $\epsilon = 0.0001$

(g) Element size: 0.012. $\epsilon = 0.03$

(h) Element size: 0.016. $\epsilon = 0.0001$

(i) Element size: 0.016. $\epsilon = 0.03$

Figure 3.18.: The phase-field variable at the final time step of the notched plate problem for various element sizes and values of the phase-field interface parameter ϵ.

elements where $\phi > \tau$. For $\epsilon = 10^{-1}$ the error increases, especially for *GP100*. This is because when there is a region with only a few points with high uncertainty, a large ϵ can cause the phase field to not fully switch, hindering accuracy. The difference in accuracy is generally small, and the number of high-fidelity evaluations reduces for smaller ϵ values. Still, as observed earlier, when ϵ is small, the mechanical and phase-field problems become less stable and require more NR iterations or do not converge. Therefore, the default value we have used so far of $\epsilon = 10^{-2}$ appears to be a sensible choice.

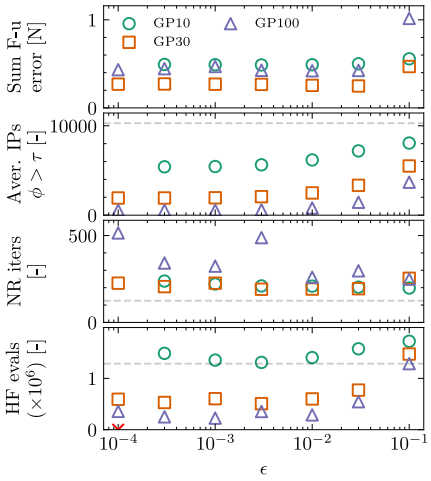


Figure 3.19.: The influence of the phase-field interface parameter ϵ on the mechanical and phase-field problem.

3.4.4. Local Mixture

To avoid being limited by the convergence of the phase field for small ϵ values, we compare the phase field to a simple approach where no phase-field model is solved. In previous demonstrations, phase-field spatial smoothing, a non-local effect, was necessary for convergence. To better study the influence of the transition zone without being limited by having to solve the phase-field PDE, we consider an alternative case with no spatial smoothing. Here, we directly use the driving force to evolve the mixture. We refer to this as the local approach since the mixture of models in each IP is independent of the uncertainty in other IPs. In addition to removing the transition zone, another benefit is that we no longer need to solve the phase field, reducing the complexity of the method. We still use a similar formulation to the phase field, with b opposing the uncertainty. We consider two variations of the local approach. The first version linearly shifts between the models:

$$\phi = \mathcal{U} - b, \quad \phi \in [0,1]. \tag{3.8}$$

In contrast to the phase field, this local shift does not necessarily lead to a spatial transition zone. When two points close in space have very different $\mathcal{U}$ values, the spatial transition can still be sharp. However, the linear shift ensures that a small change in

strain does not cause a completely different constitutive model to be used—the stress remains continuous with respect to the strain. Alternatively, we can use a step function, directly switching between the models when the uncertainty reaches the threshold:

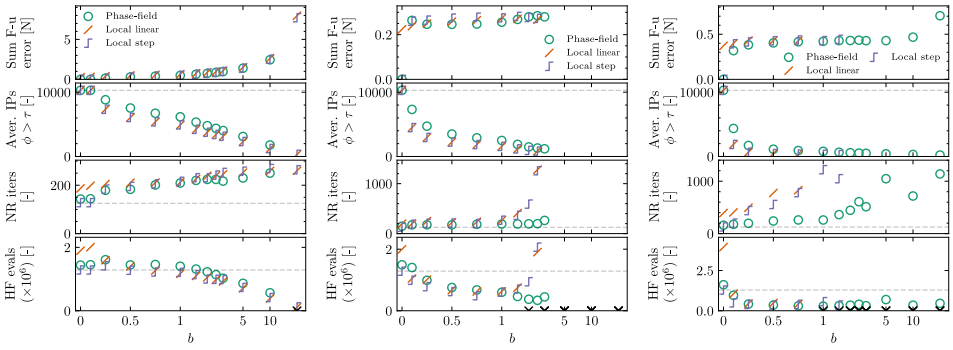
$$\phi = \begin{cases} 0 & \text{if } \mathcal{U} < b, \\ 1 & \text{if } \mathcal{U} \geq b. \end{cases} \tag{3.9}$$

Note that both versions require the high-fidelity model when $\mathcal{U} \geq b$. Similar to the phase-field approach, this method suffers from time-step dependency because the mixture uses the uncertainty of the previous step. Therefore, we also perform several iterations of the staggered approach, with $k_{max} = 3$.

In Figure 3.20, we show how b influences the results for these two local approaches compared to the phase-field approach. Across all simulations, the errors are nearly identical between the models. Because the local methods do not have an interface, they always have fewer IPs that require high-fidelity model evaluations for a given uncertainty. The number of Newton-Raphson iterations is similar when b is low. However, for larger b values, the local approaches require significantly more iterations with *GP30* and *GP100*. In addition, the local approaches start failing for these models as b increases. Removing the interface can thus be beneficial in reducing the number of high-fidelity evaluations for specific settings, yet generally makes the model less stable and less robust.

3.4.5. Surrogate Data

So far, we have shown the influence of the various parameters on the behavior of the mixture of models. We now study how the training dataset size affects the reduction in high-fidelity samples in more detail. To investigate whether this framework can handle complex phase fields, we consider a plate with holes to create numerous local regions



(a) GP 10 (b) GP 30 (c) GP 100

Figure 3.20.: Comparison of the opposing force parameter b for the various approaches. For the phase field we set $\epsilon = 10^{-2}$ and $\omega = 10^{-3}$. Note that the x-axis is linear from zero to one, and then scales logarithmically. All simulations that fail to converge fail due to the mechanical problem.

with high stresses that challenge the GP surrogates and induce a switch. We constrain the left side of the domain and incrementally prescribe a displacement to the right side of the domain. We use $\epsilon = 10^{-2}$, $\omega = 10^{-3}$, and $b = 1$ for the phase field, and do not consider the local approach here.

Figure 3.21 shows the number of high-fidelity evaluations during the simulation as we vary the amount of training data in the GP surrogate. For three of these cases, we plot the full-field phase field at the end of the simulation in Figure 3.22. These figures show the number of high-fidelity evaluations quickly decreasing as the GP is trained with more training data. Still, at a certain point, the downward trend appears to converge, and the number of high-fidelity evaluations is not further reduced. This is because the GP fails to capture localization with strains outside its training data range, which go up to a strain norm of 10%, a choice that must be made prior to running the simulation. This shows the difficulty of creating an appropriate dataset in advance. Different techniques for generating datasets are appropriate for different surrogates. Generally, the dataset should be generated by sampling the input (strain) space such that it covers the expected inputs that occur in the simulation of interest. For path-dependent materials, the input space should also include the internal variables, which is commonly achieved by sampling random, non-monotonic load paths. A high-quality dataset leads to a high-quality surrogate, increasing the efficiency of the approach. However, adequately covering the input space *a priori* is difficult, which is why our proposed method is advantageous.

3.5. Conclusion

This work introduces an approach for mixing constitutive models in finite element analysis and tests its performance by combining a physics-based constitutive model with a data-driven surrogate counterpart. The method dynamically determines where to use each model based on the surrogate uncertainty, creating a spatially varying mixture that preserves accuracy while reducing computational cost. This is achieved by

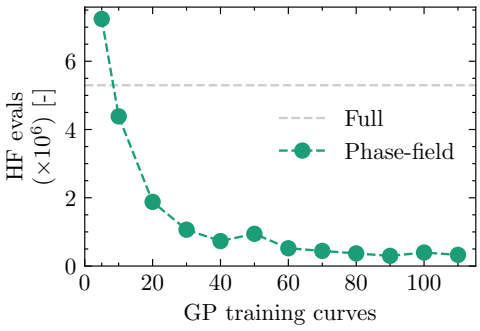
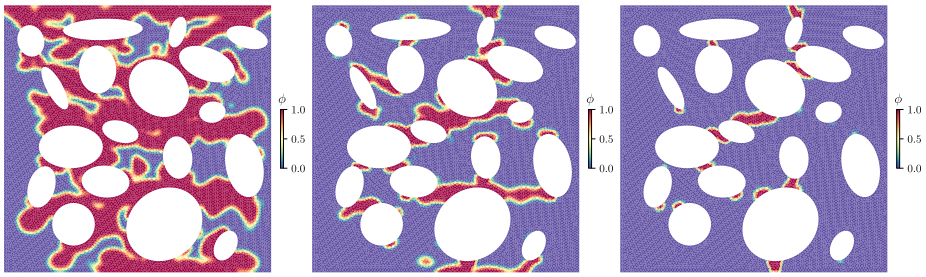


Figure 3.21.: GP training dataset size influence on the number of high-fidelity evaluations. The plot shows the average of 5 runs with different datasets. The full-model is used as a reference. For very small datasets, our approach can use more evaluations than the full model due to the staggered iterations ($k_{max} = 3$).



(a) *GP10* (b) *GP30* (c) *GP100*

Figure 3.22.: Phase-field variable ϕ at the maximum prescribed displacement for surrogates with varying amounts of training data.

solving a phase field based on the surrogate model uncertainty that determines which constitutive model is used. Then, given the mixture of constitutive models, the mechanical problem of interest is solved. Solving the problem in a staggered way introduces time-step dependence, which can be resolved by performing multiple staggered iterations per load step. Running several iterations brings additional costs but is necessary for the robustness of the method.

For computational feasibility, the results have been obtained for single-scale problems, where an analytical constitutive model acts as the high-fidelity model. A Gaussian Process surrogate is used, as it naturally provides an estimate of its uncertainty. Because this surrogate cannot capture history-dependent effects, the experiments consider only monotonic loading scenarios. However, our approach is model-agnostic and does not depend on the specific surrogate or high-fidelity model, but on the accuracy of the surrogate uncertainty, and therefore the conclusions hold for more general multiscale models and history-dependent surrogates. Future work should compare the approach for other material behaviors and loading scenarios, such as cyclic loading.

The use of a phase field was motivated by the hypothesis that the spatial smoothness would improve the numerical stability of the mechanical problem. To test this, we studied the effect of decreasing the size of the interface. As there is a limit to the minimum interface width for the phase field itself, we compared the phase-field approach to an alternative local approach with no spatial smoothing. There, we used the surrogate model if its uncertainty is below a threshold, and the high-fidelity model otherwise, creating a sharp transition between the models. We demonstrated on a notched plate problem that while the local approach can work and reduce the high-fidelity evaluations further, it is less robust than using a phase field. For this reason, the spatial smoothness introduced by the phase field is beneficial.

The results demonstrate that the mixing approach significantly reduces the number of high-fidelity model evaluations compared to the full model. The quality of the surrogate model directly impacts performance—better-trained surrogate models with more data allow for increased computational savings. If a surrogate model can be curated such that it accurately captures the behavior during the full simulation, then the uncertainty remains low, and no additional high-fidelity model evaluations are required.

This framework then adds the extra cost of computing the surrogate uncertainty. However, creating an all-encompassing surrogate model is very challenging, and it is generally difficult to know in advance whether the surrogate model is sufficient. Therefore, this mixture of models uses the surrogate where possible while still enabling an accurate simulation. If the surrogate is poorly trained and can only capture a small part of the behavior, the additional Newton-Raphson iterations to obtain a consistent solution can result in more high-fidelity model evaluations than simply using the high-fidelity model without this approach. It is, however, unlikely that this would be the case in practice. This work thus provides a framework for accelerating multiscale simulations where micromodel evaluations are computationally expensive. The approach works independently of the specific surrogate model and can, therefore, be combined with the latest advances in developing surrogate models to enable multiscale simulations.

A. Gaussian Process regression

A comprehensive treatment of GPs can be found in (Rasmussen, 2003). A Gaussian Process is a collection of random variables, any finite number of which follow a joint Gaussian distribution. GPs are specified by a mean $m(\mathbf{x})$ and a covariance function $k(\mathbf{x},\mathbf{x}')$, written as:

$$f(\mathbf{x}) \sim \mathrm{GP}(m(\mathbf{x}), k(\mathbf{x},\mathbf{x}')). \tag{3.10}$$

For simplicity, it is common to assume a zero mean function, $m(\mathbf{x}) = 0$. The covariance function $k(\mathbf{x},\mathbf{x}')$ specifies the correlation between outputs at different inputs, often capturing how similar inputs should yield similar outputs. In this work, we use a squared exponential (radial basis) kernel function:

$$k(\mathbf{x},\mathbf{x}') = \sigma_f^2 \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\ell^2}\right), \tag{3.11}$$

where σ_f^2 is the signal variance, and ℓ is the characteristic length scale.

Given a training dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i) | i = 1,...,n\}$ with inputs $\mathbf{X} = [\mathbf{x}_1,...,\mathbf{x}_n]^T$ and outputs $\mathbf{y} = [y_1,...,y_n]^T$, we aim to predict the output $f(\mathbf{x}_*)$ at a new test point $\mathbf{x}_*$. Assuming noisy observations $y_i = f(\mathbf{x}_i) + \epsilon_i$ where $\epsilon_i \sim \mathcal{N}(0,\sigma_n^2)$, the joint distribution of the training outputs and the test output becomes:

$$\begin{bmatrix} \mathbf{y} \\ f(\mathbf{x}_*) \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(\mathbf{X},\mathbf{X}) + \sigma_n^2\mathbf{I} & K(\mathbf{X},\mathbf{x}_*) \\ K(\mathbf{x}_*,\mathbf{X}) & K(\mathbf{x}_*,\mathbf{x}_*) \end{bmatrix}\right), \tag{3.12}$$

where $K(\mathbf{X},\mathbf{X})$ denotes the $n \times n$ matrix of covariances evaluated at all pairs of training points, $K(\mathbf{X},\mathbf{x}_*)$ is the $n \times 1$ vector of covariances between training points and the test point, and $K(\mathbf{x}_*,\mathbf{x}_*)$ is the covariance of the test point with itself. The posterior distribution of $f(\mathbf{x}_*)$ given the observations becomes:

$$p(f(\mathbf{x}_*)|\mathbf{X},\mathbf{y},\mathbf{x}_*) = \mathcal{N}(\mu_*,\sigma_*^2) \tag{3.13}$$

$$\mu_* = K(\mathbf{x}_*,\mathbf{X})[K(\mathbf{X},\mathbf{X}) + \sigma_n^2\mathbf{I}]^{-1}\mathbf{y} \tag{3.14}$$

$$\sigma_*^2 = K(\mathbf{x}_*,\mathbf{x}_*) - K(\mathbf{x}_*,\mathbf{X})[K(\mathbf{X},\mathbf{X}) + \sigma_n^2\mathbf{I}]^{-1}K(\mathbf{X},\mathbf{x}_*) \tag{3.15}$$

The predictive mean μ_* serves as our point estimate, while the predictive variance σ_*^2 quantifies our uncertainty about the estimate. This uncertainty naturally increases as we move away from the training data.

There are several methods for optimizing the hyperparameters of a GP. We use the maximum likelihood estimation approach, which finds the hyperparameters that maximize the marginal likelihood of the observed data. The hyperparameters $\boldsymbol{\theta}$ (i.e. $\boldsymbol{\theta} = \{\sigma_f, \ell, \sigma_n\}$) are optimized by maximizing the log marginal likelihood:

$$\log p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}) = -\frac{1}{2}\mathbf{y}^T[K(\mathbf{X},\mathbf{X}) + \sigma_n^2\mathbf{I}]^{-1}\mathbf{y} - \frac{1}{2}\log|K(\mathbf{X},\mathbf{X}) + \sigma_n^2\mathbf{I}| - \frac{n}{2}\log(2\pi) \tag{3.16}$$

We perform this optimization using the L-BFGS-B algorithm, starting from 20 different initial configurations to avoid local optima.

there is a shift from one level of abstraction (or structure) to another, which feels like an upwards movement in an hierarchy, and yet somehow the successive "upward" shifts turn out to give rise to a closed cycle.

Douglas Hofstadter, I Am a Strange Loop

4

MICROSCALE-CONDITIONED SURROGATES FOR DESIGNING MULTISCALE MATERIALS

Emerging sustainable materials increasingly rely on engineered hierarchy and microstructure to achieve control of their properties and mechanical behavior. Optimizing these materials with controllable microstructures requires efficient multiscale simulations. Data-driven surrogate models for the microscale can accelerate multiscale simulations, but require large amounts of data even for a fixed microstructure. When a range of microstructures is considered, as is the case in multiscale optimization, even more data is needed to train a surrogate. To overcome this challenge, we condition a hybrid physics-data surrogate on microstructural variables using a hypernetwork. This approach enables accurate predictions of multiscale mechanical behavior for a mycelium-woodchip composite material, even when trained on small datasets. The conditioned surrogate makes multiscale simulations of functionally graded structures tractable, and we validate it against a full FE^2 simulation. We optimize a graded multiscale disk, and reduce the peak stress by 43% compared to one with a random microstructure. Then, we go one step further, conditioning the network directly on manufacturing variables that can have a complex influence on the microstructure. This is a practical route to engineer the microscale for desired macroscale behavior. This contribution highlights the benefits of microarchitectured structures and demonstrates how conditioned surrogate models enable their multiscale optimization, which will accelerate the development and design of future sustainable materials and structures.

4.1. Introduction

Homogeneous structural materials are widespread due to their uniformity and ease of manufacturing. However, in nature, many heterogeneous materials evolved, such as bones, teeth, wood, and bamboo, as these are more mechanically efficient designs (Yoshinari *et al.*, 2013). These are examples of functionally graded materials, characterized by having continuous spatial variation of material properties at the macroscale. Precisely manufacturing a functionally graded material similarly enables the design of more efficient structures. However, optimizing such a grading requires relating the local microstructure to the macroscopic response.

FE^2 is a concurrent approach to simulate both scales, where microscale representative volume elements (RVEs) are embedded in a macroscale domain. Solving a separate boundary value problem for every quadrature point makes this very computationally demanding. Instead, data-driven models can act as surrogates for the microscale simulations, estimating the homogenized microscale response at a fraction of the computational costs. Many surrogates have been developed, both purely data-driven and based on physical principles, capturing the behavior in materials ranging from hyperelasticity and viscoelasticity to plasticity, damage, and fracture (Ammendolea *et al.*, 2026; Chafia *et al.*, 2025; Fuhg *et al.*, 2024). For hyperelastic materials in particular, several architectures exist that emphasize satisfying all physical requirements, such as thermodynamic consistency and polyconvexity (Klein *et al.*, 2022b; Linka *et al.*, 2021). For linear elastic materials, data-driven approaches have enabled multiscale graded optimization by predicting an effective elasticity tensor from the microstructure (Chandrasekhar *et al.*, 2023; Kim *et al.*, 2021; Zheng *et al.*, 2021). For nonlinear materials, however, surrogates are typically trained to map strains to homogenized stresses for a single, fixed microstructure, and therefore cannot account for varying underlying microstructures. This makes them unsuitable for simulating functionally graded composites, where the microstructure varies across the domain.

One way of overcoming this limitation is by using a surrogate model that takes the complete microstructure as input, for example, using a convolutional neural network (CNN) (Pitz and Pochiraju, 2024; Rao and Liu, 2020) or a graph neural network (GNN) (Storm *et al.*, 2024). Microstructural geometric variables can also be used to directly predict the effective properties of the material (Black and Najafi, 2023). Alternatively, they can be passed as additional inputs, for example to the hidden state in a gated recurrent unit (Mozaffar *et al.*, 2019), or encoded into a small vector before concatenating it to the hidden embedding in a network (Jailin and Baranger, 2025). These approaches have the potential to be used in the optimization of a graded material, but generally still require vast amounts of training data. Meta-learning or transfer learning can adapt or fine-tune a base model to limited new data from a different underlying distribution (Caruana, 1994; Finn *et al.*, 2017; Ghane *et al.*, 2024; Heidenreich *et al.*, 2024), but still require separate data collection for each new setting, making it infeasible for continuous gradings.

An effective approach to reducing the required amount of training data in surrogates is to incorporate a physical bias in the model. Physically Recurrent Neural Networks (PRNNs) have shown a remarkable ability to learn based on tiny datasets, overcoming the bottleneck of costly data generation (Maia *et al.*, 2023). A key feature of PRNNs is

their modularity, inheriting the properties of the embedded microscopic constitutive models. For instance, embedding a history-dependent material model yields a history-dependent PRNN. The material properties of the embedded material model can be modified directly without requiring retraining or additional data collection (Kovács *et al.*, 2025). The PRNN is thus effectively conditioned on these material parameters, despite being trained with only a single parameter value. Recent work has proposed an energy-based PRNN formulation that is thermodynamically consistent (Maia *et al.*, 2026).

In a similar vein, a Deep Material Network (DMN) learns the homogenization behavior of a multi-phase material (Liu *et al.*, 2019a), while maintaining the microscale material models. DMNs have recently been conditioned on the microstructure, such as by interpolating between several DMNs trained on different RVEs, by conditioning on the volume fraction (Li, 2024), and by conditioning on a latent space learned from the microscale geometry (Wei *et al.*, 2025). Since a DMN essentially learns the homogenization scheme, this conditioning can only be done by modifying the weights of the network. The weights of the DMN are thus predicted by a separate neural network; such a second network is generally called a *hypernetwork*. HyperCANs follow the same principle, where, based on the geometry of a unit cell truss lattice, a hypernetwork learns the weight parameters of an input convex neural network that predicts the strain energy density given a deformation gradient (Zheng *et al.*, 2024).

In this work, we opt for PRNNs because they can already account for variations in the microscale material properties, and extend them to be conditioned on microstructural variables using a hypernetwork, which we refer to as a *HyPRNN*. A schematic overview of the methods in this work is presented in Figure 4.1. Our contributions are as follows:

1. *Conditioning a PRNN on microstructural variables.* While traditional PRNNs can already account for differences in microstructural material properties, they cannot account for geometric variations in the microstructure. We enable this using a hypernetwork that learns the PRNN weights based on the microstructural variables.
2. *PRNNs for geometric nonlinearity in finite strains.* To capture geometrically nonlinear effects, we introduce a more flexible nonlinear PRNN encoder. In addition, we modify the energy-based PRNN formulation of (Maia *et al.*, 2026) to account for large deformations.
3. *Demonstration of a tractable multiscale optimization.* With the HyPRNN, we iteratively run a graded multiscale simulation inside an optimization algorithm to find a grading that leads to better mechanical performance.
4. *Conditioning on manufacturing variables.* When the microstructure exhibits a complex geometry that cannot be parametrized explicitly, a common strategy is to condition on a black-box latent space learned by a generative model that captures its properties (Wei *et al.*, 2025). We introduce a new approach that instead conditions directly on the manufacturing variables generating the microstructure, eliminating the need for such an additional model. This approach is demonstrated by conditioning on the inputs of a discrete-element method simulation that creates the microscopic geometry.

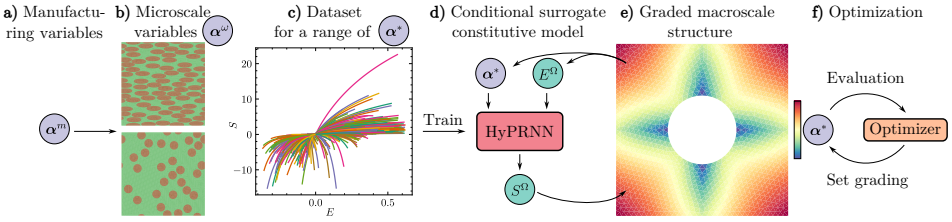


Figure 4.1.: Overview for optimizing a graded multiscale material using a surrogate model conditioned on either microscale variables $\boldsymbol{\alpha}^{\omega}$ or manufacturing variables $\boldsymbol{\alpha}^{m}$.

4.2. Inspiration: mycelium-woodchip composites

While this work focuses on computational methods, we base our requirements on applications such as functionally graded bio-based composites. In particular, we are interested in mycelium-woodchip composites, which are actively researched for the construction, packaging, and insulation industries (Wang *et al.*, 2024), and benefit from microscale structure. Mycelium is considered the vegetative root-like structure of fungi and is composed of hyphae, which are tube-like fibers with a diameter of about 1 [μm]. Growing on a substrate of woodchips or natural fibers, it acts as a binder that forms a lightweight yet stiff biodegradable composite (Gantenbein *et al.*, 2022; Schyck *et al.*, 2026). For instance, Figure 4.2a-b shows woodchips in a mold, to which inoculated water is added, forming a mycelium-bound composite. As an engineered living material, its unique properties include the ability to grow and self-repair (McBee *et al.*, 2021; Nettersheim *et al.*, 2024), and has been explored for mechanical use (Gantenbein *et al.*, 2022; Zhang *et al.*, 2023). Controlling the substrate properties and mycelium growth can tailor a functionally graded material to specific macroscale behavior.

Modeling this macroscale behavior requires the mechanical properties of the individual constituents. Mechanical tests on pure mycelium by Islam et al. revealed that its Young's modulus depends on the density, giving an average of $E = 1.3$ [MPa] for a density of 34 [kg/m^3], and Poisson's ratio $\nu = 0.275$ [-] (Lamé parameters $\lambda = 0.62$ [MPa], $\mu = 0.51$ [MPa]) (Islam *et al.*, 2017, 2018a,b). We adopt these properties and, motivated by their density dependence, also consider a variation in μ. Mycelium is often characterized similarly to foam, and its constitutive behavior is commonly modeled using a hyperelastic material model (Islam *et al.*, 2018a,b; Wang *et al.*, 2024). For the woodchips, isotropy and standard beech properties are assumed, with $E = 13000$ [MPa] and $\nu = 0.375$ [-].

Various aspects of the manufacturing process of a mycelium-woodchip composite are shown in Figure 4.2. Woodchips are inoculated with mycelium spores and placed in formwork, where the mycelium grows over the span of a few weeks to create the composite material. The growth depends on many factors, such as available nutrients, water content, and air. A graded material can be formed by, for example, controlling the distribution of both woodchips and pellets. Figure 4.2c,e shows these constituents, and Figure 4.2d shows the result of numerically simulating the woodchip deposition process. Experimentally finding the optimal grading of woodchip distribution and angle is not feasible, and even numerically this remains challenging.

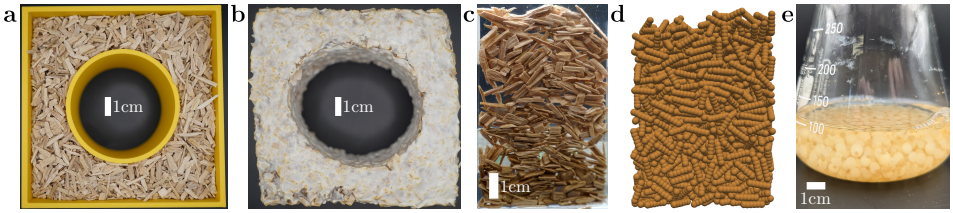


Figure 4.2.: The manufacturing process of a mycelium-woodchip composite. **a** shows formwork in which woodchips are dropped. After inoculation with mycelium and a few weeks of growth, this becomes a mycelium-woodchip composite sample as shown in **b**. One method of creating the woodchip distribution is by depositing them in a container. **c** shows a side-view picture of such a deposition in a thin container, and **d** shows a discrete-element method based gravity deposition aiming to mimic this process. In **e** a picture of pure mycelium pellets is shown.

4.3. Finite strain mechanics

This section briefly introduces the computational methods and notations associated with finite strain continuum mechanics and hyperelasticity. For a reference point $\mathbf{X}$ in the initial configuration of a body undergoing deformation, we can define a function $\boldsymbol{\varphi}$ that maps the point to an updated configuration $\mathbf{x} = \boldsymbol{\varphi}(\mathbf{X}, t)$ at time t. The deformation gradient of the material follows as

$$\mathbf{F} = \frac{\partial \boldsymbol{\varphi}}{\partial \mathbf{X}}, \quad \text{with} \quad J = \det \mathbf{F} > 0. \tag{4.1}$$

Commonly used deformation tensors are the right Cauchy-Green tensor ($\mathbf{C}$) and the Green-Lagrange strain ($\mathbf{E}$):

$$\mathbf{C} = \mathbf{F}^T \mathbf{F}, \qquad \mathbf{E} = \frac{1}{2}(\mathbf{C} - \mathbf{I}). \tag{4.2}$$

A hyperelastic solid derives stress from a stored strain energy density $\Psi = \Psi(\mathbf{F})$. The stress measures are computed as

$$\mathbf{P} = \frac{\partial \Psi}{\partial \mathbf{F}}, \qquad \mathbf{S} = 2\frac{\partial \Psi}{\partial \mathbf{C}}, \qquad \mathbf{S} = \mathbf{F}^{-1}\mathbf{P}, \tag{4.3}$$

where $\mathbf{P}$ is the first and $\mathbf{S}$ the second Piola–Kirchhoff stress tensor. $\mathbf{C}$, $\mathbf{E}$, and $\mathbf{S}$ are symmetric tensors. We employ a simple hyperelastic material model in this work, following a compressible neo-Hookean formulation. With Lamé parameters μ and λ, its energy in 2D is defined as:

$$\Psi(\mathbf{F}) = \frac{\mu}{2}(\mathrm{tr}(\mathbf{C}) - 2) - \mu \ln J + \frac{\lambda}{2}(\ln J)^2. \tag{4.4}$$

The second Piola-Kirchhoff stress then follows as

$$\mathbf{S} = \mu(\mathbf{I} - \mathbf{C}^{-1}) + \lambda \ln J\, \mathbf{C}^{-1}. \tag{4.5}$$

In a two-scale setting (FE^2), the macroscopic problem on domain Ω provides the macroscopic deformation gradient $\mathbf{F}^\Omega$ to an RVE that resolves equilibrium in the microscale domain ω. The microscale problem returns the homogenized stress as a volume

average:

$$\mathbf{P}^{\Omega} = \frac{1}{|\omega|}\int_{\omega} \mathbf{P}^{\omega}\, d\omega. \tag{4.6}$$

4.4. Microstructure-conditioned PRNNs

4.4.1. PRNNs for finite strain

In this work we will use PRNNs as a surrogate for the microscale model, building on earlier work that demonstrated that PRNNs perform well in a finite strain framework (Maia *et al.*, 2024). The PRNN consists in an *encoder - material - decoder* framework, with operations that conceptually mirror computational homogenization in an RVE. The encoder learns to compute microscopic strain quantities from the macroscopic strain input. These microscopic strains are passed through a set of *fictitious material points* employing physics-based microscopic material models to produce microscopic stresses. These material models are the same ones also used as constitutive equations in the RVE for data generation, in this case neo-Hookean. The decoder then combines the microscopic stresses into a macroscopic stress. The general flow of the PRNN used in this work is thus as follows:

$$\mathbf{F}^{\Omega} \rightarrow \mathbf{E}^{\Omega} \underbrace{\rightarrow \hat{\mathbf{E}}^{\omega}_{m_1,\dots,M}}_{\text{encoder}} \underbrace{\rightarrow \hat{\mathbf{S}}^{\omega}_{m_1,\dots,M}}_{\text{material model}} \underbrace{\rightarrow \hat{\mathbf{S}}^{\Omega}}_{\text{decoder}} \rightarrow \hat{\mathbf{P}}^{\Omega}. \tag{4.7}$$

Here, M is a hyperparameter that corresponds to the number of fictitious material points used. Note that these microscopic quantities are not linked to specific points in the micromodel. Depending on the architecture, the distribution of the fictitious internal variables can still closely match those of the RVE (Maia *et al.*, 2026). However, unlike in the RVE, compatibility and equilibrium of these microscopic quantities are not enforced.

In the small-strain plasticity setting that inspired the original PRNN (Maia *et al.*, 2023), most of the nonlinearity comes from the material behavior, which is captured by the material model present in the network. In our finite-strain hyperelastic model, a significant amount of additional geometric nonlinearity can occur. With the material model capturing material nonlinearity, and the decoder conceptually mirroring the homogenization step, the encoder is the natural place for capturing geometric effects. We therefore explore a nonlinear encoder with a high flexibility. In addition, PRNNs can be made thermodynamically consistent (Maia *et al.*, 2026). This works when summing the strain energy densities resulting from the material models, or equivalently, fixing the decoder to be a scaled transpose of the encoder. We use this energy based PRNN as alternative to the nonlinear PRNN.

There are two conditions we want the PRNNs to satisfy. First, in the undeformed state, all microscopic strains should vanish: $\mathbf{F}^{\Omega} = \mathbf{I} \rightarrow \hat{\mathbf{E}}^{\omega}_{m} = \mathbf{0}$. This causes all microscopic stresses and energies to remain zero, leading to a zero predicted stress. Second, to be able to compute the response of the fictitious material points, which are given by a classical neo-Hookean model, the encoder must output valid deformation tensors. For this neo-Hookean model, this means ensuring that $\hat{\mathbf{C}}^{\omega}_{m}$ is positive definite.

Nonlinear PRNN

The first formulation we explore is designed around enabling a high flexibility in the model. The approach we take is to first allow arbitrary nonlinear operations on the inputs, and then constrain this intermediate state to satisfy the two conditions outlined previously.

For the encoder, we use a neural network, NN^E that takes $\mathbf{E}^\Omega$ as input and outputs $3M$ variables (three local strain components per fictitious material point), gathered in $\boldsymbol{\theta}$. Considering the first condition, we apply a zero-shift to ensure $\boldsymbol{\theta}$ is zero in the undeformed state:

$$\bar{\boldsymbol{\theta}} = \mathrm{NN}^E(\mathbf{E}^\Omega) - \mathrm{NN}^E(\mathbf{0}). \tag{4.8}$$

Each $\bar{\boldsymbol{\theta}}_m \in \mathbb{R}^3$ is then mapped to $\hat{\mathbf{C}}^\omega_m$ through a Cholesky-style parametrization,

$$\hat{\mathbf{C}}^\omega_m = \mathbf{L}_m \mathbf{L}_m^\top, \qquad \mathbf{L}_m = \begin{bmatrix} \sigma(\theta_{m,1}) & 0 \\ \theta_{m,2} & \sigma(\theta_{m,3}) \end{bmatrix}, \tag{4.9}$$

where $\sigma(x) = \mathrm{softplus}(x) - \ln(2) + 1$ is a shifted softplus chosen so that $\sigma(0) = 1$. This ensures that when $\bar{\boldsymbol{\theta}}_m$ is zero, $\hat{\mathbf{C}}^\omega_m = \mathbf{I}$, satisfying the first condition. In addition, the resulting $\hat{\mathbf{C}}^\omega_m$ is positive definite by definition, and thus satisfies the second condition.

Algorithm 2 Nonlinear encoder layer in 2D

Require: $\mathbf{E}^\Omega \in \mathbb{R}^{2\times 2}$, neural network NN^E with output size $3M$
Ensure: $\hat{\mathbf{C}}^\omega_m \in \mathbb{R}^{2\times 2}$ for $m = 1, \ldots, M$
1: Evaluate network and apply zero-shift: $\bar{\boldsymbol{\theta}} = \mathrm{NN}^E(\mathbf{E}^\Omega) - \mathrm{NN}^E(\mathbf{0})$
2: **for** $m = 1, \ldots, M$ **do**
3: $\quad \mathbf{L}_m = \begin{pmatrix} \mathrm{softplus}(\theta_{m,1}) - \ln(2) + 1 & 0 \\ \theta_{m,2} & \mathrm{softplus}(\theta_{m,3}) - \ln(2) + 1 \end{pmatrix}$
4: $\quad \hat{\mathbf{C}}^\omega_m = \mathbf{L}_m \mathbf{L}_m^\top$
5: **end for**

$\hat{\mathbf{C}}^\omega_m$ = is passed through the material model to obtain $\hat{\mathbf{S}}^\omega_m$. Then a decoder combines all predicted microscopic stress tensors into the macroscopic stress. Previous work on PRNNs for finite strains compared various decoders (Maia *et al.*, 2024). We adopt a sparse decoder, which combines the stresses in a component-wise manner:

$$\hat{\mathbf{S}}^\Omega_i = \sum_m \hat{\mathbf{S}}^\omega_{m,i} \, \mathrm{softplus}(w_{m,i}). \tag{4.10}$$

Energy PRNN

As an alternative to the nonlinear PRNN, we follow earlier work on creating thermodynamically consistent PRNNs (Maia *et al.*, 2026) and present an energy-based model. The general idea is to compute the strain energy density as a weighted sum of the strain energy densities of the fictitious material points:

$$\Psi^\Omega = \sum_m |w_m| \hat{\psi}^\omega_m \tag{4.11}$$

with w_m a scalar learned by the PRNN.

To satisfy the two conditions discussed earlier, we formulate the encoder based on the true strain $\mathbf{H}^\Omega = \frac{1}{2}\ln(\mathbf{C}^\Omega)$. Here, the matrix logarithm $\ln(\mathbf{C}^\Omega)$ is obtained by applying the logarithm to the eigenvalues of $\mathbf{C}^\Omega$ while retaining its eigenvectors.[1] The encoder is defined as:

$$\hat{\mathbf{C}}^\omega_m = \exp(2\hat{\mathbf{H}}^\omega_m), \qquad \hat{\mathbf{H}}^\omega_m = \mathbb{A}_m : \mathbf{H}^\Omega, \tag{4.12}$$

where $\mathbb{A}_m$ is a learned fourth-order tensor that applies a linear mapping for each fictitious material point. The encoder thus maps strain linearly in log space. The matrix exponential ensures that if $\hat{\mathbf{H}}^\omega_m$ is symmetric, then $\hat{\mathbf{C}}^\omega_m$ is symmetric positive definite. Given that $\mathbf{H}^\Omega$ is symmetric, we can reduce $\mathbb{A}_m$ to have nine independent components and make $\hat{\mathbf{H}}^\omega_m$ symmetric by construction. In Mandel notation $\{\mathbf{H}\} = [H_{11}, H_{22}, \sqrt{2}H_{12}]^\top$ we can write this as $\mathbf{A}_m \in \mathbb{R}^{3\times3}$:

$$\{\hat{\mathbf{H}}^\omega_m\} = \mathbf{A}_m\{\mathbf{H}^\Omega\}, \qquad \mathbf{A}_m = \mathbf{I} + \text{reshape}_{3\times3}(\boldsymbol{\theta}_m), \quad \boldsymbol{\theta}_m \in \mathbb{R}^9. \tag{4.13}$$

In the undeformed state $\mathbf{H}^\Omega = \mathbf{0}$, the linear transformation results in $\hat{\mathbf{H}}^\omega_m = \mathbf{0}$ and $\hat{\mathbf{C}}^\omega_m = \mathbf{I}$. For any admissible deformation, $\hat{\mathbf{C}}^\omega_m$ is symmetric positive definite. Therefore, both conditions hold for every $\mathbf{A}_m$. The identity in $\mathbf{A}_m$ is not strictly required, but it lets $\boldsymbol{\theta}_m$ represent only the deviation of $\hat{\mathbf{H}}^\omega_m$ from $\mathbf{H}^\Omega$. If $\boldsymbol{\theta}_m$ is initialized close to zero, the model starts from the macroscopic response, which simplifies training.

The fictitious material points use $\hat{\mathbf{C}}^\omega_m$ and output a strain energy density $\hat{\psi}^\omega_m$. The decoder, given in Equation 4.11, combines the strain energy densities. The energy PRNN thus has 10 learnable parameters per material point. The stress is obtained by differentiating Ψ, either using automatic differentiation or analytically using the chain rule. As the stress is the exact gradient of a scalar potential, the tangent is symmetric with hyperelastic materials points and the network is thermodynamically consistent. However, polyconvexity and rank-one convexity are not guaranteed, and any symmetry in the RVE is not enforced.

Algorithm 3 Energy PRNN with a neo-Hookean material model in 2D

Require: $\mathbf{H}^\Omega \in \mathbb{R}^{2\times2}$, parameters $\boldsymbol{\theta} \in \mathbb{R}^{9\times m}$, $\mathbf{w} \in \mathbb{R}^m$
Ensure: Macroscopic stress $\hat{\mathbf{S}}^\Omega$

$\hat{\Psi}^\Omega = 0$
for $m = 1, \dots, M$ **do**
  $\mathbf{A}_m = \mathbf{I} + \text{reshape}_{3\times3}(\boldsymbol{\theta}_m)$
  $\{\hat{\mathbf{H}}^\omega_m\} = \mathbf{A}_m\{\mathbf{H}^\Omega\}$ ▷ Mandel notation, (4.13)
  $\hat{\mathbf{C}}^\omega_m = \exp(2\hat{\mathbf{H}}^\omega_m)$
  $\ln \hat{J}^\omega_m = \text{tr}\,\hat{\mathbf{H}}^\omega_m$
  $\hat{\psi}^\omega_m = \frac{\mu_m}{2}(\text{tr}\,\hat{\mathbf{C}}^\omega_m - 2) - \mu_m \ln \hat{J}^\omega_m + \frac{\lambda_m}{2}(\ln \hat{J}^\omega_m)^2$ ▷ Specific to a neo-Hookean model.
  $\hat{\Psi}^\Omega = \hat{\Psi}^\Omega + |w_m|\hat{\psi}^\omega_m$
end for
$\hat{\mathbf{S}}^\Omega = \partial\hat{\Psi}^\Omega / \partial\mathbf{E}^\Omega$

[1] Gradients of the eigendecomposition are singular for repeated eigenvalues (e.g., when $\mathbf{C}^\Omega = \mathbf{I}$). We therefore evaluate the logarithm for the symmetric positive definite 2×2 matrix with a closed-form expression that avoids explicit eigenvectors.

4.4.2. A hypernetwork-based PRNN

PRNNs cannot generally be conditioned on microstructural parameters apart from material parameters. Different routes can be explored to still include them in the PRNN architecture. The straightforward approach to condition a data-driven model on additional parameters is by concatenating the variables as additional inputs. For the PRNN, this approach conflicts with the role of the encoder, which is meant to map only the macroscopic strain onto a latent representation of microscopic strains. While the nonlinear encoder could in theory support additional inputs, the energy-based formulation would require further modifications.

Instead, we preserve the existing PRNN architecture, and use a hypernetwork $\mathcal{H}$ that learns the PRNN encoder and decoder parameters based on microstructural variables. In effect, the hypernetwork learns a continuous mapping from microstructural variables to the space of PRNN weights, rather than producing a single set of weights tied to one microstructure. The encoder and decoder weights are learned by the same hypernetwork, and the only trainable parameters then become the hypernetwork parameters themselves. These are learned in a single training stage, directly from RVE simulation data on different microstructures. We provide a schematic overview of a nonlinear HyPRNN in Figure 4.3, and an energy HyPRNN in Figure 4.4.

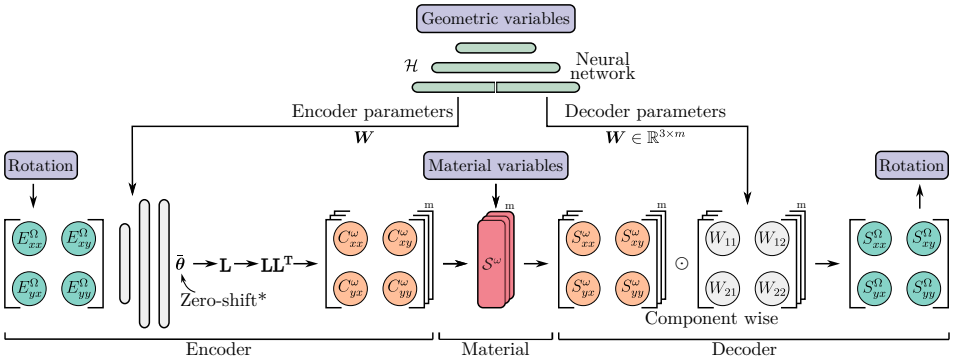


Figure 4.3.: Overview of the nonlinear HyPRNN. *Zero-shift: see Equation 4.8.

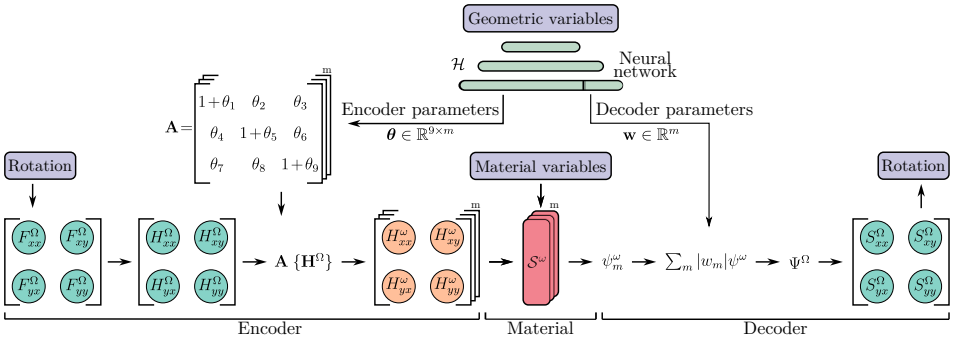


Figure 4.4.: Overview of the energy HyPRNN. $\{\cdot\}$ refers to a tensor in Mandel notation.

Geometric parameters of the microscale are used as inputs to $\mathcal{H}$. Material parameters that affect the microscopic constitutive model can be directly modified in the fictitious material points on a per-sample basis. This has been shown to work even for large changes of parameters, and no retraining of the network is required (Kovács *et al.*, 2025). Microscale orientations can be accounted for by rotating the input deformation and output stress accordingly.

4.5. Results: conditioning on microscale variables

In this results section, we compare a HyPRNN against a neural network and numerically validate it in a multiscale simulation, before performing a graded multiscale optimization.

4.5.1. Numerical setup

This section describes the software and modeling choices used for the numerical experiments. All code, as well as some pretrained surrogate models, is accessible on GitHub at `https://github.com/SLIMM-Lab/hyprnn`.

Parametrization and data generation

To train the surrogate models, a dataset is generated by simulating RVEs spanning a range of microstructural parameters. The RVEs consist of random packings of ellipses, representing the woodchips, surrounded by the mycelium. We consider four parameters: the ellipse aspect ratio r, the woodchip volume fraction V_f, the mycelium shear modulus μ, and the ellipse orientation ϕ. The first two are geometric parameters, and μ is a material parameter As all ellipses follow the same orientation ϕ, it can be accounted for by rotating the deformation to align with the ellipse direction before passing it to the PRNN. Since we limit ourselves to 2D in this work, this is computed as:

$$\bar{\mathbf{F}} = \mathbf{Q}^T \mathbf{F} \mathbf{Q}, \quad \mathbf{Q} = \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix}. \tag{4.14}$$

The output is then back-rotated to obtain the actual stress $\mathbf{S}$.

$$\mathbf{S} = \mathbf{Q} \bar{\mathbf{S}} \mathbf{Q}^T \tag{4.15}$$

This way the orientation can be included by pre- and post-processing the surrogate's inputs and outputs, and does not need to be included in the dataset.

The material variable μ is included in the dataset, but it is not necessary for the random ellipse packing of the RVE. This packing is parametrized based on r and V_f. V_f determines the number of ellipses, and r is are shared for all ellipses within one RVE. For a given aspect ratio, we compute the length and width of the ellipses such that the area of each ellipse remains constant, covering 0.8% of the RVE domain. Ellipses are then iteratively placed at random coordinates in a square domain until the desired volume fraction is reached. Ellipses crossing a boundary are wrapped to the opposite

boundary to maintain a periodic geometry. If a newly placed ellipse overlaps with an existing one, it is discarded and a new random location is attempted. Gmsh (Geuzaine and Remacle, 2009a) is used to mesh the domain. To produce a robust mesh, we verify that the number of nodes on opposite edges matches, and restart the procedure if this is not the case.

We simulate the RVEs using *DOLFINx* (Baratta *et al.*, 2025) with periodic boundary conditions. The load direction is selected randomly in **F** space, and the RVE is loaded monotonically for 50 time steps up to $\|\mathbf{F}-\mathbf{I}\| = 0.5$. If a time step fails to converge, the state is reverted to the last converged step, and the step is repeated with eight sub-increments. However, data from the intermediate sub-increments is not included in the dataset. If sub-increments also fail to converge, the simulation is stopped and only the steps up to the last converged step are included in the dataset.

The influence of changing the microstructural parameters on the stress response is shown in Figure 4.5. In these responses, only the E_{xx} component is loaded for illustrative purposes. In tension, the simulations succeed for all 50 time steps. However, in compression, the simulations fail significantly earlier. Similar stress differences to those in this figure also occur for the other stress directions.

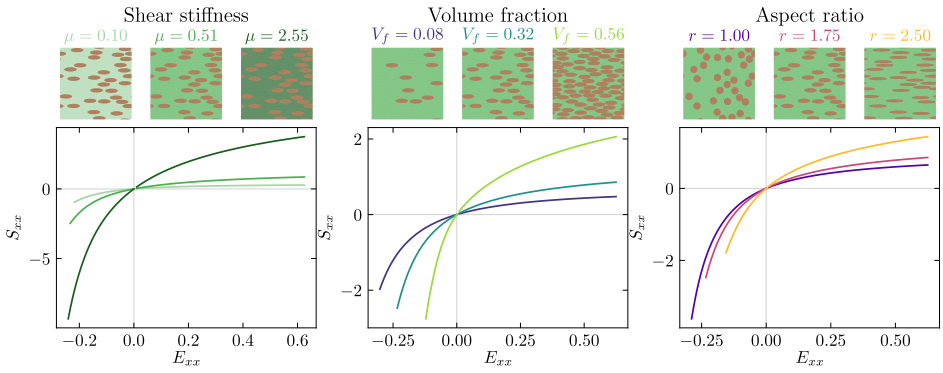


Figure 4.5.: Influence of the microstructural parameters. $\mu = 0.51$, $V_f = 0.32$, $r = 1.75$ are used as default values; one of the variables is set to the minimum or maximum value in the dataset. The sample is displaced in the E_{xx} component only, in both tension and compression.

Surrogate model training

The HyPRNN, including the embedded material model, is implemented in JAX (Bradbury *et al.*, 2018). Two-thirds of the dataset is used for training, and the remainder is split evenly between validation and test sets. The training is done in **E**-**S** space, since their symmetry allows for reducing the input and output dimensions of the surrogates, which is generally advantageous for data-driven models. An L2 loss is minimized using the Adam optimizer (Kingma and Ba, 2014), with a fixed learning rate of 0.001, until the validation loss has not improved for 50 epochs. All computations, including the training of the surrogates, are performed on a CPU. Just-in-time compilation and

vectorized computations enable efficient training and inference. Since there is no history dependency considered here, all steps along a load path are computed in a single batched operation, together with steps from other load paths. A batch size of 2 load paths is used for all models.

Multiscale simulations

All macroscopic simulations are also performed in *DOLFINx*, where the *dolfinx_materials* (Bleyer, 2024) library enables the use of RVE micromodels and JAX-based surrogates as constitutive models. For surrogate-based multiscale simulations, automatic differentiation is used to compute the tangent stiffness tensor. Just-in-time compilation is used here too, resulting in a longer computation time for the first time step, but then enabling much faster subsequent steps. For the RVE-based multiscale simulations, we compute the tangents with central difference for ease of implementation. The RVE is solved with a solver tolerance of 10^{-10}, the central difference perturbation is 10^{-8}. The macroscale problem is solved with a tolerance of 10^{-6}. Substepping with eight sub-increments is applied at the macroscale in case of non-convergence. The micromodels are evaluated in parallel.

4.5.2. Conditioning comparisons

In this section, we compare the different HyPRNN architectures. We additionally compare their performance to that of a standard neural network. Since PRNNs with a linear encoder can extrapolate material parameters after having been trained with only one (Kovács *et al.*, 2025), we first separately consider the effect of conditioning only on material parameters or only on geometric parameters, before combining them.

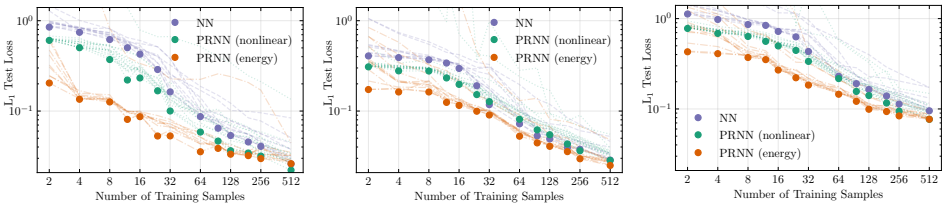
Three datasets, $\mathcal{D}^{mat}$, $\mathcal{D}^{geo}$, and $\mathcal{D}^{all}$, are generated to test the influence of different microstructural parameters. For $\mathcal{D}^{mat}$, only μ varies, for $\mathcal{D}^{geo}$, V_f and r change, and for $\mathcal{D}^{all}$ all three parameters are varied. An overview is given in Table 4.1. The shear modulus μ is selected randomly in the range $[0.10, 2.55]$. Since whole ellipses are placed one at a time, V_f is restricted to discrete increments. Rather than generating a new mesh for every (V_f, r) combination that would arise from uniform sampling of r, we draw from a predefined set of (V_f, r) meshes. The combinations are a grid with $V_f \in \{0.08, 0.16, 0.24, 0.32, 0.40, 0.48, 0.56\}$, and $r \in \{1.00, 1.25, 1.50, 1.75, 2.00, 2.25, 2.50\}$, giving a total of 49 combinations.

Table 4.1.: Overview of the three datasets. Fixed parameters show their value; varying parameters show their range.

	Parameters			# Samples		
	μ	V_f	r	Train	Val	Test
$\mathcal{D}^{mat}$	[0.10, 2.55]	0.40	2.00	2048	512	512
$\mathcal{D}^{geo}$	0.51	$\{0.08, \ldots, 0.56\}$	$\{1.00, \ldots, 2.50\}$	2048	512	512
$\mathcal{D}^{all}$	[0.10, 2.55]	$\{0.08, \ldots, 0.56\}$	$\{1.00, \ldots, 2.50\}$	2048	512	512

We create learning curves for different hyperparameter settings of the neural network, the nonlinear HyPRNN, and the energy HyPRNN. For $\mathcal{D}^{mat}$, no hypernetwork is necessary in the PRNN as μ directly affects the fictitious material points. The ability of a PRNN to predict for different material parameters was already shown for plasticity in (Kovács *et al.*, 2025). In contrast to that study, where only a single value is seen in training, the material parameter is varied in the training data here, in order to allow for a meaningful comparison with a neural network. The neural network, like the nonlinear PRNN, predicts $\mathbf{S}^{\Omega}$ from $\mathbf{E}^{\Omega}$, with microscopic variables as additional inputs.

The hyperparameters include the choice of the activation functions and the network size, which for the PRNN corresponds to the number of material points. A more detailed overview of hyperparameters considered is given in Appendix A. Each configuration is trained 10 times per number of training samples with different samples drawn from the training pool, and the average loss over these 10 training runs is plotted in Figure 4.6a. Each plot includes learning curves across a wide range of hyperparameters, and therefore different levels of complexity. This avoids having to either separately perform model selection for each dataset size, or first perform model selection on the largest dataset size and then risk biasing our results on smaller dataset sizes due to overfitting. The lowest average loss per type is marked with a circle, giving an envelope of the best models. All models improve consistently as more training data is added. The nonlinear PRNN achieves lower losses than the NN. The energy PRNN performs better still, particularly in the low-data regime, where it reaches a low loss with only a few samples. As the number of samples increases, the nonlinear PRNN and NN losses catch up with the energy PRNN. Still, it is remarkable how the energy PRNN can achieve consistently high accuracies here even though — in contrast to the nonlinear encoder one — it has no internal mechanism to capture geometric nonlinearities.



(a) $\mathcal{D}^{mat}$ (b) $\mathcal{D}^{geo}$ (c) $\mathcal{D}^{all}$

Figure 4.6.: Overview of learning curves on the different datasets, plotted on a log-log scale. Each line represents the learning curve for one configuration of hyperparameters, averaged over 10 runs. The lowest point for each type of model (NN, nonlinear (Hy)PRNN, energy (Hy)PRNN) is highlighted to show the overall trend.

The same process is repeated for $\mathcal{D}^{geo}$, and the results are shown in Figure 4.6b. Here, the conditioning happens solely via the hypernetwork. A similar trend to before is observed, with the energy HyPRNN performing better in the low-data regime, but all models obtaining similar losses for large sample sizes. The error decreasing to a low value indicates that the hypernetwork approach to conditioning a PRNN is effective.

Figure 4.6c shows the resulting learning curves for $\mathcal{D}^{all}$, combining both the geometric and material parameters. The first thing that can be observed is that the error is higher

than in the other two learning curve plots. This is because the dataset is more complex, and also reaches higher stress values, making the absolute difference larger than the relative difference in prediction. Due to the definition of the range of μ, the average μ in $\mathcal{D}^{all}$ is also higher than μ in $\mathcal{D}^{geo}$. The gap in Figure 4.6a between the energy PRNN and the NN in the low-data regime is larger than the gap in Figure 4.6b, confirming the particular benefit that the PRNN with a linear transformation in the encoder offers in that it can transfer between different material property values without being trained for it. The nonlinear PRNN loss is closer to that of the NN, showing that it does not transfer material properties as well as an energy PRNN. This is because the nonlinear PRNN relies less strongly on the physics in the material model for reproducing the data.

For all the datasets, the differences between the types of models are significant, and variations for specific hyperparameters within one type are smaller. This indicates that the overall results are not overly sensitive to the specific settings used, such as the number of material points or a specific number of neurons in a layer. Sigmoid activation performs much better than ReLU activation in all networks (NN, nonlinear encoder, hypernetwork). Furthermore, HyPRNN models with only the mycelium material points perform better than those with additional woodchip material points when there is little available data. These effects on the learning curves are shown in Appendix A. We suspect that the large contrast between the stiffness of the mycelium and the woodchip degrades training performance. This phenomenon deserves more attention in future work.

A test curve from $\mathcal{D}^{all}$ with predictions from several representative models is visualized in Figure 4.7 with pair-wise stress-strain plots. For this specific curve, all surrogates predict the same overall shape, but quantitatively the nonlinear HyPRNN performs best. The NN trained with 32 samples shows a significantly higher error compared to the other curves. The energy PRNN predictions also give a relatively high error for this sample. The nonlinear HyPRNN used here has twelve material points, a single hidden layer in the encoder, and three hidden layers in the hypernetwork, which is eight nodes wide and uses sigmoid activations. The energy HyPRNN has twelve material points and a hypernetwork with a single hidden layer with sigmoid activation. The neural network has four hidden layers of 64 nodes, with sigmoid activation.

4.5.3. FE^2 validation

Dataset losses quantify pointwise accuracy, but do not fully correlate with the surrogate performance in a full FE^2 simulation. The nonlinear interactions between the points might cause the solution to drift, the tangent stiffness plays a direct role, and the distribution of explored points differs from the test set distribution. We therefore validate our method on a simple structural example: a 3-point bending beam. The beam measures 4.0×0.3 [m], and the top is vertically loaded in five increments over a width of 0.2 [m] at the midspan. Symmetry is used to simulate only the left half of the beam, fixing the bottom left corner, and preventing horizontal movement at the symmetric edge. Note that this setup somewhat overconstrains a typical 3-point bending beam, as it does not allow for any horizontal movement. The applied grading of the simulation is shown in Figure 4.8, spanning a wide range of the considered parameters. The volume fraction gradually increases along the beam height; the shear modu-

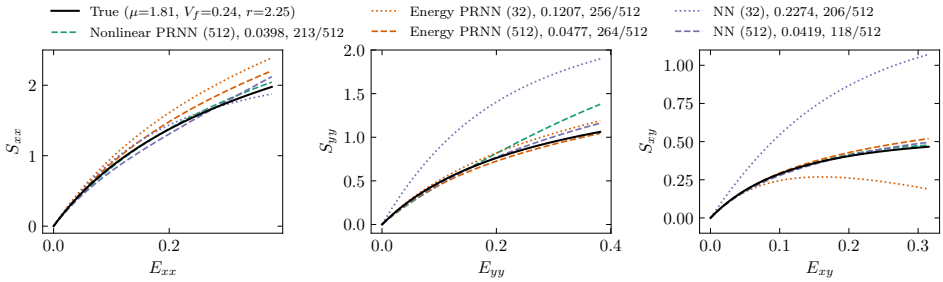


Figure 4.7.: For specific model architectures, the instances with the median test losses are selected (out of 10 training runs) such that the responses represent that of a single training run. For these models, predictions are made on the sample for which the energy HyPRNN with 32 samples has the median error. The legend description is as follows: [model name (number of training samples), error on this sample, rank of sample error out of all test errors (1 is best, 512 is worst)]. The nonlinear HyPRNN and NN perform better on this sample than on their median sample, making the quality of their predictions in this plot lie on the optimistic side.

lus increases towards the beam center. The woodchip aspect ratio and orientation, in contrast, jump discontinuously between an inner and an outer band along the height.

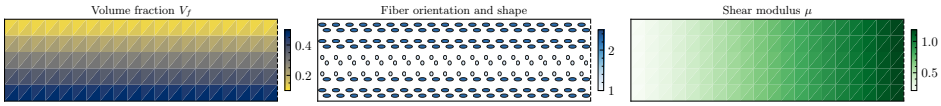


Figure 4.8.: The applied grading throughout the domain of the 3-point bending beam. Only the left half of the domain is visualized. The volume fraction gradually increases from top to bottom. The woodchips are oriented horizontally and slender at the top and bottom, and more rounded and vertically oriented along the middle of the beam. The mycelium shear modulus is low near the supports, and increases gradually towards the center.

For the energy HyPRNN and the NN, two variations are considered, one with 32 training samples and one with 512 training samples. For the nonlinear HyPRNN, we only consider a model trained on 512 samples. A comparison of the displacement fields of surrogate-based simulations to the RVE-based simulation is visualized in Figure 4.9. In the RVE-based reference solution the midspan deflection is 0.115 [m] with strains reaching 27%. All HyPRNN-based surrogates show relatively good agreement, and the energy HyPRNN with 512 samples shows a near-perfect match. While the NN and nonlinear HyPRNN with 512 samples reach similarly low errors on the test set, their displacements in these multiscale simulations are less accurate than the energy HyPRNN. The NN with 32 samples shows very poor performance. In Appendix B, the stress strain paths at a few quadrature points are shown, clearly showing how the neural network surrogate produces a nonzero stress in the undeformed state, whereas the HyPRNN models do not.

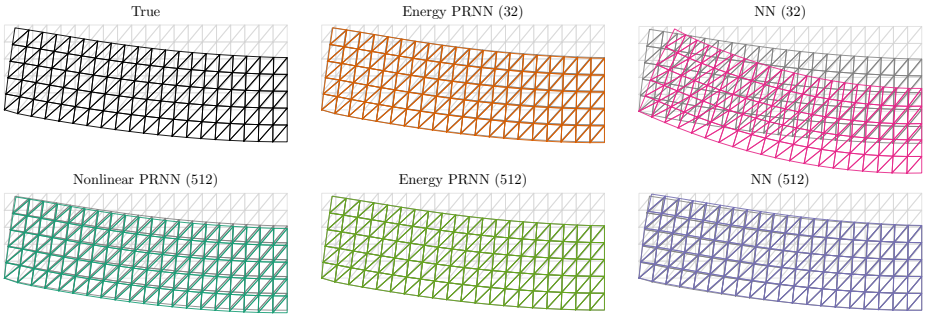


Figure 4.9.: Deformations following the multiscale simulations of a 3-point bending beam. Each plot shows the initial mesh in light gray, the ground truth RVE-based FE^2 simulation in dark gray, and the surrogate-based simulation.

We compare the computational cost of the different simulations. All computation times are obtained on a standard desktop PC with an Intel Xeon W-2223 processor. As discussed earlier, the RVE-based simulation uses central differences to compute the tangent, which is suboptimal in terms of the simulation time. The results are presented in Table 4.2, showing the much lower simulation time with the surrogate models. The data generation and training times are one-time costs, whereas the online simulation time is repeated for each new graded simulation. Still, this large data generation time illustrates the relevance of data requirements for surrogate modeling in multiscale analysis. The first and remaining time steps are reported separately, since the first time step includes the just-in-time compilation of the surrogates. The stark contrast between the RVE and surrogate based simulation time highlights the benefit of surrogate-based simulations. This contrast would only increase for larger macroscopic simulations with more elements, or those running for more time steps.

Table 4.2.: Computation wall-time in seconds for multiscale simulations with the different constitutive models and their number of training samples indicated in brackets. For the RVE-based simulation, initialization includes creating the RVE meshes. Both the meshing and the RVE simulations are performed in parallel across 8 processes. The RVE simulation and data generation timings are from a single run; the surrogate training and simulation timings are averaged across three runs. Data-generation times are identical because they use the same data pool.

Constitutive model (training samples)		RVE	Energy PRNN (32)	Energy PRNN (512)	Nonlinear PRNN (512)	NN (512)
Offline stages	Data generation	–	2035.97	32575.51	32575.51	32575.51
	Training	–	53.98	230.86	1134.89	410.09
Online simulations	Total	5900.52	2.41	2.17	2.12	1.00
	Initialization	137.49	0.18	0.25	0.44	0.37
	Time step 1	965.75	2.16	1.85	1.56	0.59
	Time steps 2-5	4797.27	0.06	0.07	0.12	0.04

4.5.4. Graded multiscale optimization

We now turn to using our surrogate as a constitutive model to optimize a multiscale graded material. Disks subject to rotation or internal loads are commonly studied in the functionally graded materials literature (Abdalla *et al.*, 2020; Bayat *et al.*, 2008; Durodola and Attia, 2000), where a graded thickness is typically used to reduce peak stresses. We adopt a similar setup but grade the microstructural parameters rather than the thickness under an internal pressure. An overview of the structure and boundary conditions is provided in Figure 4.10. The goal is to minimize peak stresses. The parameters only vary radially, allowing for a 1D parametrization. The aspect ratio and woodchip angle are combined into r_{combi} to enable rotated woodchips in a radially consistent manner. Specifically, $r_{\text{combi}} \in [0.4, 2.5]$, where values below one are equivalent to setting $r = 1/r_{\text{combi}}$ with angle $\phi = 90°$ with respect to the radial direction, while for values above one, $r = r_{\text{combi}}$ and $\phi = 0°$. In this parametrization, the woodchips can thus only be either parallel or perpendicular to the radial direction. The optimization variables are V_f, r_{combi}, and μ, and we use the energy PRNN with 512 training samples.

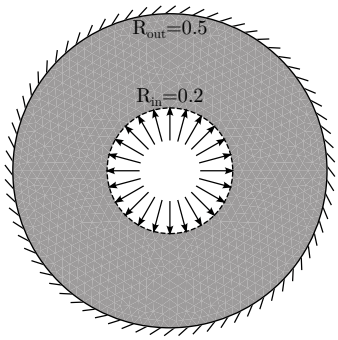


Figure 4.10.: Problem setup for the disk loaded with internal pressure. A distributed load is applied to the inner boundary, while the outer boundary is fixed.

The optimization problem is defined as minimizing the peak Cauchy stress under a given load, subject to a maximum acceptable deformation. Accounting for symmetry, only one quarter of the disk is simulated. The maximum acceptable deformation is set to 0.05 (16.6% of the thickness) through a linearly scaling penalty on the loss when exceeded. We use the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) (Hansen and Ostermeier, 2001), an off-the-shelf, iterative, derivative-free algorithm, to optimize our objective function. Alternative optimization algorithms, such as Bayesian optimization, are also expected to work, and the choice of optimizer is not expected to affect our conclusions.

Three cases are compared: one random, one uniform, and one graded with four control points per variable through the thickness, between which the values are linearly interpolated. The first of these is included as an unoptimized reference solution, while the other two represent optimization problems with increasing flexibility (three versus twelve parameters). For the random initialization, the angle and aspect ratio are randomly chosen, μ is set to 1.5, and the volume fraction is modified to exactly satisfy the deformation constraint using the bisection method, leading to a peak stress of 4.34

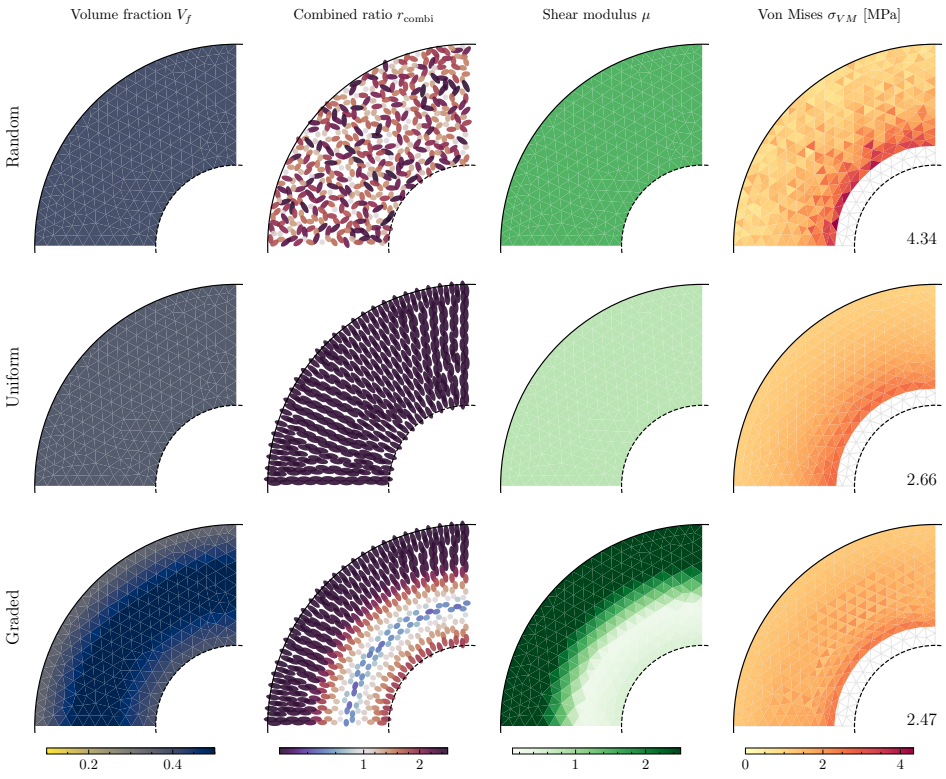


Figure 4.11.: The optimized graded structures and their resulting average stresses. The average stresses are obtained by first computing the Cauchy stress, and then using the von Mises formula. The maximum stresses are 4.34, 2.66, and 2.47 [MPa] for the random, uniform, and graded structure respectively. The von Mises plot shows the stresses in the deformed configuration.

[MPa]. The optimized results are visualized in Figure 4.11. An efficient uniform parametrization with a low peak stress of 2.66 [MPa] is reached after 1099 simulations. The graded structure further decreases this peak stress to 2.47 [MPa] after 2163 simulations. The optimization finds that setting r_{combi} below one is better inside the disk, rather than directly at the edge.

4.6. Results: conditioning on manufacturing variables

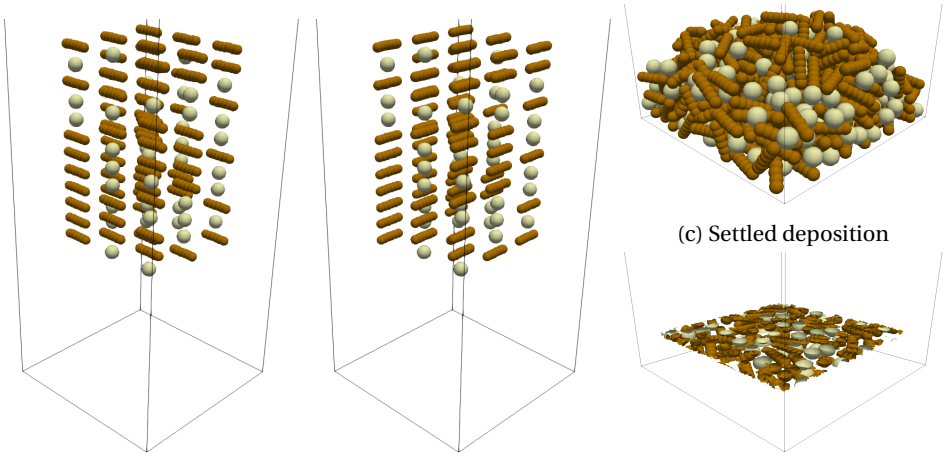
Precisely controlling the microscale geometry as assumed in Section 4.5 is not yet feasible for most materials. Manufacturing process variables influence the microscale, but finding an exact relation between these variables and the microscale geometry is not straightforward. We therefore showcase the ability to condition the PRNN on variables from the manufacturing process. This serves as a computational proof-of-concept aiming to inspire future experimental work.

Mycelium-woodchip composites can be created by depositing woodchips from a bag or container into formwork, and then inoculating them with mycelium. We simulate this deposition process and consider two variables in the process. First, small round mycelium pellets can be added to the mix, decreasing the resulting woodchip volume fraction. We assume that these have the same material properties as the mycelium that grows in the formwork. Second, the initial orientation of the dropped woodchips can be controlled, for example by using a sieve, resulting in anisotropy.

4.6.1. Dataset

To mimic the deposition process of the woodchips, we use Yade, a discrete-element method simulator (Smilauer *et al.*, 2023). The general approach involves initializing the woodchips as clumps of spheres, and dropping them inside a 3D box with gravity. As we focus here on demonstrating our approach, we make the pragmatic choice of not calibrating against experiments. Each woodchip is modeled as a rigid stack of connected spheres. All woodchips are initialized with the same orientation θ, as visualized in Figures 4.12a & 4.12b, but a small random velocity is added to each to avoid an unrealistically aligned result. As a result, most woodchips deviate to some extent from their initial orientation, but the remaining effect does lead to anisotropy. The pellets are modeled as single spheres, with a radius 1.5 times as large as the spheres in the woodchips. We control the *pellet fraction*: the ratio between the volume of the pellets and the volume of the woodchips before deposition. An example of a settled deposition is visualized in Figure 4.12c.

Once the woodchips have settled, the configuration is sliced by a horizontal plane at a height of four times the radius of the spheres used to model the woodchips, as shown in Figure 4.12d. Only a central section of the box is included to avoid boundary effects of the 3D deposition. Each intersected sphere is mapped to a disk in a 2D simulation (without gravity), whose radius equals the radius of the sphere's cross-section at the cutting height. This 2D simulation has periodic boundary conditions, and a relaxation phase allows clumps that were overlapping other clumps at the boundary to untangle. Then, a convex hull around each clump is drawn, before shrinking each hull to avoid

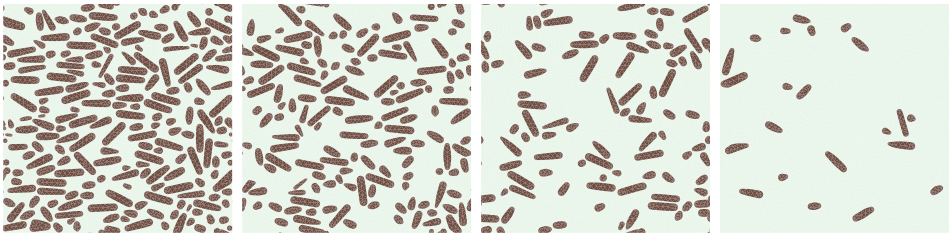


(a) Initialization θ =0° (b) Initialization θ =90° (c) Settled deposition (d) Sliced section

Figure 4.12.: 3D visualizations of the deposition process with a 30% pellet fraction. The settings differ between the visualizations. In the actual deposition, the woodchips and pellets are smaller and more numerous than shown here.

overlap. Finally, this geometry is meshed using Gmsh to create a periodic 2D mesh, as shown in Figure 4.13. Only meshes from a single initial orientation are required since it can be accounted for by rotating the deformation and stress tensor as discussed in Section 4.5.1; here we set the initial orientation to 0°.

This approach of taking a 2D slice from the 3D deposition does somewhat disconnect the manufacturing variable from the generated microstructure; the final model used is only indirectly related to the process that simulated it. Ideally, the 3D deposition would be directly used, but we transfer it to 2D to reduce the computational cost of simulating the mechanical behavior of the microstructures. The resulting meshes are subjected to monotonic loading in random directions to create a dataset. Since the HyPRNN is conditioned on fewer parameters than before, we create a smaller dataset consisting of 512 samples, 256 used for training, and 128 each for validation and testing. The dataset is visualized in Figure 4.14. An energy-based HyPRNN is trained on this dataset.



(a) Pellet fraction: 0% (b) Pellet fraction: 30% (c) Pellet fraction: 60% (d) Pellet fraction: 90%

Figure 4.13.: Generated 2D microstructures for various pellet fractions, all with an initial orientation of 0°.

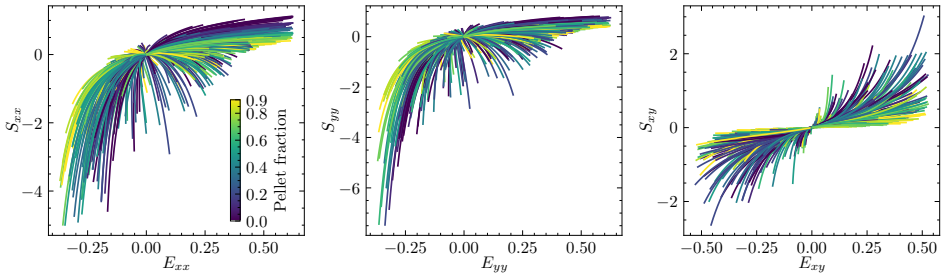
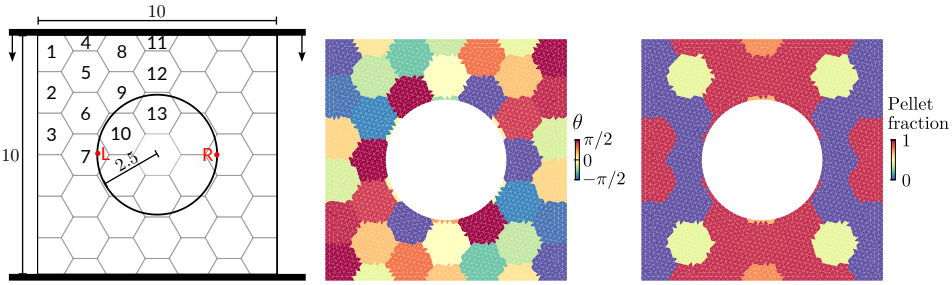


Figure 4.14.: Visualization of the dataset created. The colormap shows the influence of different pellet fractions on the mechanical response.

4.6.2. Deformation optimization

Next, we study whether the manufacturing variables can lead to a noticeable difference in a sample's deformation. A compression test is simulated for a square sample with a hole; the setup is visualized in Figure 4.15a. The objective is to minimize the horizontal deformation of the hole, which we denote as bulging and quantify as: $u_x^R - u_x^L$, where R and L refer to the respective nodes in Figure 4.15a. The top plate is gradually lowered for a total deformation of 20% of the sample height. This loading is imposed with a contact model using the penalty method in order to replicate a compression test and allow for separation from the plate. Since creating a continuous grading is challenging in practice, we instead partition the domain into hexagonal regions. Using symmetry, this results in 13 regions for a total of 26 parameters to optimize. The optimizer settings are identical to those of Section 4.5.4.



(a) The thirteen parametrized regions are numbered.

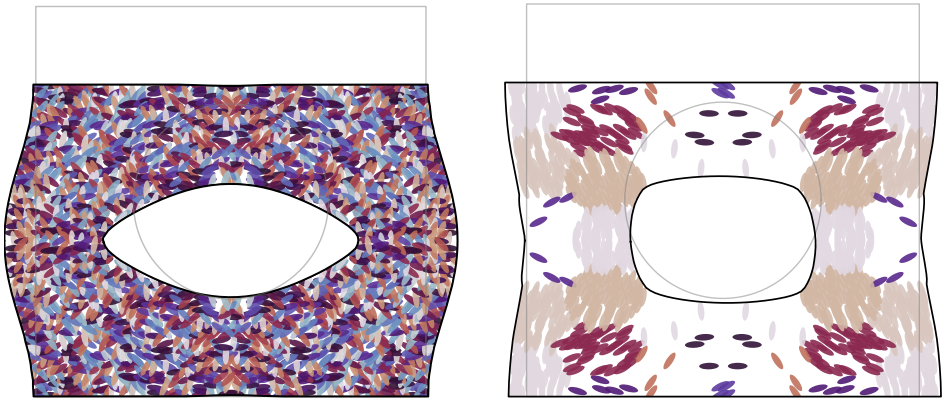
(b) Optimized θ

(c) Optimized pellet fraction

Figure 4.15.: Simulation setup and results of the deformation optimization.

We compare the optimized result to a baseline case with 0% pellet fraction and a random orientation in each element. Figures 4.15b and 4.15c show the optimized woodchip orientation and pellet fraction. The resulting microstructure distribution and deformations are visualized together in Figure 4.16. A clear difference is observed, with the baseline bulging significantly outward, whereas the optimized result bulges slightly inward. The optimized pattern shows an X-like structure that forms bands from the corners of the sample to the L and R points. Next to the bands there are parts with in-

termediate pellet fraction. The optimized pattern of the orientation is far from trivial. To verify the effectiveness of the orientation, we rerun the simulation with the optimized pellet fraction but a fixed orientation of θ=90°, which leads to a worse result with a bulge of 0.689. This example demonstrates how a HyPRNN can be used to find the manufacturing variables that lead to a desired mechanical behavior.



(a) Baseline: bulge = 1.528.

(b) Optimized: bulge = -0.287.

Figure 4.16.: Deformed results from the baseline and optimized compression test. The ellipses represent both manufacturing variables: their sparsity depends (inversely) on the pellet fraction, and their orientation directly reflects their deposition orientation and deformation. Note that the size of the ellipses does not reflect the scale of the woodchips, for which we assume a separation of scales.

4.7. Conclusion

In order to develop a surrogate model for a range of different microstructures, we propose a HyPRNN: a PRNN conditioned on microstructural variables using a hypernetwork. We compared two finite-strain HyPRNN formulations, namely a flexible nonlinear encoder and an energy-based formulation that maps strains linearly in logarithmic space. Both formulations predict zero stress in the undeformed state and guarantee valid deformation tensors for the embedded material models. The energy-based formulation is thermodynamically consistent. The models performed similarly when trained on large datasets, but the stronger physics-based inductive bias of the energy HyPRNN resulted in lower errors on test samples and in an FE^2 simulation. Material parameters such as the shear modulus can be varied directly in the embedded constitutive models without retraining, though less effectively with the nonlinear encoder. Geometric properties, such as the volume fraction, are passed as inputs to the hypernetwork. We handled orientations of the microstructure by training with one fixed orientation, and then pre- and post-processing the surrogate's inputs and outputs to account for the orientation. This conditioned surrogate model enabled graded multiscale simulations. For a simple 3-point bending experiment of a graded multiscale beam, the surrogate-based simulations reduced the computation time from 5900

seconds to 2 seconds, while maintaining high accuracy. By performing a multiscale optimization, we showed the ability to find a material grading that significantly reduces peak stresses.

We circumvent the need for an additional model to parametrize microstructural geometries (e.g., a conditional variational autoencoder or a conditional denoising diffusion model) and instead condition directly on the manufacturing variables used to generate the microstructure. Specifically, we conditioned the HyPRNN on inputs from a discrete-element simulation that mimics the deposition of woodchips and mycelium pellets. The resulting model was used in an optimization loop to substantially alter the deformation pattern of a sample under compression. The numerical examples in this work are limited to path-independent materials in 2D. Given the success of PRNNs in capturing more complex constitutive behavior such as elasto-viscoplasticity and debonding, a natural extension is to test our HyPRNN formulation with those material models. Furthermore, the material models and optimized structures should be calibrated and validated with physical experiments. Overall, this work highlights the necessity of microstructure-aware data-driven surrogates for multiscale simulations. Furthermore, we demonstrate the potential that precise control of microstructural variables can bring.

A. Hyperparameter configurations

In Table 4.3 we present the hyperparameters used to create the learning curves and compare the model performance. A wider range was explored in earlier prototyping. We did not perform a full grid sweep over the possible settings for each model, but followed a more manual iterative process. For example, for the neural network, we found that ReLU performed worse than the other activations, and therefore did not try all model sizes with it. For the hypernetwork, we further considered an alternative setup that scaled the hidden layer sizes depending on the number of its outputs (the number of encoder weights), but found no gain.

In Figure 4.17 we highlight learning curves for models with woodchip material points, one for the nonlinear and one for the energy PRNN. The woodchip material points cause significantly worse losses in the low-data regime. Given sufficient data, the models learn to make good predictions, but are still among the worst relative to other configurations.

The influence of the number of material points on the test loss is visualized in Figure 4.18. The performance is remarkably similar across a wide range of values. This insensitivity is beneficial for model selection.

B. FE^2 validation stress paths

In Figure 4.20 we plot the stress strain paths of several quadrature points from during the simulations in Section 4.5.3. The selected quadrature points are shown in Figure 4.19. We omit the results of the NN (32) surrogate, as its errors are high, and including its paths requires a substantial increase of the axes bounds, making the other paths

Table 4.3.: Hyperparameter sweep options for the various models. The neural network hidden layer sizes are specified by the number of neurons of the hidden layer, excluding the input and output layers. Scalar material point values reflect the number of mycelium material points, the [6,3] array reflects 6 mycelium material points, and 3 woodchip material points.

Hyperparameter	HyPRNN Nonlinear	HyPRNN Energy	NN
Material points	2, 3, 6, 12, 24, [6,3]	6, 12, 18, 24, [12,6]	–
Hypernetwork activations	sigmoid, ReLU	sigmoid, ReLU	–
Hypernetwork hidden sizes	(8), (8,8,8)	(4), (8), (12), (24), (8,8)	–
Encoder activations	softplus, sigmoid, ReLU	–	–
Encoder hidden sizes	(8), (8,8,8), (16), (16,16,16)	–	–
NN activations	–	–	sigmoid, tanh, ReLU
NN hidden sizes	–	–	(8,8), (16,16), (16,16,16), (32,32,32), (64,64), (64,64,64), (64,64,64,64), (128,128,128)

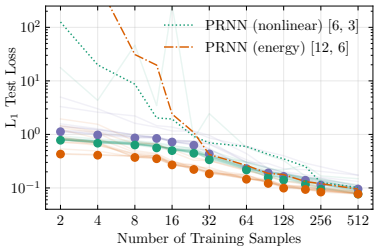
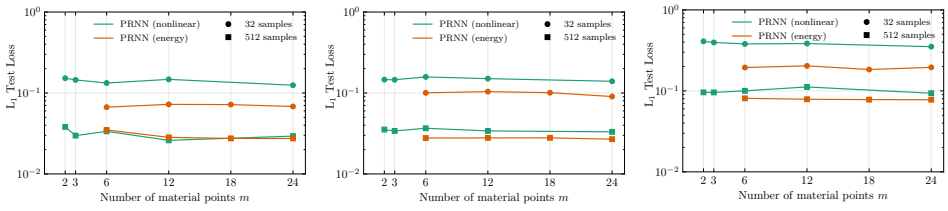


Figure 4.17.: Learning curves on $\mathcal{D}^{all}$ with two models that embed woodchip material points highlighted.



(a) $\mathcal{D}^{mat}$ (b) $\mathcal{D}^{geo}$ (c) $\mathcal{D}^{all}$

Figure 4.18.: Influence of the number of fictitious material points on the test loss, given either 32 or 512 training samples. Results are averaged over 10 training runs per configuration.

more challenging to compare. This figure clearly demonstrates how the NN is not constrained to have zero stresses in the undeformed state, showing a significant bias. All PRNN models follow the ground truth response.

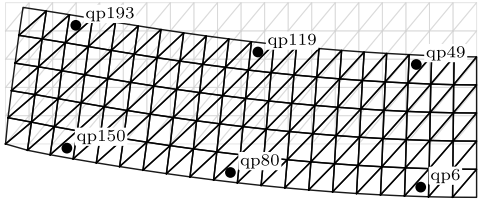


Figure 4.19.: Selected quadrature points.

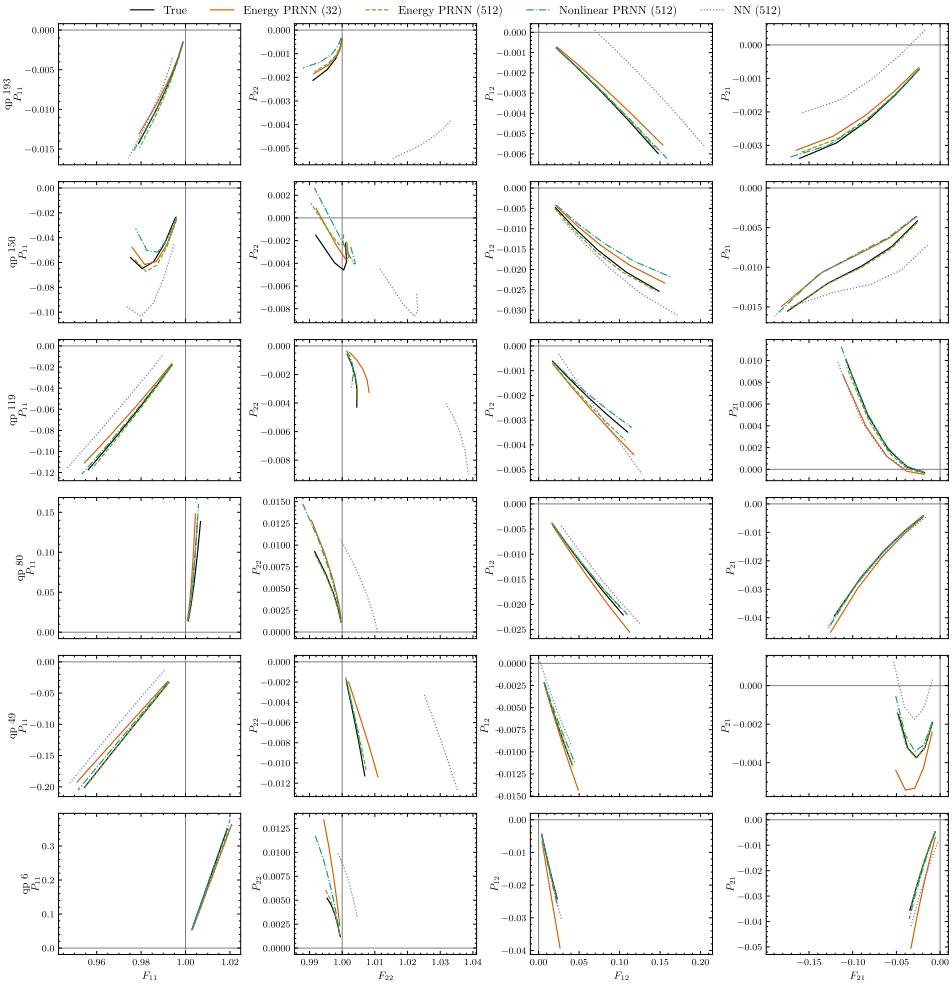


Figure 4.20.: Stress strain paths for selected quadrature points in the 3-point bending beam.

5
CONCLUSION

The aim of this dissertation was to make multiscale finite element simulations tractable using data-driven techniques. Data-driven surrogates can be much faster to evaluate, but there are open challenges. Training an accurate surrogate model requires a large training data set that is expensive to obtain, and this can itself become a computational bottleneck. For surrogate models to be used in practice, they need to be reliable. The microstructural design variables should appear as explicit variables in the surrogate model to study their influences. Finally, the added benefit of multiscale simulations in real-life material design needs to be demonstrated.

Rather than exploring a single method in depth, this work proposes various methods, each tackling these problems from a different angle. In Chapter 2, graph neural networks (GNNs) were used as surrogate models that approximate the microscale response, rather than only the homogenized values. This is a high-fidelity approach that enables accounting for microscale variations while using a data-driven surrogate model that accelerates the simulation. Chapter 3 focused on maintaining an accurate solution even if the surrogate is only accurate in a limited domain. By using a probabilistic surrogate in order to obtain its prediction uncertainty, the reliability of the surrogate-based simulation is increased. In principle, this approach can be wrapped around any surrogate model. Then, in Chapter 4, the ability to use surrogate-based multiscale models in optimizing a structure was demonstrated. Specifically, a physically recurrent neural network (PRNN) was used to replace the microscale model while accounting for microstructural variations. The inductive bias in the network enabled accurate predictions for a wide range of parameters with little training data.

Each method is a hybrid approach that combines parts of classical micromechanical models with data-driven surrogates, rather than having one replace the other. Together, they show that microstructure-aware multiscale simulations can be made tractable. Here, we will list the main contributions of the research. Then, future research directions to overcome current limitations are recommended and a perspective is given on the broader direction of data-driven computational mechanics.

5.1. Contributions

The contributions made in this work are summarized as follows:

- A graph-based surrogate was introduced that predicts the microscopic strain field from the macroscopic strain and an RVE mesh. An embedded microscopic constitutive model evaluates the microscopic stress, which is then homogenized to recover the macroscopic stress. In contrast to alternative surrogates, detailed full-field microscale information is retained.

- Various training strategies for the graph neural network were compared, which showed the benefit of including the microscopic constitutive model during training. Although the microscopic stress is fully determined by the predicted microscopic strain, including the stress in the loss function improved not only the stress prediction but also the strain prediction itself.

- Surrogate models are typically trained on data generated from RVE geometries. As the graph neural network can handle arbitrary meshes, it was shown that the graph neural network can be trained on small, cheaper to evaluate micromodels, and still predict larger ones. This could thus reduce the computation cost of data generation.

- To maintain accurate multiscale simulations when strains occur that are used outside the surrogates' training distribution, an adaptive approach was introduced in which the prediction uncertainty of a probabilistic surrogate triggers a switch back to the high-fidelity model. This enabled accurate simulations for surrogates trained on very little data.

- The stability of this adaptive approach was improved with an uncertainty-driven phase-field, where each phase represented the constitutive model in use.

- Previous work on PRNNs considered finite strains, but the applied deformations were moderate. Aiming to capture geometric nonlinearity, a nonlinear encoder was developed with increased flexibility. An energy-based formulation was also modified to operate on the true strain so that it accounts for large deformations.

- Surrogates are typically trained for a single, fixed microstructure, and therefore cannot account for varying underlying microstructures. To enable this, PRNNs were conditioned on microstructural parameters using a hypernetwork, resulting in a flexible but data-efficient surrogate.

- Finally, we picked up to the mycelium example from Chapter 1 and its many underlying challenges. The microscale grading of a mycelium-woodchip composite was optimized using a hypernetwork PRNN to significantly reduce peak stress. As precisely controlling the microscale is difficult for many materials, we further showed that we can condition directly on manufacturing variables, avoiding the need to parametrize complex microstructural geometries. This demonstrated that optimization across multiple scales is tractable.

5.2. Future research directions

Data-driven methods are very popular in research, and only time will tell to what extent they will be integrated within our computational frameworks. Here, some recommendations are given for future work. These start with directions that improve the methods discussed in this thesis, before zooming out to broader trends and challenges in the field.

5.2.1. Improving presented methods

The presented methods in this dissertation have limitations that future work should seek to surmount. Specific potential directions for future research are listed below.

Graph neural networks

- **Simplicity** A significant limitation of using GNNs in practice is their complexity, with dozens of hyperparameters to tweak. Future efforts should investigate creating recipes for robust hyperparameter settings with minimal free variables. To speculate, such a recipe could take the form of having: the number of message-passing layers depend on a physical quantity, such as the mesh size; the networks are made wide and expressive, regularized by a single knob; and the batch size is tied to the number of training samples.
- **Rotational equivariance** Defining the edge features that provide information about the closest surrounding voids in Cartesian (x, y) coordinates removed rotational equivariance from the model. An alternative measure, such as simply using the total distance to each void, gives up some expressivity but would restore equivariance and therefore abide by physics and simplify the training loss landscape.
- **Long-range message passing** For more complex problems, the used message passing can become limiting. Alternative multiscale graphs can better capture long range dependencies without needing excessive message passing layers (e.g., Fortunato *et al.*, 2022).
- **Transfer to unseen material properties** Similar to in PRNNs, the properties of embedded constitutive models can in theory easily be modified. Future research should find the extent to which predictions stay accurate when these properties are varied post-training.

Mixing constitutive models

- **FE^2 simulations** The study should be repeated in an actual multiscale setup. This could surface limitations related to the GP accuracy as well as the stability of the multiscale simulation with the added complexity of the staggered updating approach.
- **Probabilistic physics-based surrogates** A more expressive surrogate, such as a physics-embedded one, is needed to capture more complex constitutive models. Techniques such as ensembles or Monte Carlo dropout (Gal and Ghahramani, 2016) yield neural network uncertainty estimates, but these estimates are less-well calibrated than a GP posterior (Foong *et al.*, 2019), and their assumptions

might not be valid for physics-embedded architectures with strong inductive biases. Future work should investigate this effect, and develop new methods to quantify the prediction uncertainty of physics-based surrogate models.

- **Surrogate-agnostic opposing force** The phase-field hyperparameter of the opposing force b, which acts as a threshold that the uncertainty needs to overcome, depends on the specific surrogate used. An alternative formulation should seek to normalize the surrogate uncertainty in a consistent way to alleviate this dependency.
- **More phases** For complex problems, more than two phases could be used to allow having several specialized surrogate models in one domain.

Graded multiscale simulations

- **PRNN material models** It was found that embedding the woodchip material models in the PRNN degraded its performance in the low-data regime. This counters some of the intuition of PRNNs to include all nonlinear material models, and its cause should be investigated.
- **Surrogate accuracy** The nonlinear HyPRNN has a similar training error to the energy HyPRNN for 512 training samples, but performs worse in the multiscale simulation. This can be caused by accuracy for small strains, tangent-stiffness quality, and solution drift. Further work should isolate these phenomena to better understand their causes and potential remedies.
- **Gradient-based optimization** A derivative-free optimizer was used for up to 26 parameters. Since the surrogate is fully differentiable, an adjoint optimization could be used to efficiently propagate derivatives through the entire simulation, enabling a faster and finer optimization.
- **Geometry optimization** The macroscale geometry was fixed, but could be parametrized and included in the optimization.

5.2.2. Recommendations on surrogate modeling in general

This section discusses some limitations and high-level recommendations for surrogate modeling in the broader computational mechanics field. One observation is that many surrogate models accelerate multiscale simulations on academic benchmarks, yet few see adoption in industry. Academics should actively seek collaborations with industry where they can demonstrate the current state-of-the-art methods. This shows what is now possible and can raise industrial desires, while simultaneously highlighting where current methods still fall short. The recommendations are listed below.

Demonstrating reliability Data-driven surrogates are models, and therefore not always accurate. Misclassifying a cat as a dog is harmless, but producing an incorrect stress field for a load-bearing structure is not. Physics-based models can improve robustness, and uncertainty quantification can detect out-of-distribution responses. Two largely unexplored techniques that could significantly increase trust are robustness testing and mechanistic interpretability.

Robustness testing analyzes the failure modes of a model by probing it with challenging

scenarios. For instance, the hidden state of an RNN can drift when loading stalls for many iterations, and the response becomes nonsensical if loading continues afterward. These tests should outline the boundaries within which the model performs well, and identify the scenarios to avoid.

Mechanistic interpretability aims to understand the internal workings of neural networks. Recent work has identified the inner workings of parts of convolutional neural networks and large language models (Gurnee *et al.*, 2026; Meng *et al.*, 2022; Olah *et al.*, 2020). Some work on understanding data-driven models for material models shows how the RNN hidden state stores plastic strain in its memory (Pati and Mallikarjunaiah, 2025). While still very early, untangling the mechanisms that a data-driven model learns would let us calibrate our trust to what it has actually learned.

Increasing complexity As discussed in the introduction of this thesis, increasing tractability should not come at the cost of reducing fidelity. However, many current data-driven approaches are designed to capture a single mechanical phenomenon. Future work should focus on incorporating physics from multiple sources, such as thermo-chemo-mechanical coupling.

Smarter data generation Adaptive sampling and multi-fidelity data techniques could further alleviate data-related bottlenecks. Two-stage runs could be implemented, where the macroscale problem is first simulated with a simple constitutive model. A subset of the resulting stress-strain paths can then be used to generate microscale data to train the surrogate used in the second simulation. This way, expensive data is only generated in the space relevant for the simulation. This process could be repeated, iteratively generating more relevant data and a better surrogate model.

Multiscale benchmarks A shared benchmark could make it easier to compare the vast number of surrogate models. It would need to contain several categories, with various types of materials and loading conditions. This highlights which surrogate is fit for which purpose, and what its data requirements and extrapolation capabilities are.

Foundation models While writing this dissertation, foundation models have gained significant attention, especially in the context of large language models but also for physical systems (Herde *et al.*, 2024; McCabe *et al.*, 2023). Because of this, a common question is whether foundation models for computational mechanics are promising, or we should instead focus on task-specific physics-based models. Relevant to this discussion is "The bitter lesson", an observation from computer science that states: *general methods that leverage computation ultimately outperform models based on human-knowledge* (Sutton, 2019), and which holds for machine learning models ranging from vision to chess. If the bitter lesson holds true for computational mechanics, then foundation models are likely to succeed.

The main challenge for foundation models is the sheer diversity of problems in computational mechanics. Boundary conditions, failure mechanisms, multiscale and multi-physics couplings all complicate the search for a unified "language" to operate on. This variety of problems also causes a shortage of data, whereas the bitter lesson relies on an abundance of data. Despite efforts to overcome this limitation, such as the Genesis Mission (The White House, 2025), foundation models that can handle the diversity of problems remain out of reach. For the foreseeable future, domain-specific models

with built-in human-knowledge created for a specific application in mind will outperfom general foundation models in computational mechanics.

Unified computational ecoystem Computational mechanics would benefit greatly from a shared, open-source software ecosystem that was widely adopted. This would allow new methods to be released with easy-to-install packages alongside their papers, increasing their adoption and fair benchmarking, and avoid repeated re-implementation of the same methods on in-house software. The main challenge is catering to the wide variety of interests in the community, ranging from complex physical phenomena to differentiable solvers to adaptive meshing algorithms.

ACKNOWLEDGEMENTS

During most of my master's, I did not expect to pursue a PhD. Although I was studying civil engineering, I wanted to learn more about machine learning. That curiosity turned into an MSc thesis I thoroughly enjoyed, and by the end of it my appetite for research had only grown enough to continue with a PhD. Looking back now, I can see how far I have come since, and this section is for the people who helped me get here.

First, I want to thank my daily supervisor and co-promotor, **Iuri**. You were one of the main reasons I enjoyed my MSc thesis so much, and I feel lucky to have kept working with and learning from you throughout my PhD. You live and breathe science, and your enthusiasm is contagious. I also value the friendship we built outside of work. I doubt many PhD students can say they have played Dungeons and Dragons with their supervisor.

Next, I want to thank my promotor, **Frans**. There were many meetings where I sought advice on choosing between two options and left with a better third one. I especially appreciate how you could dive into the details without losing sight of the bigger picture. Those meetings were not only vital to the work, but often also filled with laughter.

I also want to thank the members of my doctoral committee, **Bert Sluys**, **Matthijs Langelaar**, **Jeremy Bleyer**, **Ondřej Rokoš**, and **Tito Andriollo**, for taking the time to read and evaluate my work.

Much of my work would have been far less interesting without the people I collaborated with. **Steve**, thank you for welcoming me to Columbia University, making me feel part of your group, and sharing your endless stream of ideas when we brainstormed. **Tito**, our chance meeting showed me that conferences really can lead to collaborations, and it has been a joy to work with you. In the final stage of my PhD, I wanted to bring my computational models to life. **Kunal** and **Sarah**, thank you for showing me how complex and messy the real world is, which gave me a new appreciation for the reproducibility of computational simulations. I am also grateful for the chance to supervise students. **Lex**, **Jos**, **Rasmus**, and **Tobias**, it was a pleasure to work with you.

It is often said that a PhD is largely a solitary journey. That may be true for some, but it was far from my experience, mostly thanks to my wonderful colleagues. Calling you *colleagues* does not do you justice; *friends* is a much better fit. SLIMM Lab gang and office mates **Anne**, **Leon**, and **Marina**, starting around the same time meant we could always help each other out, and I really appreciate all your support. Thank you for distracting me. Spending every day in the office with you made work a lot more fun, and I am glad our friendship has carried on beyond work. The same goes for **Martin**, **Til**, and **Winston**. Thank you for the discussions at work and for always being up for spontaneous plans. **Debashis**, **Deborah**, **Dragan**, **Fanxiang**, **Hoessein**, **Koert**, **Nan**, **Pieter**, **Rafid**, **Renan**, **Sergio**, **Sijmen**, **Suman**, **Tom**, **Xinrui**, and many others from our own and neighboring groups, thank you for the lunches, CM meetings, and conferences we shared. My visit to Columbia University became quite eventful with protests on campus, and I am grateful to my peers **Antonina**, **Linda**, **Luca**, and **Nhon** for accompanying

me during that time.

Thank you, **Iris**, **Jacqueline**, and **Sandra**, as well as **Jaap**, for keeping everything on the sixth floor running smoothly. I have lost count of how many times I walked into your office to book a meeting room. I am also grateful to everyone who made the Delft AI Labs possible, and especially **Charlotte** and **Roos-Anne** for connecting us all. I thank everyone involved in the Delft High Performance Computing Centre, without whose cluster many of my simulations would still be running today. Whenever I got stuck on implementing something, **ptrblck** had usually answered my PyTorch question before I could ask it, and **dokken** patiently debugged the FEniCS code in my forum posts.

What kept me sane during these years was spending time on things other than work. Thank you to everyone in the EA and DAISI communities, especially **Jan**, **Annemieke**, **Paula**, **Rickard**, **Koen**, **Maurits**, and **Nils**, for making the PhD feel like a part-time job at times. So long, and thanks for all the fish. Anyone who knows me even a little knows that I love board games, and I want to thank **Bas**, **Francois**, **Hans**, **Karim**, **Koos**, **Niels**, and **Pim** for waiting for me to finish my turn. **Amarins**, **Bente**, **Daan**, **Lisa**, and **Tessa**, thanks for all the rondjes. **Nóra**, köszönöm bástye for the dippies, and feel free to put yourself in any category you like here.

Finally, I want to thank my family for their unconditional love. **Mam**, **Pap**, **Marit**, **Sander**, **Berend**, **Leonie**, **Felien**, and **Evan**, being with you always feels like home.

Joep Storm
October 2026

REFERENCES

Abdalla, H. M. A., D. Casagrande and L. Moro (Aug. 2020). 'Thermo-mechanical analysis and optimization of functionally graded rotating disks'. In: *Journal of Strain Analysis for Engineering Design* 55 (5-6), pp. 159–171.

Abueidda, D. W., M. Almasri, R. Ammourah, U. Ravaioli, I. M. Jasiuk and N. A. Sobh (2019). 'Prediction and optimization of mechanical properties of composites using convolutional neural networks'. In: *Composite Structures* 227, p. 111264.

Abueidda, D. W., S. Koric, N. A. Sobh and H. Sehitoglu (2021). 'Deep learning for plasticity and thermo-viscoplasticity'. In: *International Journal of Plasticity* 136, p. 102852.

Aldakheel, F., E. S. Elsayed, T. I. Zohdi and P. Wriggers (2023). 'Efficient multiscale modeling of heterogeneous materials using deep neural networks'. In: *Computational Mechanics*, pp. 1–17.

Alheit, B., M. Peirlinck and S. Kumar (2026). 'COMMET: orders-of-magnitude speed-up in finite element method via batch-vectorized neural constitutive updates'. In: *Computer Methods in Applied Mechanics and Engineering* 452, p. 118728.

Ammendolea, D., F. Greco, L. Leonetti and A. Pascuzzo (Feb. 2026). 'Prediction of anisotropic damage evolution in nacre-inspired composites by using a data-driven nonlinear homogenization approach'. In: *Composite Structures* 377. ISSN: 02638223. DOI: 10.1016/j.compstruct.2025.119869.

Balay, S., S. Abhyankar, M. F. Adams, S. Benson, J. Brown, P. Brune, K. Buschelman, E. M. Constantinescu, L. Dalcin, A. Dener, V. Eijkhout, J. Faibussowitsch, W. D. Gropp, V. Hapla, T. Isaac, P. Jolivet, D. Karpeev, D. Kaushik, M. G. Knepley, F. Kong, S. Kruger, D. A. May, L. C. McInnes, R. T. Mills, L. Mitchell, T. Munson, J. E. Roman, K. Rupp, P. Sanan, J. Sarich, B. F. Smith, S. Zampini, H. Zhang, H. Zhang and J. Zhang (2024). *PETSc Web page*. https://petsc.org/.

Baratta, I. A., J. P. Dean, J. S. Dokken, M. Habera, J. S. Hale, C. N. Richardson, M. E. Rognes, M. W. Scroggs, N. Sime and G. N. Wells (2023). *DOLFINx: the next generation FEniCS problem solving environment*. preprint. DOI: 10.5281/zenodo.10447666.

Baratta, I. A., J. P. Dean, J. S. Dokken, M. U. of Luxembourg; Rafinex Sarl HABERA, T. Jack; University of Luxembourg; Faculty of Science, M. (D. of Engineering (DoE); HALE, C. N. Richardson, M. E. Rognes, M. W. Scroggs, N. Sime and G. N. Wells (Dec. 2025). 'DOLFINx: The next generation FEniCS problem solving environment'. In: *Zenodo*.

Battaglia, P. W., J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner *et al.* (2018). 'Relational inductive biases, deep learning, and graph networks'. In: *arXiv preprint arXiv:1806.01261*.

Bayat, M., M. Saleem, B. B. Sahari, A. M. Hamouda and E. Mahdi (July 2008). 'Analysis of functionally graded rotating disks with variable thickness'. In: *Mechanics Research Communications* 35 (5), pp. 283–309.

Bishop, J. E., J. M. Emery, R. V. Field, C. R. Weinberger and D. J. Littlewood (2015). 'Direct numerical simulations in solid mechanics for understanding the macroscale effects of microscale material variability'. In: *Computer methods in applied mechanics and engineering* 287, pp. 262–289.

Black, N. and A. R. Najafi (Jan. 2023). 'Deep neural networks for parameterized homogenization in concurrent multiscale structural optimization'. In: *Structural and Multidisciplinary Optimization* 66 (1), pp. 1–25.

Bleyer, J. (2024). *dolfinx_materials: A Python package for advanced material modelling*. Version v0.3.0. URL: https://doi.org/10.5281/zenodo.13882183.

Boettinger, W. J., J. A. Warren, C. Beckermann and A. Karma (2002). 'Phase-field simulation of solidification'. In: *Annual review of materials research* 32.1, pp. 163–194.

Borkowski, L., C. Sorini and A. Chattopadhyay (2022). 'Recurrent neural network-based multiaxial plasticity model with regularization for physics-informed constraints'. In: *Computers & Structures* 258, p. 106678.

Bradbury, J., R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne and Q. Zhang (2018). 'JAX: composable transformations of Python+NumPy programs'. In: *http://github.com/jax-ml/jax*.

Brandstetter, J., D. Worrall and M. Welling (2022). 'Message passing neural PDE solvers'. In: *arXiv preprint arXiv:2202.03376*.

Cang, R., H. Li, H. Yao, Y. Jiao and Y. Ren (2018). 'Improving direct physical properties prediction of heterogeneous materials from imaging data via convolutional neural network and a morphology-aware generative model'. In: *Computational Materials Science* 150, pp. 212–221.

Cao, Y., M. Chai, M. Li and C. Jiang (2022). 'Bi-stride multi-scale graph neural network for mesh-based physical simulation'. In: *arXiv preprint arXiv:2210.02573*.

Caruana, R. (1994). 'Learning Many Related Tasks at the Same Time with Backpropagation'. In: *Advances in Neural Information Processing Systems* 7.

Chafia, Z., J. Yvonnet and J. Bleyer (Mar. 2025). 'A Data-Driven-based homogenization method to simulate the anisotropic damage of brittle heterogeneous structures'. In: *Computer Methods in Applied Mechanics and Engineering* 437, p. 117747. ISSN: 0045-7825. DOI: 10.1016/J.CMA.2025.117747.

Chandrasekhar, A., S. Sridhara and K. Suresh (Jan. 2023). 'Graded multiscale topology optimization using neural networks'. In: *Advances in Engineering Software* 175, p. 103359. ISSN: 0965-9978.

Chen, L.-Q. (2002). 'Phase-field models for microstructure evolution'. In: *Annual review of materials research* 32.1, pp. 113–140.

Cheng, G., X. Li, Y. Nie and H. Li (2019). 'FEM-Cluster based reduction method for efficient numerical prediction of effective properties of heterogeneous material in nonlinear range'. In: *Computer Methods in Applied Mechanics and Engineering* 348, pp. 157–184.

Davis, T. A. (2006). *Direct methods for sparse linear systems*. SIAM.

Delft High Performance Computing Centre (DHPC) (2022). *DelftBlue Supercomputer (Phase 1)*. https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase1.

Dornheim, J., L. Morand, H. J. Nallani and D. Helm (2024). 'Neural networks for constitutive modeling: From universal function approximators to advanced models and the integration of physics'. In: *Archives of Computational Methods in Engineering* 31.2, pp. 1097–1127.

Durodola, J. F. and O. Attia (May 2000). 'Deformation and stresses in functionally graded rotating disks'. In: *Composites Science and Technology* 60 (7), pp. 987–995.

Eshelby, J. D. (1957). 'The determination of the elastic field of an ellipsoidal inclusion, and related problems'. In: *Proceedings of the royal society of London. Series A. Mathematical and physical sciences* 241.1226, pp. 376–396.

Ferrari, M. (2024). 'A Comparison of Sparse Solvers for Severely Ill-Conditioned Linear Systems in Geophysical Marker-In-Cell Simulations'. In: *arXiv preprint arXiv:2409.11515*.

Fey, M. and J. E. Lenssen (2019). 'Fast Graph Representation Learning with PyTorch Geometric'. In: *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

Finn, C., P. Abbeel and S. Levine (July 2017). *Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks*. URL: https://proceedings.mlr.press/v70/finn17a.html.

Foong, A., D. R. Burt, Y. Li and R. E. Turner (2019). 'Pathologies of factorised gaussian and mc dropout posteriors in bayesian neural networks'. In: *stat* 1050.2, p. 24.

Fortunato, M., T. Pfaff, P. Wirnsberger, A. Pritzel and P. Battaglia (2022). 'Multiscale meshgraphnets'. In: *arXiv preprint arXiv:2210.00612*.

Frankel, A. L., C. Safta, C. Alleman and R. E. Jones (2022). 'Mesh-based graph convolutional neural networks for modeling materials with microstructure'. In: *Journal of Machine Learning for Modeling and Computing* 3.1.

Fritzen, F., M. Fernández and F. Larsson (2019). 'On-the-fly adaptivity for nonlinear two-scale simulations using artificial neural networks and reduced order modeling'. In: *Frontiers in Materials* 6, p. 75.

Fuhg, J. N., G. A. Padmanabha, N. Bouklas, B. Bahmani, W. C. Sun, N. N. Vlassis, M. Flaschel, P. Carrara and L. D. Lorenzis (Nov. 2024). 'A Review on Data-Driven Constitutive Laws for Solids'. In: *Archives of Computational Methods in Engineering 2024 32:3* 32 (3), pp. 1841–1883.

Gal, Y. and Z. Ghahramani (2016). 'Dropout as a bayesian approximation: Representing model uncertainty in deep learning'. In: *international conference on machine learning*. PMLR, pp. 1050–1059.

Gantenbein, S., E. Colucci, J. Käch, E. Trachsel, F. B. Coulter, P. A. Rühs, K. Masania and A. R. Studart (Dec. 2022). 'Three-dimensional printing of mycelium hydrogels into living complex materials'. In: *Nature Materials 2022 22:1* 22 (1), pp. 128–134.

Gao, H., M. J. Zahr and J.-X. Wang (2022). 'Physics-informed graph neural Galerkin networks: A unified framework for solving PDE-governed forward and inverse problems'. In: *Computer Methods in Applied Mechanics and Engineering* 390, p. 114502.

George, A. and E. Ng (1988). 'On the complexity of sparse QR and LU factorization of finite-element matrices'. In: *SIAM journal on scientific and statistical computing* 9.5, pp. 849–861.

Gerasimov, T. and L. De Lorenzis (2016). 'A line search assisted monolithic approach for phase-field computing of brittle fracture'. In: *Computer Methods in Applied Mechanics and Engineering* 312, pp. 276–303.

Geuzaine, C. and J. F. Remacle (Sept. 2009a). 'Gmsh: A 3-D 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.

Geuzaine, C. and J.-F. Remacle (2009b). 'Gmsh: A 3-D 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.

Ghane, E., M. Fagerström and M. Mirkhalaf (Sept. 2024). 'Recurrent neural networks and transfer learning for predicting elasto-plasticity in woven composites'. In: *European Journal of Mechanics - A/Solids* 107, p. 105378.

Ghavamian, F. and A. Simone (2019). 'Accelerating multiscale finite element simulations of history-dependent materials using a recurrent neural network'. In: *Computer Methods in Applied Mechanics and Engineering* 357, p. 112594.

Gladstone, R. J., H. Rahmani, V. Suryakumar, H. Meidani, M. D'Elia and A. Zareei (2023). 'GNN-based physics solver for time-independent PDEs'. In: *arXiv preprint arXiv:2303.15681*.

Goswami, S., C. Anitescu and T. Rabczuk (2019). 'Adaptive phase field analysis with dual hierarchical meshes for brittle fracture'. In: *Engineering Fracture Mechanics* 218, p. 106608.

Gupta, A., A. Bhaduri and L. Graham-Brady (2023). 'Accelerated multiscale mechanics modeling in a deep learning framework'. In: *Mechanics of Materials*, p. 104709.

Gurnee, W., E. Ameisen, I. Kauvar, J. Tarng, A. Pearce, C. Olah and J. Batson (2026). 'When models manipulate manifolds: The geometry of a counting task'. In: *arXiv preprint arXiv:2601.04480*.

Gurson, A. L. (1977). 'Continuum theory of ductile rupture by void nucleation and growth: Part I—Yield criteria and flow rules for porous ductile media'. In.

Hamdi, E. and E. Lejeune (2026). 'Towards robust surrogate models: Benchmarking machine learning approaches to expediting phase field simulations of brittle fracture'. In: *Computer Methods in Applied Mechanics and Engineering* 449, p. 118526.

Hamilton, W. L. (2020). 'Graph representation learning'. In: *Synthesis Lectures on Artifical Intelligence and Machine Learning* 14.3, pp. 1–159.

Hansen, N. and A. Ostermeier (2001). 'Completely derandomized self-adaptation in evolution strategies.' In: *Evolutionary computation* 9 (2), pp. 159–195.

Hashin, Z. and S. Shtrikman (1963). 'A variational approach to the theory of the elastic behaviour of multiphase materials'. In: *Journal of the Mechanics and Physics of Solids* 11.2, pp. 127–140.

Heidenreich, J. N., C. Bonatti and D. Mohr (Jan. 2024). 'Transfer learning of recurrent neural network-based plasticity models'. In: *International Journal for Numerical Methods in Engineering* 125 (1), e7357. DOI: 10.1002/NME.7357.

Herde, M., B. Raonić, T. Rohner, R. Käppeli, R. Molinaro, E. De Bezenac and S. Mishra (2024). 'Poseidon: Efficient foundation models for pdes'. In: *Advances in Neural Information Processing Systems* 37, pp. 72525–72624.

Hernandez, J. A., M. A. Caicedo and A. Ferrer (2017). 'Dimensional hyper-reduction of nonlinear finite element models via empirical cubature'. In: *Computer methods in applied mechanics and engineering* 313, pp. 687–722.

Islam, M. R., G. Tudryn, R. Bucinell, L. Schadler and R. C. Picu (Oct. 2017). 'Morphology and mechanics of fungal mycelium'. In: *Scientific Reports 2017 7:1* 7 (1), pp. 13070–. ISSN: 2045-2322.

Islam, M. R., G. Tudryn, R. Bucinell, L. Schadler and R. C. Picu (Dec. 2018a). 'Mechanical behavior of mycelium-based particulate composites'. In: *Journal of Materials Science* 53 (24), pp. 16371–16382. ISSN: 15734803.

Islam, M. R., G. Tudryn, R. Bucinell, L. Schadler and R. C. Picu (Dec. 2018b). 'Stochastic continuum model for mycelium-based bio-foam'. In: *Materials & Design* 160, pp. 549–556. ISSN: 0264-1275.

J. Ghaboussi J.H. Garrett Jr., X. W. (1991). 'Knowledge-Based modeling of material behavior with neural networks'. In: *Journal of Engineering Mechanics* 117.1, pp. 132–153.

Jacobs, R. A., M. I. Jordan, S. J. Nowlan and G. E. Hinton (1991). 'Adaptive mixtures of local experts'. In: *Neural computation* 3.1, pp. 79–87.

Jailin, C. and E. Baranger (Oct. 2025). 'Material-embedding Physics-Augmented Neural Networks: A first application to constitutive law parameterization'. In: *Computer Methods in Applied Mechanics and Engineering* 445, p. 118188. ISSN: 0045-7825.

Kalina, K. A., L. Linden, J. Brummund and M. Kästner (2023). 'FE ANN: an efficient data-driven multiscale approach based on physics-constrained neural networks and automated data mining'. In: *Computational Mechanics* 71.5, pp. 827–851.

Karniadakis, G. E., I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang and L. Yang (2021). 'Physics-informed machine learning'. In: *Nature Reviews Physics* 3.6, pp. 422–440.

Kim, C., J. Lee and J. Yoo (Dec. 2021). 'Machine learning-combined topology optimization for functionary graded composite structure design'. In: *Computer Methods in Applied Mechanics and Engineering* 387, p. 114158. ISSN: 0045-7825.

Kingma, D. P. and J. L. Ba (Dec. 2014). 'Adam: A Method for Stochastic Optimization'. In: *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. URL: https://arxiv.org/abs/1412.6980v9.

Klein, D. K., M. Fernández, R. J. Martin, P. Neff and O. Weeger (2022a). 'Polyconvex anisotropic hyperelasticity with neural networks'. In: *Journal of the Mechanics and Physics of Solids* 159, p. 104703.

Klein, D. K., R. Ortigosa, J. Martínez-Frutos and O. Weeger (2022b). 'Finite electroelasticity with physics-augmented neural networks'. In: *Computer Methods in Applied Mechanics and Engineering* 400, p. 115501.

Klein, D. K., R. Ortigosa, J. Martínez-Frutos and O. Weeger (2024). 'Nonlinear electroelastic finite element analysis with neural network constitutive models'. In: *arXiv preprint arXiv:2402.07007*.

Kovács, N., I. B. C. M. Rocha, F. P. van der Meer, C. Furtado and P. P. Camanho (2025). 'Uncertainty Quantification in Multiscale Modeling of Polymer Composite Materials Using Physically Recurrent Neural Networks'. In: URL: http://arxiv.org/abs/2504.11625.

Krokos, V., S. Bordas and P. Kerfriden (2022a). 'A Graph-based probabilistic geometric deep learning framework with online physics-based corrections to predict the criticality of defects in porous materials'. In: *arXiv preprint arXiv:2205.06562*.

Krokos, V., V. Bui Xuan, S. Bordas, P. Young and P. Kerfriden (2022b). 'A Bayesian multiscale CNN framework to predict local stress fields in structures with microscale features'. In: *Computational Mechanics* 69.3, pp. 733–766.

Lee, K. and K. T. Carlberg (2020). 'Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders'. In: *Journal of Computational Physics* 404, p. 108973.

Lejeune, E. (2020). 'Mechanical MNIST: A benchmark dataset for mechanical metamodels'. In: *Extreme Mechanics Letters* 36, p. 100659.

Li, M.-J., Y. Lian, Z. Cheng, L. Li, Z. Wang, R. Gao and D. Fang (2025). 'A clustering adaptive Gaussian process regression method: Response patterns based real-time prediction for nonlinear solid mechanics problems'. In: *Computer Methods in Applied Mechanics and Engineering* 436, p. 117669.

Li, Q., Z. Han and X.-M. Wu (2018). 'Deeper insights into graph convolutional networks for semi-supervised learning'. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 32. 1.

Li, T. (Feb. 2024). 'Micromechanics-informed parametric deep material network for physics behavior prediction of heterogeneous materials with a varying morphology'. In: *Computer Methods in Applied Mechanics and Engineering* 419, p. 116687.

Li, Z., N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart and A. Anandkumar (2020a). 'Fourier neural operator for parametric partial differential equations'. In: *arXiv preprint arXiv:2010.08895*.

Li, Z., N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart and A. Anandkumar (2020b). 'Neural operator: Graph kernel network for partial differential equations'. In: *arXiv preprint arXiv:2003.03485*.

Li, Z., N. Kovachki, K. Azizzadenesheli, B. Liu, A. Stuart, K. Bhattacharya and A. Anandkumar (2020c). 'Multipole graph neural operator for parametric partial differential equations'. In: *Advances in Neural Information Processing Systems* 33, pp. 6755–6766.

Linka, K., M. Hillgärtner, K. P. Abdolazizi, R. C. Aydin, M. Itskov and C. J. Cyron (2021). 'Constitutive artificial neural networks: a fast and general approach to predictive data-driven constitutive modeling by deep learning'. In: *Journal of Computational Physics* 429, p. 110010.

Lino, M., C. Cantwell, A. A. Bharath and S. Fotiadis (2021). 'Simulating Continuum Mechanics with Multi-Scale Graph Neural Networks'. In: *arXiv preprint arXiv:2106.04900*.

Liu, X., S. Tian, F. Tao and W. Yu (2021). 'A review of artificial neural networks in the constitutive modeling of composite materials'. In: *Composites Part B: Engineering* 224, p. 109152.

Liu, Z., M. Bessa and W. K. Liu (2016). 'Self-consistent clustering analysis: an efficient multi-scale scheme for inelastic heterogeneous materials'. In: *Computer Methods in Applied Mechanics and Engineering* 306, pp. 319–341.

Liu, Z., C. T. Wu and M. Koishi (Mar. 2019a). 'A deep material network for multiscale topology learning and accelerated nonlinear modeling of heterogeneous materials'. In: *Computer Methods in Applied Mechanics and Engineering* 345, pp. 1138–1168.

Liu, Z., C. Wu and M. Koishi (2019b). 'A deep material network for multiscale topology learning and accelerated nonlinear modeling of heterogeneous materials'. In: *Computer Methods in Applied Mechanics and Engineering* 345, pp. 1138–1168.

Logarzo, H. J., G. Capuano and J. J. Rimoli (2021). 'Smart constitutive laws: Inelastic homogenization through machine learning'. In: *Computer methods in applied mechanics and engineering* 373, p. 113482.

Lu, L., P. Jin and G. E. Karniadakis (2019). 'Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators'. In: *arXiv preprint arXiv:1910.03193*.

Lu, L., P. Jin, G. Pang, Z. Zhang and G. E. Karniadakis (2021). 'Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators'. In: *Nature machine intelligence* 3.3, pp. 218–229.

Luo, C. (2023). 'Fast staggered schemes for the phase-field model of brittle fracture based on the fixed-stress concept'. In: *Computer Methods in Applied Mechanics and Engineering* 404, p. 115787.

Madou, K. and J.-B. Leblond (2012). 'A Gurson-type criterion for porous ductile solids containing arbitrary ellipsoidal voids—II: Determination of yield criterion parameters'. In: *Journal of the Mechanics and Physics of Solids* 60.5, pp. 1037–1058.

Maia, M. A., K. A. Meyer, A. M. C. M. van Gils, I. B. C. M. Rocha and F. P. van der Meer (Oct. 2026). 'On architectural choices for interpretability and thermodynamic consistency in Physically Recurrent Neural Networks in the low-data regime'. In: URL: https://arxiv.org/abs/2610.04067v1.

Maia, M. A., I. B. Rocha, P. Kerfriden and F. P. van der Meer (Mar. 2023). 'Physically recurrent neural networks for path-dependent heterogeneous materials: Embedding constitutive models in a data-driven surrogate'. In: *Computer Methods in Applied Mechanics and Engineering* 407, p. 115934.

Maia, M., I. Rocha, D. Kovačević and F. P. van der Meer (Nov. 2024). 'Physically recurrent neural network for rate and path-dependent heterogeneous materials in a finite strain framework'. In: *Mechanics of Materials* 198, p. 105145.

Masi, F., I. Stefanou, P. Vannucci and V. Maffi-Berthier (2021). 'Thermodynamics-based Artificial Neural Networks for constitutive modeling'. In: *Journal of the Mechanics and Physics of Solids* 147, p. 104277.

Maurizi, M., C. Gao and F. Berto (2022). 'Predicting stress, strain and deformation fields in materials and structures with graph neural networks'. In: *Scientific Reports* 12.1, p. 21834.

McBee, R. M., M. Lucht, N. Mukhitov, M. Richardson, T. Srinivasan, D. Meng, H. Chen, A. Kaufman, M. Reitman, C. Munck, D. Schaak, C. Voigt and H. H. Wang (Dec. 2021). 'Engineering living and regenerative fungal–bacterial biocomposite structures'. In: *Nature Materials 2021 21:4* 21 (4), pp. 471–478. ISSN: 1476-4660. DOI: 10.1038/s41563-021-01123-y. URL: https://www.nature.com/articles/s41563-021-01123-y.

McCabe, M., B. R.-S. Blancard, L. H. Parker, R. Ohana, M. Cranmer, A. Bietti, M. Eickenberg, S. Golkar, G. Krawezik, F. Lanusse *et al.* (2023). 'Multiple physics pretraining for physical surrogate models'. In: *arXiv preprint arXiv:2310.02994*.

Meng, K., D. Bau, A. Andonian and Y. Belinkov (2022). 'Locating and editing factual associations in gpt'. In: *Advances in neural information processing systems* 35, pp. 17359–17372.

Mozaffar, M., R. Bostanabad, W. Chen, K. Ehmann, J. Cao and M. Bessa (2019). 'Deep learning predicts path-dependent plasticity'. In: *Proceedings of the National Academy of Sciences* 116.52, pp. 26414–26420.

Nettersheim, I. H., N. S. Sotelo, J. C. Verdonk and K. Masania (Sept. 2024). 'Engineered living composite materials'. In: *Composites Science and Technology* 256, p. 110758. ISSN: 0266-3538. DOI: 10.1016/J.COMPSCITECH.2024.110758.

Nguyen-Thanh, C., V. P. Nguyen, A. de Vaucorbeil, T. K. Mandal and J.-Y. Wu (2020). 'Jive: An open source, research-oriented C++ library for solving partial differential equations'. In: *Advances in Engineering Software* 150, p. 102925.

Ohana, R., M. McCabe, L. Meyer, R. Morel, F. J. Agocs, M. Beneitez, M. Berger, B. Burkhart, S. B. Dalziel, D. B. Fielding *et al.* (2024). 'The well: a large-scale collection of diverse physics simulations for machine learning'. In: *Advances in Neural Information Processing Systems* 37, pp. 44989–45037.

Olah, C., N. Cammarata, L. Schubert, G. Goh, M. Petrov and S. Carter (2020). 'Zoom in: An introduction to circuits'. In: *Distill* 5.3, e00024–001.

Paszke, A., S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.* (2019). 'Pytorch: An imperative style, high-performance deep learning library'. In: *Advances in neural information processing systems* 32.

Pati, C. and S. Mallikarjunaiah (2025). 'Neural Networks as Physics-Consistent Surrogates: An\textit {Explainable AI} Validation Framework for Learning Constitutive Relations'. In: *arXiv preprint arXiv:2512.02064*.

Peherstorfer, B., D. Butnaru, K. Willcox and H.-J. Bungartz (2014). 'Localized discrete empirical interpolation method'. In: *SIAM Journal on Scientific Computing* 36.1, A168–A192.

Pfaff, T., M. Fortunato, A. Sanchez-Gonzalez and P. W. Battaglia (2020). 'Learning mesh-based simulation with graph networks'. In: *arXiv preprint arXiv:2010.03409*.

Pitz, E. and K. Pochiraju (Aug. 2024). 'A neural network transformer model for composite microstructure homogenization'. In: *Engineering Applications of Artificial Intelligence* 134, p. 108622. ISSN: 0952-1976. DOI: 10.1016/J.ENGAPPAI.2024.108622.

Raissi, M., P. Perdikaris and G. E. Karniadakis (2019). 'Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations'. In: *Journal of Computational physics* 378, pp. 686–707.

Rajaraman, P. K., T. Manteuffel, M. Belohlavek and J. J. Heys (2017). 'Combining existing numerical models with data assimilation using weighted least-squares finite element methods'. In: *International Journal for Numerical Methods in Biomedical Engineering* 33.1, e02783.

Rao, C. and Y. Liu (Nov. 2020). 'Three-dimensional convolutional neural network (3D-CNN) for heterogeneous material homogenization'. In: *Computational Materials Science* 184, p. 109850.

Rasmussen, C. E. (2003). 'Gaussian processes in machine learning'. In: *Summer school on machine learning*. Springer, pp. 63–71.

Rocha, I. (2019). 'Numerical and experimental investigation of hygrothermal aging in laminated composites'. In.

Rocha, I., P. Kerfriden and F. van der Meer (2021). 'On-the-fly construction of surrogate constitutive models for concurrent multiscale mechanical analysis through probabilistic machine learning'. In: *Journal of Computational Physics: X* 9, p. 100083.

Rocha, I. B., P. Kerfriden and F. Van Der Meer (2023). 'Machine learning of evolving physics-based material models for multiscale solid mechanics'. In: *Mechanics of Materials* 184, p. 104707.

Rocha, I. B., F. P. van der Meer, L. A. Mororó and L. J. Sluys (2020). 'Accelerating crack growth simulations through adaptive model order reduction'. In: *International Journal for Numerical Methods in Engineering* 121.10, pp. 2147–2173.

Ryckelynck, D. (2009). 'Hyper-reduction of mechanical models involving internal variables'. In: *International Journal for numerical methods in engineering* 77.1, pp. 75–89.

Saad, Y. (2003). *Iterative methods for sparse linear systems*. SIAM.

Schyck, S., M. Ablonczy, S. Patranabish and K. Masania (May 2026). 'Shaping of Biohybrid Functional Living Materials'. In: *Advanced Functional Materials* 36 (38), e30836.

Seymour, L. M., J. Maragh, P. Sabatini, M. Di Tommaso, J. C. Weaver and A. Masic (2023). 'Hot mixing: Mechanistic insights into the durability of ancient Roman concrete'. In: *Science advances* 9.1, eadd1602.

Simo, J. and R. Taylor (1986). 'A return mapping algorithm for plane stress elastoplasticity'. In: *International Journal for numerical methods in engineering* 22.3, pp. 649–670.

Smilauer, V., V. Angelidakis, E. Catalano, R. Caulk, B. Chareyre, W. Chevremont, S. Dorofeenko, J. Duriez, N. Dyck, J. Elias *et al.* (2023). 'Yade documentation'. In: *arXiv preprint arXiv:2301.00611*.

Storm, J., I. B. Rocha and F. P. van der Meer (July 2024). 'A microstructure-based graph neural network for accelerating multiscale simulations'. In: *Computer Methods in Applied Mechanics and Engineering* 427, p. 117001.

Sutton, R. (Mar. 2019). *The Bitter Lesson*. http://www.incompleteideas.net/IncIdeas/BitterLesson.html. Accessed: 2026-05-27.

Takamoto, M., T. Praditia, R. Leiteritz, D. MacKinlay, F. Alesiani, D. Pflüger and M. Niepert (2022). 'Pdebench: An extensive benchmark for scientific machine learning'. In: *Advances in neural information processing systems* 35, pp. 1596–1611.

The White House (Nov. 2025). *Launching the Genesis Mission*. Executive Order, Presidential Actions. Accessed: 2026-06-04. URL: https://www.whitehouse.gov/presidential-actions/2025/11/launching-the-genesis-mission/.

United Nations (2015). *Transforming our world: the 2030 Agenda for Sustainable Development*. A/RES/70/1. URL: https://sdgs.un.org/2030agenda.

Vetter, J. S. (2017). 'Contemporary high performance computing'. In: *Contemporary high performance computing*. Chapman and Hall/CRC, pp. 3–11.

Vijayaraghavan, S., L. Wu, L. Noels, S. Bordas, S. Natarajan and L. A. Beex (2023). 'A data-driven reduced-order surrogate model for entire elastoplastic simulations applied to representative volume elements'. In: *Scientific Reports* 13.1, p. 12781.

Vlassis, N. N., R. Ma and W. Sun (2020). 'Geometric deep learning for computational mechanics part i: Anisotropic hyperelasticity'. In: *Computer Methods in Applied Mechanics and Engineering* 371, p. 113299.

Vlassis, N. N. and W. Sun (2023). 'Geometric learning for computational mechanics Part II: Graph embedding for interpretable multiscale plasticity'. In: *Computer Methods in Applied Mechanics and Engineering* 404, p. 115768.

Vogler, M., R. Rolfes and P. Camanho (2013). 'Modeling the inelastic deformation and fracture of polymer composites–Part I: Plasticity model'. In: *Mechanics of Materials* 59, pp. 50–64.

Wang, C., L.-y. Xu and J.-s. Fan (2020). 'A general deep learning framework for history-dependent response prediction based on UA-Seq2Seq model'. In: *Computer Methods in Applied Mechanics and Engineering* 372, p. 113357.

Wang, H., J. Tao, Z. Wu, K. Weiland, Z. Wang, K. Masania and B. Wang (June 2024). 'Fabrication of Living Entangled Network Composites Enabled by Mycelium'. In: *Advanced Science* 11 (24).

Wang, J.-J., C. Wang, J.-S. Fan and Y. Mo (2022). 'A deep learning framework for constitutive modeling based on temporal convolutional network'. In: *Journal of Computational Physics* 449, p. 110784.

Wei, T.-J., W.-N. Wan and C.-S. Chen (Apr. 2025). 'Deep Material Network: Overview, applications and current directions'. In: URL: http://arxiv.org/abs/2504.12159.

Wu, J.-Y., V. P. Nguyen, C. T. Nguyen, D. Sutula, S. Sinaie and S. P. Bordas (2020a). 'Phase-field modeling of fracture'. In: *Advances in applied mechanics* 53, pp. 1–183.

Wu, L., N. G. Kilingar, L. Noels *et al.* (2020b). 'A recurrent neural network-accelerated multi-scale model for elasto-plastic heterogeneous materials subjected to random cyclic and non-proportional loading paths'. In: *Computer Methods in Applied Mechanics and Engineering* 369, p. 113234.

Yoshinari, W. Kaysser, B. Rabin, A. Kawasaki and R. G. M. Ford (2013). *Functionally Graded Materials: Design, Processing and Applications*. Vol. 5. Springer Science & Business Media.

Yvonnet, J. and Q.-C. He (2007). 'The reduced model multiscale method (R3M) for the non-linear homogenization of hyperelastic media at finite strains'. In: *Journal of Computational Physics* 223.1, pp. 341–368.

Zhang, M., Z. Zhang, R. Zhang, Y. Peng, M. Wang and J. Cao (Nov. 2023). 'Lightweight, thermal insulation, hydrophobic mycelium composites with hierarchical porous structure: Design, manufacture and applications'. In: *Composites Part B: Engineering* 266, p. 111003.

Zheng, L., D. M. Kochmann and S. Kumar (Nov. 2024). 'HyperCAN: Hypernetwork-driven deep parameterized constitutive models for metamaterials'. In: *Extreme Mechanics Letters* 72.

Zheng, L., S. Kumar and D. M. Kochmann (Sept. 2021). 'Data-driven topology optimization of spinodoid metamaterials with seamlessly tunable anisotropy'. In: *Computer Methods in Applied Mechanics and Engineering* 383, p. 113894.

CURRICULUM VITÆ

Joep Storm

30-07-1995 Born in Delft, Netherlands.

Education

2021–2026 PhD. Computational Mechanics
Delft University of Technology
Thesis: Reliable and data-efficient machine learning for tractable multiscale material simulations
Promotor: Dr.ir. F.P. van der Meer
Copromotor: Dr. I.B.C.M. Rocha

2018–2021 MSc Structural Engineering
Delft University of Technology
Thesis: High-dimensional numerical optimization of fibre reinforced polymers with variational autoencoders and Bayesian optimization
Supervisors: Dr. I.B.C.M. Rocha & Dr.ir. T. Li Piani & Prof.dr.ir. L.J. Sluys

2017–2018 Premaster programme
Delft University of Technology

2012–2016 BSc Civil Engineering
The Hague University of Applied Sciences

List of Publications

4. **Storm, J.**, Rocha, I. B. C. M., S. Schyk, K. Masania, & van der Meer, F. P. (2026). Microstructure-Conditioned Surrogate Models for Graded Multiscale Optimization of Mycelium Composites *In preparation*.
3. Jakobsen, R. B., Kristensen, T. S., **Storm, J.**, Rocha, I. B. C. M., & Andriollo, T. (2026). Predicting plastic strain localization in porous solids using graph neural networks. *Available at SSRN 5828366*.
2. **Storm, J.**, Sun, W., Rocha, I. B. C. M., & van der Meer, F. P. (2025). Mixing data-driven and physics-based constitutive models using uncertainty-driven phase fields. *International Journal for Numerical Methods in Engineering, 126*(20), e70162.
1. **Storm, J.**, Rocha, I. B. C. M., & van der Meer, F. P. (2024). A microstructure-based graph neural network for accelerating multiscale simulations. *Computer Methods in Applied Mechanics and Engineering, 427*, 117001.

